```
                        ; ****************************************************************************
                        ; RETRODOS.SYS (PCDOS 7.1 Kernel) - RETRO DOS v5.0 by ERDOGAN TAN - 12/09/2023
                        ; ----------------------------------------------------------------------------
                        ; Last Update: 06/08/2025 - Retro DOS v5.0 (Modified PCDOS 7.1)
                        ; ----------------------------------------------------------------------------
                        ; Beginning: 26/12/2018 (Retro DOS 4.0), 01/10/2022 (Retro DOS 4.2)
                        ; ----------------------------------------------------------------------------
                        ; Assembler: NASM version 2.15
                        ; ----------------------------------------------------------------------------
                        ;      ((nasm retrodos5.s -l retrodos5.txt -o PCDOS.SYS -Z error.txt))
                        ; ----------------------------------------------------------------------------
                        ; Included binary file: IBMDOS7.BIN (Retro DOS 5.0 - PCDOS 7.1 IBMDOS.COM)
                        ; ****************************************************************************
                        ;

                        ; 12/09/2023 - Retro DOS v5.0 Kernel -dosbios- ('ibmbio7.s')
                        ; Modified from 'iosys6.s' (11/09/2023, Retro DOS v4.2 Kernel's IO.SYS) file
                        ; as below:
                        ;
                        ;    1) Retro DOS v4.2 IO.SYS is based on disassembled source code
                        ;   of MSDOS 6.21 IO.SYS, derived using MSDOS 6.0 source code.
                        ;
                        ;    2) Labels, names, comments, explanations and structure definitions
                        ;   about procedures and code details are almost entirely taken from
                        ;   the original MSDOS 6.0 source code, except for the details that
                        ;   Erdogan Tan personally experienced. Some of them are incompatible
                        ;   with PCDOS 7.1 code. But they have not been deleted to preserve
                        ;   the originality of the descriptions.)
                        ;
                        ;    3)'ibmbio7.s' contains the BIOSLOADER (MSLOADER) section located in
                        ;   the 1st 4 sectors of the IBMBIO.COM file on disk. This is a method
                        ;   from older DOS versions (3 sectors for MSDOS 6.22).
                        ;   The MSDOS/PCDOS boot sector code only reads these MSLOADER/BIOSLOADER
                        ;   sectors and transfers control to the MSLOADER/BIOSLOADER code.
                        ;   BUT!!! The Retro DOS v3 (& v5) boot sector code loads the entire
                        ;   MSDOS.SYS/PCDOS.SYS -combined- kernel file into memory at once.
                        ;   So, hence the Retro DOS boot sector code, 'retrodos5.s' file
                        ;       contains slightly different IO.SYS/IBMBIO.COM INITialization code
                        ;   than the original PCDOS/MSDOS. It does not include
                        ;   the MSLOADER/BIOSLOADER section. The 'retrodos5.s' and 'ibmbio7.s'
                        ;   files are almost identical except their INIT codes.)
                        ;
                        ; ('iosys6.s' has been converted to 'ibmbio7.s' and 'retrodos42.s' has been
                        ; converted to 'retrodos5.s'. 'ibmbio7.s' is IBMBIO.COM source code file
                        ; while 'retrodos5.s' is source code of Retro DOS v5 kernel file 'PCDOS.SYS'.
                        ; 'retrodos5.s' includes 'ibmdos7.bin' or IBMDOS.COM as binary file.)

                        ; ----------------------------------------------------------------------------

                        ; 20/12/2022 - Modifications for initiating IO.SYS by Retro DOS v2 boot sector
                        ;
                        ;          (Retro DOS v2 BS loads IO.SYS & MSDOS.SYS as single kernel file
                        ;          with name of 'MSDOS.SYS'. Retro DOS init code -for IO.SYS init-
                        ;          is different than original MSDOS IO.SYS LOADER and INIT code.)
                        ;
                        ;          ((RETRODOS.SYS/MSDOS.SYS can be loaded by a fake IO.SYS for
                        ;          using it with MSDOS 5.0 boot sector & as bootable MSDOS disk.
                        ;          For that, fake IO.SYS must load 'MSDOS.SYS' at 1000h:0000h.))
                        ;
                        ; 18/12/2022 - Modified MSDOS 5.0 IO.SYS (for using with MSDOS 5 boot sector)
                        ; 09/12/2022 - Multisection binary file format (BIOSDATA & BIOSCODE sections)
                        ; 01/10/2022 - Erdogan Tan (Istanbul)

                        ;Note: This code is a part of Retro DOS 4.0 kernel source code
                        ;     (as included binary, 'IOSYS5.BIN')
                        ;     Equivalent of MSDOS 5.0 IO.SYS, BIOSCODE and BIOSDATA and SYSINIT
                        ;                                             (except MSLOAD code)

                        ;------- Retro DOS v2 (v3) boot sector loads RETRODOS.SYS (MSDOS.SYS)
                        ;    at 1000h:0000h and loader (initialization) part of RETRODOS kernel
                        ;    moves IO.SYS (DOSBIOSCODE & DOSBIOSDATA, 'IOSYS5.BIN') to 70h:0000h.
                        ;    Then SYSINIT code to the next segment (46Dh for original MSDOS 5.0)..
                        ;    SYSINIT code relocates itself and DOSBIOSCODE and MSDOS.SYS
                        ;    (MSDOS5.BIN) according to request/setting in 'config.sys' file.

                        ; ----------------------------------------------------------------------------

                        ;============================================================================
                        ; Modified from 'retrodos3.s', Retro DOS v3.0 Kernel (IBMBIO.COM) Source code
                        ; by Erdogan Tan, 10/09/2018
                        ;============================================================================

                        ; MSBIO (IO.SYS 6.0) source files:
                        ;   MSBIO1.ASM,MSCHAR.ASM,MSDISK.ASM,MSDIOCTL.ASM,MSINT13.ASM,MSBIO2.ASM
                        ;   MSINIT.ASM,SYSINIT1.ASM,SYSCONF.ASM,SYSPRE.ASM,SYSINIT2.ASM
                        ;   SYSIMES.ASM,POWER.ASM,PTIME.ASM,MSEND.ASM

                        ;============================================================================
                        ; MSBIO
                        ;============================================================================
                        ;msbio1+mschar+msdisk+msdioctl+msint13+msbio2+
                        ;msinit+sysinit1+sysconf+syspre+sysinit2+sysimes+power+ptime+
                        ;msend,msbio,msbio;

                        ;============================================================================
                        ; RETRO DOS kernel versions by Erdogan Tan (2018-2022)
                        ;============================================================================

                        ;Retro DOS v1.0 == MSDOS 1.25 -- derived from MSDOS 1.25 source code
                        ;Retro DOS v2.0 == MSDOS 2.11 -- derived from MSDOS 2.11 source code
                        ;Retro DOS v3.0 == MSDOS 3.30 -- derived from MSDOS 3.3 & 6.0 source code
                        ;Retro DOS v4.0 == MSDOS 6.21 -- derived from MSDOS 6.0 source code (2019) (*)
                        ;Retro DOS v4.0 == MSDOS 5.0+ -- derived from MSDOS 6.0 source code (2022) (**)
                        ;Retro DOS v4.1 == MSDOS 5.0+ -- will be optimized -shortened- version (2023)
                        ;Retro DOS v4.2 == MSDOS 6.21 -- will be MSDOS 6.21 (6.22) compatible (2023)(?)
                        ;Retro DOS v5.0 == PCDOS 7.10 -- will be derived from IBM PCDOS 7.1 source code

                        ;(*) unfinished, draft, canceled (failed in 2019)
                        ;(**) MSDOS 5.0 IO.SYS & SYSINIT, MSDOS 5.0-6.22 mixed MSDOS.SYS (successed)
                        ;(?) MSDOS 6.21 IO.SYS & SYSINIT, MSDOS 6.21 MSDOS.SYS except doublespace

                        ;Disassembly: (reverse engineering via IDA Pro Free)

                        ;Retro DOS v1.0 <-- IBM PCDOS 1.1
                        ;Retro DOS v2.0 <-- IBM PCDOS 2.1 & MSDOS 2.11
                        ;Retro DOS v3.0 <-- IBM PCDOS 3.3 & MSDOS 3.3
                        ;Retro DOS v4.0 <-- MSDOS 6.21 ; 2018-2019 (*)
                        ;Retro DOS v4.0 <-- MSDOS 5.0 ; 2022 (**)
                        ;Retro DOS v5.0 <-- IBM PCDOS 7.1

                        ;----------------------------------------------------------------------------
                        ; MSDOS 6.21 IO.SYS (13/02/1994)
                        ;----------------------------------------------------------------------------

                        SECTOR_SIZE     equ     0200h           ; size of a sector
```

```
                                PAUSE_KEY       equ     7200h           ; scancode + charcode of PAUSE key
                                KEYBUF_NEXT     equ     041Ah           ; next character in keyboard buffer
                                KEYBUF_FREE     equ     041Ch           ; next free slot in keyboard buffer
                                KEYBUF          equ     041Eh           ; keyboard buffer data
                                LOGICAL_DRIVE   equ     0504h           ; linear address of logical drive byte
                                ;DOS_SEGMENT       equ     00BFh ; v1.1  ; segment in which DOS will run
                                DOS_SEGMENT equ     00C4h           ; Retro DOS v1.0 - 13/02/2018
                                BIO_SEGMENT     equ     0060h           ; segment in which BIO is running

                                ; 24/02/2018 (Retro DOS 2.0 - MSDOS 3.3 "DISKPRM.INC" - 24/07/1987)
                                ; The following structure defines the disk parameter table
                                ; pointed to by Interrupt vector 1EH (location 0:78H)

                                struc       DISK_PARMS
00000000 ??                     .DISK_SPECIFY_1:  resb    1
00000001 ??                     .DISK_SPECIFY_2:  resb    1
00000002 ??                     .DISK_MOTOR_WAIT: resb  1 ; Wait till motor off
00000003 ??                     .DISK_SECTOR_SIZ: resb    1       ; Bytes/Sector (2 = 512)
00000004 ??                     .DISK_EOT:    resb  1     ; Sectors per track (MAX)
00000005 ??                     .DISK_RW_GAP:         resb  1     ; Read Write Gap
00000006 ??                     .DISK_DTL:    resb 1
00000007 ??                     .DISK_FORMT_GAP:  resb  1 ; Format Gap Length
00000008 ??                     .DISK_FILL:   resb  1     ; Format Fill Byte
00000009 ??                     .DISK_HEAD_STTL:  resb  1 ; Head Settle Time (MSec)
0000000A ??                     .DISK_MOTOR_STRT: resb  1 ; Motor start delay
                                .size:
                                endstruc

                                ; 09/03/2019 - Retro DOS v4.0
                                ; ----------------------------------------------------------------------------
                                ; MSEQU.INC, MSDOS 6.0, 1991

                                ftoobig     equ     80h
                                fbig        equ     40h
                                ; 12/09/2023
                                fbigbig             equ     20h  ; Retro DOS 5.0 ; PCDOS 7.1 ; FAT32 FS flag
                                romstatus   equ     1
                                romread     equ     2
                                romwrite    equ     3
                                romverify   equ     4
                                romformat   equ     5

                                ; 26/12/2018 (Retro DOS 4.0 - MSDOS 6.0 "MSBDS.INC" - 1991)
                                ; ----------------------------------------------------------------------------
                                ; 24/02/2018 (Retro DOS 2.0 - MSDOS 3.3 "MSBDS.INC" - 24/07/1987)
                                ;
                                ;  BDS is the Bios Data Structure.
                                ;
                                ;  There is one BDS for each logical drive in the system. All the BDS's
                                ;  are linked together in a list with the pointer to the first BDS being
                                ;  found in START_BDS. The BDS hold various values important to the disk
                                ;  drive. For example there is a field for last time accesses. As actions
                                ;  take place in the system the BDS are update to reflect the actions.
                                ;  For example is there is a read to a disk the last access field for the
                                ;  BDS for that drive is update to the current time.
                                ;
                                ; Values for various flags in BDS.flags.
                                ;

                                fnon_removable          equ     01h       ;For non-removable media
                                fchangeline     equ     02h       ;If changeline supported on drive
                                return_fake_bpb     equ     04h   ; When set, don't do a build BPB
                                                                  ; just return the fake one
                                good_tracklayout    equ     08h   ; The track layout has no funny sectors
                                fi_am_mult      equ     10h       ;If more than one logical for this physical
                                fi_own_physical     equ     20h   ;Signify logical owner of this physical
                                fchanged        equ     40h       ;Indicates media changed
                                set_dasd_true           equ     80h       ; Set DASD before next format
                                fchanged_by_format  equ    100h           ;Media changed by format
                                ; MSDOS 6.0
                                unformatted_media   equ    200h   ;Fixed disk only

                                ;
                                ; Various form factors to describe media
                                ;

                                ff48tpi         equ     0
                                ff96tpi         equ     1
                                ffSmall         equ     2
                                ffHardFile      equ     5
                                ffOther         equ     7
                                ; MSDOS 6.0 ("MSBDS.INC", 1991)
                                ff288                   equ     9  ; 2.88 MB drive
                                ; Retro DOS v4.0 feature only !
                                ;ff144                  equ    10  ; 1.44 MB drive

                                ; 12/09/2023
                                ; Retro DOS v4 (MDOS 5.0-6.22) BDS structure
                                ; ----------------------------------------------------------------------------
                                ; 100 bytes

                                %if 0

                                ; 26/05/2019

                                struc       BDS     ; BDS_Type
                                .link:              resd 1          ; Link to next BDS
                                .drivenum:  resb 1          ; Physical drive number
                                .drivelet:  resb 1          ; DOS drive number

                                    ;We want to embed a BPB declaration here, but we can't initialize
                                    ;it properly if we do, so we duplicate the byte/word/dword architecture
                                    ;of the BPB declaration.
                                .BPB:
                                .bytespersec:       resw 1          ; bytes per sectors ; def = 512
                                .secperclus:        resb 1          ; sectors per cluster
                                .resectors: resw 1          ; reserved sectors
                                .fats:              resb 1          ; number of fats
                                .direntries:        resw 1          ; number of root directory entries
                                .totalsecs16:       resw 1          ; total sectors on medium
                                .media:             resb 1          ; media descriptor byte ; def = 0F8h
                                .fatsecs:   resw 1          ; number of fat sectors
                                .secpertrack:       resw 1          ; sectors per track
                                .heads:             resw 1          ; number of heads
                                ;.hiddensecs:       resw 1          ; hidden sectors
                                ; MSDOS 6.0
                                .hiddensecs:        resd 1          ; hidden sectors
                                .totalsecs32:       resd 1          ; big total sectors
                                ;
                                .fatsiz:    resb 1          ; flags...
                                .opcnt:             resw 1          ; open ref. count
                                ;.volid:    resb 12         ; volume ID of medium
                                .formfactor:        resb 1          ; form factor index
                                .flags:             resw 1          ; various flags ; def: 0020h
```

```
                                .cylinders: resw 1         ; number of cylinders
                                ;
                                .R_BPB:                    ; recommended BPB
                                .rbytespersec:     resw 1
                                .rsecperclus:      resb 1
                                .rresectors:       resw 1
                                .rfats:            resb 1
                                .rdirentries:      resw 1
                                .rtotalsecs16:     resw 1
                                .rmedia:    resb 1
                                .rfatsecs:  resw 1
                                .rsecpertrack:     resw 1
                                .rheads:    resw 1
                                .rhidsecs:  resd 1
                                .rtotalsecs32:     resd 1
                                .rreserved: resb 6         ; not used (reserved)
                                ;
                                .track:            resb 1          ; last track accessed on drive
                                .bdsm_ismini:
                                .tim_lo:    resw 1         ; time of last access. keep
                                .bdsm_hidden_trks:
                                .tim_hi:    resw 1         ; these contiguous.
                                .volid:            resb 12         ; volume id of medium
                                           ;db "NO NAME    ",0
                                .vol_serial:       resd 1  ; current volume serial number from boot record
                                .filesys_id:       resb 9  ; current file system id from boot record
                                           ;db "FAT12   ",0
                                .size:
                                endstruc

                                %endif

                                ; 12/09/2023 - Retro DOS 5.0 - PCDOS 7.1 (FAT32 compatible) BDS structure
                                ; ---------------------------------------------------------------------------
                                ; 150 bytes

                                %if 1

                                struc      BDS     ; BDS_Type
00000000 ????????               .link:             resd 1          ; Link to next BDS
00000004 ??                     .drivenum:  resb 1         ; Physical drive number
00000005 ??                     .drivelet:  resb 1         ; DOS drive number

                                    ;We want to embed a BPB declaration here, but we can't initialize
                                    ;it properly if we do, so we duplicate the byte/word/dword architecture
                                    ;of the BPB declaration.
                                .BPB:
00000006 ????                   .bytespersec:      resw 1          ; bytes per sectors ; def = 512
00000008 ??                     .secperclus:       resb 1          ; sectors per cluster
00000009 ????                   .resectors: resw 1         ; reserved sectors
0000000B ??                     .fats:             resb 1          ; number of fats
0000000C ????                   .direntries:       resw 1          ; number of root directory entries
0000000E ????                   .totalsecs16:      resw 1          ; total sectors on medium
00000010 ??                     .media:            resb 1          ; media descriptor byte ; def = 0F8h
00000011 ????                   .fatsecs16:        resw 1          ; number of fat sectors
00000013 ????                   .secpertrack:      resw 1          ; sectors per track
00000015 ????                   .heads:            resw 1          ; number of heads
00000017 ????????               .hiddensecs:       resd 1          ; hidden sectors
0000001B ????????               .totalsecs32:      resd 1          ; big total sectors
                                ; ---------- FAT32 extensions to BDS ---------- Retro DOS 5.0 ----------
0000001F ????????               .fatsecs32: resd 1         ; BPB_FATSz32   ; FAT32 FAT size in sectors
00000023 ????                   .extflags:  resw 1         ; BPB_ExtFlags  ; FAT32 Extended Flags
00000025 ????                   .fsver:            resw 1          ; BPB_FSVer    ; FAT32 volume version number
00000027 ????????               .rootdirclust:     resd 1          ; BPB_RootClus  ; FAT32 root dir's 1st clust num
0000002B ????                   .fsinfo:    resw 1         ; BPB_FSInfo   ; FAT32 FSINFO sector number
0000002D ????                   .bkbootsec: resw 1         ; BPB_BkBootSec ; FAT32 backup boot sector number
0000002F <res Ch>               .reserved:  resb 12        ; BPB_Reserved ; FAT32 reserved field = 0, 12 bytes
                                ; ------------------------------------------------
0000003B ??                     .fatsiz:    resb 1         ; flags...
0000003C ????                   .opcnt:            resw 1          ; open ref. count
0000003E ??                     .formfactor:       resb 1          ; form factor index
0000003F ????                   .flags:            resw 1          ; various flags ; def: 0020h
00000041 ????                   .cylinders: resw 1         ; number of cylinders
                                ;
                                .R_BPB:                    ; recommended BPB
00000043 ????                   .rbytespersec:     resw 1
00000045 ??                     .rsecperclus:      resb 1
00000046 ????                   .rresectors:       resw 1
00000048 ??                     .rfats:            resb 1
00000049 ????                   .rdirentries:      resw 1
0000004B ????                   .rtotalsecs16:     resw 1
0000004D ??                     .rmedia:    resb 1
0000004E ????                   .rfatsecs:  resw 1
00000050 ????                   .rsecpertrack:     resw 1
00000052 ????                   .rheads:    resw 1
00000054 ????????               .rhidsecs:  resd 1
00000058 ????????               .rtotalsecs32:     resd 1
                                ; ---------- FAT32 extensions to BDS ---------- Retro DOS 5.0
0000005C ????????               .rfatsecs32:       resd 1          ;
00000060 ????                   .rextflags: resw 1         ;
00000062 ????                   .rfsver:    resw 1         ;
00000064 ????????               .rrootdirclust:    resd 1          ;
00000068 ????                   .rfsinfo:   resw 1         ; default/initial value = -1
0000006A ????                   .rbkbootsec:       resw 1          ; default/initial value = -1
0000006C <res Ch>               .rreserved: resb 12        ; default value = 0
                                ; ------------------------------------------------
                                ;
00000078 ??                     .track:            resb 1          ; last track accessed on drive (def=-1)
                                .bdsm_ismini:
00000079 ????                   .tim_lo:    resw 1         ; time of last access. keep
                                .bdsm_hidden_trks:
0000007B ????                   .tim_hi:    resw 1         ; these contiguous.
0000007D <res Ch>               .volid:            resb 12         ; volume id of medium
                                           ;db "NO NAME    ",0
00000089 ????????               .vol_serial:       resd 1  ; current volume serial number from boot record
0000008D <res 9h>               .filesys_id:       resb 9  ; current file system id from boot record
                                           ;db "FAT12   ",0
                                .size:
                                endstruc

                                %endif
                                ; ---------------------------------------------------------------------------

                                ;The assembler will generate bad data for "size bds_volid",
                                ;so we'll define an equate here.

                                VOLID_SIZ   equ    12

                                ;bdsm_ismini       equ    bds_tim_lo     ; overlapping bds_tim_lo
                                ;bdsm_hidden_trks equ     bds_tim_hi     ; overlapping bds_tim_hi

                                max_mini_dsk_num equ 23    ; max # of mini disk ibmbio can support

                                ; 29/12/2018
```

```
                                ; Retro DOS v4.0
                                ;
                                ; MSDOS 6.0 - BOOTFORM.INC

                                BOOT_SIZE       EQU 512
                                EXT_BOOT_SIGNATURE  EQU    29h ; 41 ; Extended boot signature

                                %define BOOT_SIGNATURE    [BOOT_SIZE-2]

                                struc EBPB ; EXT_BPB_INFO
00000000 ????                   .BYTESPERSECTOR:    resw 1
00000002 ??                     .SECTORSPERCLUSTER: resb 1
00000003 ????                   .RESERVEDSECTORS:   resw 1
00000005 ??                     .NUMBEROFFATS:         resb 1
00000006 ????                   .ROOTENTRIES:          resw 1
00000008 ????                   .TOTALSECTORS:         resw 1
0000000A ??                     .MEDIADESCRIPTOR:   resb 1
0000000B ????                   .SECTORSPERFAT:        resw 1
0000000D ????                   .SECTORSPERTRACK:   resw 1
0000000F ????                   .HEADS:                resw 1
00000011 ????????               .HIDDENSECTORS:        resd 1
00000015 ????????               .BIGTOTALSECTORS:   resd 1
                                .size:
                                endstruc

                                ;EXT_PHYDRV, EXT_CURHD included in the header for OS2.
                                struc EXT_BOOT ; EXT_IBMBOOT_HEADER
00000000 ??????                 .JUMP:              resb 3
00000003 ????????????????       .OEM:               resb 8
0000000B <res 19h>              .BPB:               resb EBPB.size ; 25 bytes
00000024 ??                     .PHYDRV:    resb 1
00000025 ??                     .CURHD:             resb 1
00000026 ??                     .SIG:               resb 1
00000027 ????????               .SERIAL:    resd 1
0000002B <res Bh>               .VOL_LABEL: resb 11
00000036 ????????????????       .SYSTEM_ID: resb 8
                                .size:
                                endstruc

                                ; 12/09/2023
                                ; ----------------------------
                                ; Retro DOS v5.0 (PCDOS 7.1) - FAT32 Boot Sector Parameters

                                struc XBPB ; FAT32_BPB_INFO ; 12/09/2023
00000000 ????                   .BYTESPERSECTOR:    resw 1
00000002 ??                     .SECTORSPERCLUSTER: resb 1
00000003 ????                   .RESERVEDSECTORS:   resw 1
00000005 ??                     .NUMBEROFFATS:         resb 1
00000006 ????                   .ROOTENTRIES:          resw 1
00000008 ????                   .TOTALSECTORS:         resw 1
0000000A ??                     .MEDIADESCRIPTOR:   resb 1
0000000B ????                   .SECTORSPERFAT:        resw 1
0000000D ????                   .SECTORSPERTRACK:   resw 1
0000000F ????                   .HEADS:                resw 1
00000011 ????????               .HIDDENSECTORS:        resd 1
00000015 ????????               .BIGTOTALSECTORS:   resd 1
                                ;............ FAT32 ......  + 28
00000019 ????????               .FATSIZE32:     resd 1
0000001D ????                   .EXTFLAGS:      resw 1
0000001F ????                   .FSVER:                resw 1
00000021 ????????               .ROOTDIRCLUSTER:    resd 1
00000025 ????                   .FSINFOSECTOR:         resw 1  ; (offset from FAT32 bs)
00000027 ????                   .BACKUPBOOTSECTOR:  resw 1  ; (offset from FAT32 bs)
00000029 <res Ch>               .RESERVEDBYTES:        resb 12 ; (zero bytes)
                                .size:
                                endstruc

                                struc FAT32_EXT_BOOT ; FAT32_IBMBOOT_HEADER ; 12/09/2023
00000000 ??????                 .JUMP:              resb 3
00000003 ????????????????       .OEM:               resb 8
0000000B <res 35h>              .BPB:               resb XBPB.size ; 53 bytes (25+28)
00000040 ??                     .PHYDRV:    resb 1
00000041 ??                     .CURHD:             resb 1
00000042 ??                     .SIG:               resb 1
00000043 ????????               .SERIAL:    resd 1
00000047 <res Bh>               .VOL_LABEL: resb 11
00000052 ????????????????       .SYSTEM_ID: resb 8
                                .size:
                                endstruc

                                ; ----------------------------

                                ; 23/03/2018

                                ;STATIC REQUEST HEADER (DEVSYM.INC, MSDOS 6.0, 1991)
                                STRUC SRHEAD
00000000 ??                     .REQLEN:    resb 1          ;LENGTH IN BYTES OF REQUEST BLOCK
00000001 ??                     .REQUNIT:   resb 1          ;DEVICE UNIT NUMBER
00000002 ??                     .REQFUNC:   resb 1          ;TYPE OF REQUEST
00000003 ????                   .REQSTAT:   resw 1          ;STATUS WORD
00000005 ????????????????               resb 8          ;RESERVED FOR QUEUE LINKS
                                .size:
                                endstruc

                                ; GENERIC IOCTL REQUEST STRUCTURE (DEVSYM.INC, MSDOS 6.0, 1991)
                                ;   SEE THE DOS 4.0 DEVICE DRIVER SPEC FOR FURTHER ELABORATION.
                                ;
                                struc IOCTL_REQ
00000000 <res Dh>                           resb SRHEAD.size
                                                        ;GENERIC IOCTL ADDITION.
0000000D ??                     .MAJORFUNCTION:     resb 1          ;FUNCTION CODE
0000000E ??                     .MINORFUNCTION:     resb 1          ;FUNCTION CATEGORY
0000000F ????                   .REG_SI:    resw 1
00000011 ????                   .REG_DI:    resw 1
00000013 ????????               .GENERICIOCTL_PACKET: resd 1      ; POINTER TO DATA BUFFER
                                endstruc

                                ; GENERIC IOCTL CATEGORY CODES  (IOCTL.INC, MSDOS 6.0, 1991)
                                IOC_OTHER   EQU     0       ; Other device control J.K. 4/29/86
                                IOC_SE              EQU     1       ; SERIAL DEVICE CONTROL
                                IOC_TC              EQU     2       ; TERMINAL CONTROL
                                IOC_SC              EQU     3       ; SCREEN CONTROL
                                IOC_KC              EQU     4       ; KEYBOARD CONTROL
                                IOC_PC              EQU     5       ; PRINTER CONTROL
                                IOC_DC              EQU     8       ; DISK CONTROL (SAME AS RAWIO)

                                ; DEFINITIONS FOR IOCTL_REQ.MINORFUNCTION
                                GEN_IOCTL_WRT_TRK   EQU   40H
                                GEN_IOCTL_RD_TRK    EQU   60H
                                GEN_IOCTL_FN_TST    EQU   20H     ; USED TO DIFF. BET READS AND WRTS

                                ;struc A_RETRYCOUNT  ; (IOCTL.INC, MSDOS 6.0, 1991)
                                ;.RC_COUNT: resw    1
                                ;endstruc
```

```
                                ; 29/05/2019 - Retro DOS v4.0 (DEVSYM.INC, MSDOS 6.0, 1991)

                                ;   THE DEVICE TABLE LIST HAS THE FORM:

                                ;struc SYSDEV
                                ; .NEXT:  resd 1   ;POINTER TO NEXT DEVICE HEADER
                                ; .ATT:       resw 1       ;ATTRIBUTES OF THE DEVICE
                                ; .STRAT: resw 1   ;STRATEGY ENTRY POINT
                                ; .INT:       resw 1       ;INTERRUPT ENTRY POINT
                                ; .NAME:  resb 8   ;NAME OF DEVICE (ONLY FIRST BYTE USED FOR BLOCK)
                                ; .size:
                                ;endstruc

                                ; 27/03/2018 - DEVSYM.INC - MSDOS 3.3 - 24/07/1987

                                ;
                                ; ATTRIBUTE BIT MASKS
                                ;
                                ; CHARACTER DEVICES:
                                ;
                                ; BIT 15 -> MUST BE 1
                                ;     14 -> 1 IF THE DEVICE UNDERSTANDS IOCTL CONTROL STRINGS
                                ;     13 -> 1 IF THE DEVICE SUPPORTS OUTPUT-UNTIL-BUSY
                                ;     12 -> UNUSED
                                ;     11 -> 1 IF THE DEVICE UNDERSTANDS OPEN/CLOSE
                                ;     10 -> MUST BE 0
                                ;      9 -> MUST BE 0
                                ;      8 -> UNUSED
                                ;      7 -> UNUSED
                                ;      6 -> UNUSED
                                ;      5 -> UNUSED
                                ;      4 -> 1 IF DEVICE IS RECIPIENT OF INT 29H
                                ;      3 -> 1 IF DEVICE IS CLOCK DEVICE
                                ;      2 -> 1 IF DEVICE IS NULL DEVICE
                                ;      1 -> 1 IF DEVICE IS CONSOLE OUTPUT
                                ;      0 -> 1 IF DEVICE IS CONSOLE INPUT
                                ;
                                ; BLOCK DEVICES:
                                ;
                                ; BIT 15 -> MUST BE 0
                                ;     14 -> 1 IF THE DEVICE UNDERSTANDS IOCTL CONTROL STRINGS
                                ;     13 -> 1 IF THE DEVICE DETERMINES MEDIA BY EXAMINING THE FAT ID BYTE.
                                ;       THIS REQUIRES THE FIRST SECTOR OF THE FAT TO *ALWAYS* RESIDE IN
                                ;       THE SAME PLACE.
                                ;     12 -> UNUSED
                                ;     11 -> 1 IF THE DEVICE UNDERSTANDS OPEN/CLOSE/REMOVABLE MEDIA
                                ;     10 -> MUST BE 0
                                ;      9 -> MUST BE 0
                                ;      8 -> UNUSED
                                ;      7 -> UNUSED
                                ;      6 -> IF DEVICE HAS SUPPORT FOR GETMAP/SETMAP OF LOGICAL DRIVES.
                                ;       IF THE DEVICE UNDERSTANDS GENERIC IOCTL FUNCTION CALLS.
                                ;      5 -> UNUSED
                                ;      4 -> UNUSED
                                ;      3 -> UNUSED
                                ;      2 -> UNUSED
                                ;      1 -> UNUSED
                                ;      0 -> UNUSED
                                ;

                                DEVTYP              EQU   8000H        ; BIT 15 - 1 IF CHAR, 0 IF BLOCK
                                CHARDEV         EQU   8000H
                                DEVIOCTL        EQU   4000H     ; BIT 14 - CONTROL MODE BIT
                                ISFATBYDEV      EQU   2000H     ; BIT 13 - DEVICE USES FAT ID BYTES,
                                                                ;  COMP MEDIA.
                                OUTTILBUSY      EQU   2000H     ; OUTPUT UNTIL BUSY IS ENABLED
                                ISNET               EQU   1000H        ; BIT 12 - 1 IF A NET DEVICE, 0 IF
                                                                ;  NOT.  CURRENTLY BLOCK ONLY.
                                DEVOPCL         EQU   0800H     ; BIT 11 - 1 IF THIS DEVICE HAS
                                                                ;  OPEN,CLOSE AND REMOVABLE MEDIA
                                                                ;  ENTRY POINTS, 0 IF NOT

                                EXTENTBIT       EQU   0400H     ; BIT 10 - CURRENTLY 0 ON ALL DEVS
                                                                ;  THIS BIT IS RESERVED FOR FUTURE USE
                                                                ;  TO EXTEND THE DEVICE HEADER BEYOND
                                                                ;  ITS CURRENT FORM.

                                ; NOTE BIT 9 IS CURRENTLY USED ON IBM SYSTEMS TO INDICATE "DRIVE IS SHARED".
                                ;    SEE IOCTL FUNCTION 9. THIS USE IS NOT DOCUMENTED, IT IS USED BY SOME
                                ;    OF THE UTILITIES WHICH ARE SUPPOSED TO FAIL ON SHARED DRIVES ON SERVER
                                ;    MACHINES (FORMAT,CHKDSK,RECOVER,..).

                                ; 18/03/2019 - Retro DOS v4.0
                                IOQUERY             EQU   0080H        ;Bit 7 - Supports generic IOCtl query M017

                                DEV320              EQU   0040H        ;BIT 6 - FOR BLOCK DEVICES, THIS
                                                                       ;DEVICE SUPPORTS SET/GET MAP OF
                                                                       ;LOGICAL DRIVES, AND SUPPORTS
                                                                       ;GENERIC IOCTL CALLS.
                                                                       ;FOR CHARACTER DEVICES, THIS
                                                                       ;DEVICE SUPPORTS GENERIC IOCTL.
                                                                       ;THIS IS A DOS 3.2 DEVICE DRIVER.
                                ISSPEC              EQU   0010H        ;BIT 4 - THIS DEVICE IS SPECIAL
                                ISCLOCK         EQU   0008H     ;BIT 3 - THIS DEVICE IS THE CLOCK DEVICE.
                                ISNULL              EQU   0004H        ;BIT 2 - THIS DEVICE IS THE NULL DEVICE.
                                ISCOUT              EQU   0002H        ;BIT 1 - THIS DEVICE IS THE CONSOLE OUTPUT.
                                ISCIN               EQU   0001H        ;BIT 0 - THIS DEVICE IS THE CONSOLE INPUT.
                                ; 23/07/2019 - Retro DOS v4.0
                                EXTDRVR             EQU   0002h ; (MSDOS 6.0, DEVSYM.INC, 1991)

                                ; 27/05/2018 - Retro DOS v3.0
                                ; [MSDOS 3.3, MSDISK.ASM]

                                struc INT13FRAME
00000000 ????                   .oldbp:     resw 1
00000002 ????                   .oldax:     resw 1
00000004 ????                   .oldbx:     resw 1
00000006 ????                   .oldcx:     resw 1
00000008 ????                   .olddx:     resw 1
0000000A ????????               .olddd:     resd 1
0000000E ????                   .oldf:      resw 1
                                .size:
                                endstruc

                                ; 02/06/2018 - Retro DOS v3.0
                                ; [MSDOS 3.3, BIOSTRUC.INC]

                                struc ROMBIOS_DESC          ; BIOS_SYSTEM_DESCRIPTOR
00000000 ????                   .bios_sd_leng:              resw 1
00000002 ??                     .bios_sd_modelbyte:         resb 1
                                .bios_sd_scnd_modelbyte:
00000003 ??                                         resb 1
00000004 ??                                         resb 1
```

```
00000005 ??                     .bios_sd_featurebyte1:    resb 1
00000006 ????????                                 resb 4
                                endstruc

                                ;--------------------------------------------------------------------------
                                ; MSDIOCTL.ASM - MSDOS 6.0 - 1991
                                ;--------------------------------------------------------------------------
                                ; 11/03/2019 - Retro DOS v4.0

                                ; 18/03/2019
                                DSK_TIMEOUT_ERR    EQU    80h     ; Time out error (no media present).
                                DSK_CHANGELINE_ERR EQU    06h     ; Change line error
                                DSK_ILLEGAL_COMBINATION EQU      0Ch     ; Return code of ah=18h function.
                                MULTI_TRK_ON               EQU    10000000b ; User specified multitrack=on,
                                                                     ; or system turns
                                ; IOCTL.INC - MSDOS 6.0 - 1991
                                ; ......................................................................

                                ;*** J.K.
                                ;General Guide -
                                ;Category Code:
                                ; 0... .... DOS Defined
                                ; 1... .... User defined
                                ; .xxx xxxx Code

                                ;Function Code:
                                ; 0... .... Return error if unsupported
                                ; 1... .... Ignore if unsupported
                                ; .0.. .... Intercepted by DOS
                                ; .1.. .... Passed to driver
                                ; ..0. .... Sends data/commands to device
                                ; ..1. .... Quries data/info from device
                                ; ...x .... Subfunction
                                ;
                                ; Note that "Sends/queries" data bit is intended only to regularize the
                                ; function set.  It plays no critical role; some functions may contain both
                                ; command and query elements. The convention is that such commands are
                                ; defined as "sends data".

                                ;*****************************;*
                                ; BLOCK DRIVERS          ;*
                                ;*****************************;*

                                ; IOCTL SUB-FUNCTIONS
                                IOCTL_GET_DEVICE_INFO      EQU    0
                                IOCTL_SET_DEVICE_INFO      EQU    1
                                IOCTL_READ_HANDLE  EQU    2
                                IOCTL_WRITE_HANDLE EQU    3
                                IOCTL_READ_DRIVE   EQU    4
                                IOCTL_WRITE_DRIVE  EQU    5
                                IOCTL_GET_INPUT_STATUS     EQU    6
                                IOCTL_GET_OUTPUT_STATUS EQU      7
                                IOCTL_CHANGEABLE?  EQU    8
                                IOCTL_DeviceLocOrRem?      EQU    9
                                IOCTL_HandleLocOrRem?      EQU    0Ah   ;10
                                IOCTL_SHARING_RETRY        EQU    0Bh   ;11
                                GENERIC_IOCTL_HANDLE       EQU    0Ch   ;12
                                GENERIC_IOCTL              EQU    0Dh   ;13
                                IOCTL_GET_DRIVE_MAP        EQU    0Eh   ;14
                                IOCTL_SET_DRIVE_MAP        EQU    0Fh   ;15
                                IOCTL_QUERY_HANDLE EQU    10h   ;16
                                IOCTL_QUERY_BLOCK  EQU    11h   ;17

                                ; GENERIC IOCTL SUB-FUNCTIONS
                                RAWIO                      EQU    8

                                ; RAWIO SUB-FUNCTIONS
                                GET_DEVICE_PARAMETERS      EQU    60H
                                SET_DEVICE_PARAMETERS      EQU    40H
                                READ_TRACK         EQU    61H
                                WRITE_TRACK        EQU    41H
                                VERIFY_TRACK               EQU    62H
                                FORMAT_TRACK               EQU    42H
                                GET_MEDIA_ID               EQU    66h    ;AN000;AN003;changed from 63h
                                SET_MEDIA_ID               EQU    46h    ;AN000;AN003;changed from 43h
                                GET_ACCESS_FLAG    EQU    67h    ;AN002;AN003;Unpublished function.Changed from 64h
                                SET_ACCESS_FLAG    EQU    47h    ;AN002;AN003;Unpublished function.Changed from 44h
                                SENSE_MEDIA_TYPE   EQU    68H    ;Added for 5.00


                                ; SPECIAL FUNCTION FOR GET DEVICE PARAMETERS
                                BUILD_DEVICE_BPB   EQU    000000001B

                                ; SPECIAL FUNCTIONS FOR SET DEVICE PARAMETERS
                                INSTALL_FAKE_BPB   EQU    000000001B
                                ONLY_SET_TRACKLAYOUT       EQU    000000010B
                                TRACKLAYOUT_IS_GOOD        EQU    000000100B

                                ; SPECIAL FUNCTION FOR FORMAT TRACK
                                STATUS_FOR_FORMAT  EQU    000000001B
                                DO_FAST_FORMAT             EQU    000000010B ;AN001;
                                ; CODES RETURNED FROM FORMAT STATUS CALL
                                FORMAT_NO_ROM_SUPPORT      EQU    000000001B
                                FORMAT_COMB_NOT_SUPPORTED EQU    000000010B

                                ; DEVICETYPE VALUES
                                MAX_SECTORS_IN_TRACK       EQU    63     ; MAXIMUM SECTORS ON A DISK.(Was 40 in DOS 3.2)
                                DEV_5INCH          EQU    0
                                DEV_5INCH96TPI             EQU    1
                                DEV_3INCH720KB             EQU    2
                                DEV_8INCHSS        EQU    3
                                DEV_8INCHDS        EQU    4
                                DEV_HARDDISK               EQU    5
                                DEV_OTHER          EQU    7
                                ;DEV_3INCH1440KB   EQU    7
                                DEV_3INCH2880KB            EQU    9
                                ; Retro DOS v2.0 - 26/03/2018
                                ;;DEV_TAPE         EQU    6
                                ;;DEV_ERIMO        EQU    8
                                ;DEV_3INCH2880KB   EQU    9
                                DEV_3INCH1440KB            EQU    10

                                ;MAX_DEV_TYPE              EQU    9      ; MAXIMUM DEVICE TYPE THAT WE
                                                                     ; CURRENTLY SUPPORT.
                                MAX_DEV_TYPE               EQU    10

                                struc A_SECTORTABLE
00000000 ????                   .ST_SECTORNUMBER:  resw   1
00000002 ????                   .ST_SECTORSIZE:            resw   1
                                .size:
                                endstruc

                                ; 25/03/2019 - Retro DOS v4.0  (MSDOS 6.0, BPB.INC, IOCTL.INC)
```

```
                        ; MSDOS 6.0 - BPB.INC - 1991
                        ; ####
                        ;** BIOS PARAMETER BLOCK DEFINITION
                        ;
                        ;   The BPB contains information about the disk structure.  It dates
                        ;   back to the earliest FAT systems and so FAT information is
                        ;   intermingled with physical driver information.
                        ;
                        ;   A boot sector contains a BPB for its device; for other disks
                        ;   the driver creates a BPB.  DOS keeps copies of some of this
                        ;   information in the DPB.
                        ;
                        ;   The BDS structure contains a BPB within it.

                        ; 01/01/2024
                        %if 0

                        struc A_BPB
                        .BPB_BYTESPERSECTOR:      resw   1
                        .BPB_SECTORSPERCLUSTER:   resb   1
                        .BPB_RESERVEDSECTORS:     resw   1
                        .BPB_NUMBEROFFATS: resb   1
                        .BPB_ROOTENTRIES:  resw   1
                        .BPB_TOTALSECTORS: resw   1
                        .BPB_MEDIADESCRIPTOR:     resb   1
                        .BPB_SECTORSPERFAT:       resw   1
                        .BPB_SECTORSPERTRACK:     resw   1
                        .BPB_HEADS:         resw   1
                        .BPB_HIDDENSECTORS:       resw   1
                                            resw   1
                        .BPB_BIGTOTALSECTORS:     resw   1
                                            resw   1
                                            resb   6      ; NOTE:  many times these
                        ;                                 ;       6 bytes are omitted
                        ;                                 ;       when BPB manipulations
                        ;                                 ;       are performed!
                        .size:
                        endstruc

                                            %else

                                            ; 14/04/2024
                                            ; 01/01/2024 - Retro DOS v5.0

                                            struc A_BPB
00000000 ????               .BYTESPERSECTOR:    resw 1
00000002 ??                 .SECTORSPERCLUSTER: resb 1
00000003 ????               .RESERVEDSECTORS:   resw 1
00000005 ??                 .NUMBEROFFATS:         resb 1
00000006 ????               .ROOTENTRIES:          resw 1
00000008 ????               .TOTALSECTORS:         resw 1
0000000A ??                 .MEDIADESCRIPTOR:   resb 1
0000000B ????               .SECTORSPERFAT:        resw 1
0000000D ????               .SECTORSPERTRACK:   resw 1
0000000F ????               .HEADS:                resw 1
00000011 ????????           .HIDDENSECTORS:        resd 1
00000015 ????????           .BIGTOTALSECTORS:   resd 1
                            ;............ FAT32 ......  + 28
00000019 ????????           .FATSIZE32:     resd 1
0000001D ????               .EXTFLAGS:      resw 1
0000001F ????                               .FSVER:                resw 1
00000021 ????????           .ROOTDIRCLUSTER:    resd 1
00000025 ????               .FSINFOSECTOR:         resw 1  ; (offset from FAT32 bs)
00000027 ????               .BACKUPBOOTSECTOR:  resw 1  ; (offset from FAT32 bs)
00000029 <res Ch>           .RESERVEDBYTES:        resb 12 ; (zero bytes)
                                                ; 14/04/2024
00000035 ???????????????                           resb 6  ; A_BPB.size must be 59
                                            .size:
                                            endstruc

                                            %endif

                                            struc A_DEVICEPARAMETERS
00000000 ??                 .DP_SPECIALFUNCTIONS:     resb   1
00000001 ??                 .DP_DEVICETYPE:           resb   1
00000002 ????               .DP_DEVICEATTRIBUTES:     resw   1
00000004 ????               .DP_CYLINDERS:            resw   1
00000006 ??                 .DP_MEDIATYPE:            resb   1
00000007 <res 3Bh>          .DP_BPB:            resb   A_BPB.size
00000042 ????               .DP_TRACKTABLEENTRIES:    resw   1
00000044 <res FCh>          .DP_SECTORTABLE:   resb   MAX_SECTORS_IN_TRACK * A_SECTORTABLE.size
                            endstruc

                                            struc A_TRACKREADWRITEPACKET
00000000 ??                 .TRWP_SPECIALFUNCTIONS:   resb   1
00000001 ????               .TRWP_HEAD:         resw   1
00000003 ????               .TRWP_CYLINDER:           resw   1
00000005 ????               .TRWP_FIRSTSECTOR: resw   1
00000007 ????               .TRWP_SECTORSTOREADWRITE: resw   1
00000009 ????????           .TRWP_TRANSFERADDRESS:    resd   1
                            endstruc

                            ;AN001; - FP_TRACKCOUNT is only meaningful when FP_SPECIALFUNCTIONS bit 1 = 1.
                                            struc A_FORMATPACKET
00000000 ??                 .FP_SPECIALFUNCTIONS:     resb   1  ; db ?
00000001 ????               .FP_HEAD:           resw   1  ; dw ?
00000003 ????               .FP_CYLINDER:             resw   1  ; dw ?
00000005 ????               .FP_TRACKCOUNT:           resw   1  ; dw 1 ; !
                            endstruc

                                            struc A_VERIFYPACKET
00000000 ??                 .VP_SPECIALFUNCTIONS:     resb   1
00000001 ????               .VP_HEAD:           resw   1
00000003 ????               .VP_CYLINDER:             resw   1
                            endstruc

                                            struc A_MEDIA_ID_INFO
00000000 ????               .MI_LEVEL:          resw   1  ; dw 0 ; !         ;J.K. 87 Info. level
00000002 ????????           .MI_SERIAL:         resd   1  ; dd ?             ;J.K. 87 Serial #
00000006 <res Bh>           .MI_LABEL:          resb   11 ; db 11 DUP (' ') ;!      ;J.K. 87 volume label
00000011 ????????????????   .MI_SYSTEM:         resb   8  ; db 8 DUP (' ')  ;!      ;J.K. 87 File system type
                            endstruc

                                            struc A_DISKACCESS_CONTROL    ;AN002; Unpublished function. Only for Hard file.
00000000 ??                 .DAC_SPECIALFUNCTIONS:    resb   1  ; db 0 ; ! ;AN002; Always 0
00000001 ??                 .DAC_ACCESS_FLAG:  resb   1  ; db 0 ; !
                                                          ; Non Zero - allow disk I/O to unformatted hard file
                            endstruc                      ; 0 - Disallow disk I/O to unformatted hard file


                                            struc A_MEDIA_SENSE                     ; Media sense structure added 5.00
00000000 ??                 .MS_ISDEFAULT:            resb   1      ; If 1 type returned is drv default
00000001 ??                 .MS_DEVICETYPE:           resb   1      ; Drive type
00000002 ??                 .MS_RESERVED1:            resb   1      ; RESERVED
```

```
00000003 ??                     .MS_RESERVED2:            resb    1       ; RESERVED
                                endstruc

                                ;*********************************;*
                                ; CHARACTER DEVICES (PRINTERS)    ;*
                                ;*********************************;*

                                ;RAWIO SUB-FUNCTIONS
                                GET_RETRY_COUNT    EQU     65H
                                SET_RETRY_COUNT    EQU     45H

                                struc A_RETRYCOUNT
00000000 ????                   .RC_COUNT:         resw 1
                                endstruc

                                ;*********************************;*             ;J.K. 4/29/86
                                ; CHARACTER DEVICES (SCREEN)      ;*
                                ;*********************************;*             ;J.K. 4/29/86
                                ;
                                ;SC_MODE_INFO       struc
                                ;SC_INFO_LENGTH     DW      9
                                ;SC_MODE            DB      0
                                ;SC_COLORS          DW      0
                                ;SC_WIDTH           DW      0
                                ;SC_LENGTH          DW      0
                                ;SC_MODE_INFO       ends
                                ;
                                ;SC_INFO_PACKET_LENGTH     EQU     9             ;LENGTH OF THE INFO PACKET.

                                ;SUBFUNCTIONS FOR CON$GENIOCTL
                                ;GET_SC_MODE               EQU     60h
                                ;SET_SC_MODE               EQU     40h
                                ;The following subfunctions are reserved for installable CODE PAGE switch
                                ;console devices. - J.K. 4/29/86
                                ;Get_active_codepage       equ     6Ah
                                ;Invoke_active_codepage  equ      4Ah
                                ;Start_designate_codepage equ     4Ch
                                ;End_designate_codepage  equ      4Dh
                                ;Get_list_of_designated_codepage equ 6Bh
                                ;J.K. 4/29/86 *** End of Con$genioctl equates & structures

                                ;-----------------------------------------------------------------------------
                                ; MULT.INC - MSDOS 6.0 - 1991
                                ;-----------------------------------------------------------------------------
                                ; 18/03/2019

                                ; The current set of defined multiplex channels is (* means documented):
                                ;
                                ;   Channel(h)  Issuer          Receiver    Function
                                ;      00       server          PSPRINT     print job control
                                ;     *01       print/apps      PRINT       Queueing of files
                                ;      02       BIOS            REDIR       signal open/close of printers
                                ;
                                ;      05       command         REDIR       obtain text of net int 24 message
                                ;     *06       server/assign   ASSIGN      Install check
                                ;
                                ;      08       external driver IBMBIO      interface to internal routines
                                ;
                                ;      10       sharer/server   Sharer      install check
                                ;      11       DOS/server      Redir       install check/redirection funcs
                                ;      12       sharer/redir    DOS         dos functions and structure maint
                                ;      13       MSNET           MSNET       movement of NCBs
                                ;      13       external driver IBMBIO      Reset_Int_13, allows installation
                                ;                                           of alternative INT_13 drivers after
                                ;                                           boot_up
                                ;      14 (IBM) DOS             NLSFUNC     down load NLS country info,DOS 3.3
                                ;      14 (MS)  APPS            POPUP       MSDOS 4 popup screen functions
                                ;      15       APPS            MSCDEX      CD-ROM extensions interface
                                ;      16       WIN386          WIN386      Windows communications
                                ;      17       Clipboard       WINDOWS     Clipboard interface
                                ;     *18       Applications    MS-Manger   Toggle interface to manager
                                ;      19       Shell
                                ;      1A       Ansi.sys
                                ;      1B       Fastopen,Vdisk   IBMBIO     EMS INT 67H stub handler
                                ;
                                ;      40h      OS/2
                                ;      41h      Lanman
                                ;      42h      Lanman
                                ;      43h      Himem
                                ;                               AL = 20h    reserved for Mach 20 Himem support
                                ;                               AL = 30h    reserved for Himem external A20 code
                                ;      44h      Dosextender
                                ;      45H      Windows profiler
                                ;      46h      Windows/286 DOS extender
                                ;      47h      Basic Compiler Vn. 7.0
                                ;      48h      Doskey
                                ;      49h      DOS 5.x install
                                ;      4Ah      Multi Purpose
                                ;                multMULTSWPDSK         0 - Swap Disk in drive A (BIOS)
                                ;                multMULTGETHMAPTR      1 - Get available HMA & ptr
                                ;                multMULTALLOCHMA       2 - Allocate HMA (bx == no of bytes)
                                ;                multMULTTASKSHELL      5 - Shell/switcher API
                                ;                multMULTRPLTOM         6 - Top Of Memory for RPL support
                                ;
                                ;                multSmartdrv           10h
                                ;                multMagicdrv           11h
                                ;      4Bh      Task Switcher API
                                ;
                                ;      4Ch      APPS            APM             Advanced power management
                                ;      4Dh      Kana Kanji Converter, MSKK
                                ;
                                ;      51h      ODI real mode support driver (for Chicago)
                                ;
                                ;      53h      POWER.EXE - used for broadcasting APM events    ; M036
                                ;      54h      POWER.EXE - used for POWER API                  ; M036
                                ;
                                ;      55h      COMMAND.COM
                                ;                multCOMFIRST           0 - API to determine whether 1st
                                ;                                           instance of command.com
                                ;                multCOMFIRSTROM        1 - API to determine whether 1st
                                ;                                           instance of ROM COMMAND
                                ;      56h      Sewell Development
                                ;               INTERLNK
                                ;
                                ;      57h      Iomega Corp.
                                ;
                                ;      AB       Unspecified IBM use
                                ;      AC       Graphics
                                ;      AD       NLS (toronto)
                                ;      AE
                                ;      AF       Mode
                                ;      B0       GRAFTABL        GRAFTABL
                                ;
                                ;      D7       Banyan VINES
```

```
                                multMULT     equ  4Ah

                                multMULTSWPDSK       equ  0      ; Swap Disk in drive A (BIOS)
                                multMULTGETHMAPTR equ     1      ; Get available HMA & ptr
                                multMULTALLOCHMA  equ     2      ; Allocate HMA (bx == no of bytes)
                                multMULTTASKSHELL equ     5      ; Shell/switcher API
                                multMULTRPLTOM       equ  6      ; Top Of Memory for RPL support

                                ;--------------------------------------------------------------------------------
                                ; WIN386.INC - MSDOS 6.0 - 1991
                                ;--------------------------------------------------------------------------------
                                ; 18/03/2019

                                ; WIN386.INC
                                ;
                                ;  Symbols and structures relating to WIN386 support.
                                ;
                                ;  Used by files in both the DOS and the BIOS.
                                ;
                                ;  Created: 7-13-89 by MRW
                                ;

                                ; WIN386 broadcast int 2fh multiplex number and subfunction numbers

                                MultWin386         equ     16h    ; Int 2f multiplex number

                                Win386_Init        equ     05h    ; Win386 initialization
                                Win386_Exit        equ     06h    ; Win386 exit
                                Win386_Devcall             equ     07h     ; Win386 device call out
                                Win386_InitDone            equ     08h     ; Win386 initialization is complete

                                ; ============================================================================

                                ;--------------------------------------------------------------------------------
                                ;
                                ; +------------------------------------------------------------------------+
                                ; |   This file    has been generated by The Interactive Disassembler (IDA)    |
                                ; |        Copyright       (c) 2013 Hex-Rays, <support@hex-rays.com>        |
                                ; |                  Licensed to: Freeware version                         |
                                ; +------------------------------------------------------------------------+
                                ;
                                ;--------------------------------------------------------------------------------

                                ;           .386
                                ;           .model flat

                                ; ============================================================================

                                ; 02/10/2023 - Retro DOS v5.0 (Modified PCDOS 7.1)
                                ; 10/12/2022
                                ; 09/12/2022
                                ; 21/10/2022
                                ; 19/10/2022
                                ; 17/10/2022, 18/10/2022
                                ; 15/10/2022, 16/10/2022
                                ; 03/10/2022
                                ; 02/10/2022
                                ; 01/10/2022 - Erdogan Tan

                                ; [[ Most of comments here are from the original MSDOS 6.0 source code ]]

                                ;--------------------------------------------------------------------------------
                                ; Start of PC-DOS 7.1 IBMBIO.COM  (IO.SYS)
                                ;--------------------------------------------------------------------------------

                                            ; [ORG 0]              ; segment 0x0070h

                                ;============================================================================
                                ; DOS BIOS (IO.SYS) DATA SEGMENT
                                ;============================================================================
                                ; 09/12/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 IO.SYS)
                                ; 30/12/2022 - Retro DOS v4.2 (Modified MSDOS 6.21 IO.SYS)
                                ; 02/10/2023 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMBIO.COM)

                                section .BIOSDATA vstart=0

                                ;--- DOSBIOS data segment -------------------------------------------------------
                                ;--------------------------------------------------------------------------------

                                ;Bios_Data segment

                                BData_start:
00000000 E9931B                 hdrv_pat:   jmp     init           ; MSBIO1.ASM, MSSBDATA.INC
                                ; --------------------------------------------------------------------------------

00000003 0000                   DosDataSg:  dw 0

                                ; DOS's int 2f handler will exit via a jump through here.
                                ; This is how the BIOS hooks int2f

                                ;BIOSDATA:0005h: ; 10/05/2023 (Note the 'bios_i2f equ 5' in 'msdos6.s')

00000005 EA                     bios_i2f:   db 0EAh               ; far jump to int_2f (segment may not be at 70h)
                                            ; 07/08/2023 - Retro DOS v4.2 IO.SYS (optimization)
                                            ; PCDOS 7.1 IBMBIO.COM - BIODATA:0006h
                                            ;dw int_2f
                                            ;dw 70h               ; BIOSDATA segment (KERNEL_SEGMENT)
00000006 [2B16]                             dw i2f_handler
                                bios_i2f_seg:      ; 10/08/2023
00000008 0203                               dw DOSBIOCODESEG      ; 02CCh for MSDOS 6.21 IO.SYS (25Ch+070h)
                                                                  ; 0364h PCDOS 7.1 IBMBIO.COM  (2F4h+070h)

0000000A 0000                   romstartaddr:      dw 0                   ; The start address for the romfind routines
                                                                  ; This is to maintain binary compatibility
                                                                  ; with DISK based DOS 5.0

                                ; This is a byte used for special key handling in the resident
                                ; console device driver. It must be here so that it can be included
                                ; in the WIN386 instance table (in INC\LMSTUB.ASM).

0000000C 00                     altah:             db 0                   ; special key handling

0000000D 00                     inHMA:             db 0                   ; flag indicates we're running from HMA
0000000E 00000000               xms:       dd 0                   ; entry point to xms if above is true

                                ; PTRSAV - pointer save
                                ;
                                ; This variable holds the pointer to the Request Header passed by a program
                                ; wishing to use a device driver. When the strategy routine is called it
                                ; puts the address of the Request header in this variable and returns.

00000012 00000000               ptrsav:            dd 0
                                auxbuf:            ;db 4 dup(0)           ; set of 1 byte buffers for com 1,2,3, and 4
```

```
1117 00000016 00000000                          db 0, 0, 0, 0 ; 19/10/2022
1118 0000001A 0000                    zeroseg:  dw 0                    ; easy way to load segment registers with zero
1119 0000001C 0000                    i13_ds:         dw 0                      ; ds register for int13 call through
1120 0000001E 0000                    prevoper:  dw 0                   ; holds int 13 request (i.e. register ax).
1121 00000020 00                      number_of_sec:    db 0                    ; holds number of secs. to read on an ecc error
1122 00000021 0000                    auxnum:           dw 0                    ; which aux device was requested
1123
1124                                  ;--------------------------------------------------------------------------
1125
1126                                  res_dev_list:
1127
1128                                  ; Device Header for the CON Device Driver
1129
1130                                  CONHeader:                        ; HEADER FOR DEVICE "CON"
1131 00000023 [3500]                              dw auxdev2
1132 00000025 7000                              dw 70h
1133 00000027 1380                    word_727:  dw 8013h
1134 00000029 [1506]                            dw strategy
1135 0000002B [2006]                            dw con_entry
1136 0000002D 434F4E2020202020        aCon:             db 'CON     '
1137 00000035 [4700]                  auxdev2:  dw prndev2              ; HEADER FOR DEVICE "AUX"
1138 00000037 7000                              dw 70h
1139 00000039 0080                              dw 8000h
1140 0000003B [1506]                            dw strategy
1141 0000003D [4106]                            dw aux0_entry
1142 0000003F 4155582020202020        aAux:             db 'AUX     '
1143 00000047 [5900]                  prndev2:  dw timdev               ; HEADER FOR DEVICE "PRN"
1144 00000049 7000                              dw 70h
1145 0000004B C0A0                    word_74B:  dw 0A0C0h
1146 0000004D [1506]                            dw strategy
1147 0000004F [2506]                            dw prn0_entry
1148 00000051 50524E2020202020        aPrn:             db 'PRN     '         ; HEADER FOR DEVICE "CLOCK$"
1149 00000059 [6B00]                  timdev:           dw dskdev
1150 0000005B 7000                              dw 70h
1151 0000005D 0880                              dw 8008h
1152 0000005F [1506]                            dw strategy
1153 00000061 [5906]                            dw tim_entry
1154 00000063 434C4F434B242020        aClock:           db 'CLOCK$  '
1155 0000006B [7B00]                  dskdev:           dw com1dev              ; HEADER FOR DISK DEVICES
1156 0000006D 7000                              dw 70h
1157                                            ;dw 8C2h
1158                                            ; 02/10/2023 - Retro DOS v5.0
1159 0000006F C248                              dw 48C2h                ; PCDOS 7.1 IBMBIO.COM - BIOSDATA:006Fh
1160                                            ;dw offset strategy
1161                                            ;dw offset dsk_entry
1162                                            ; 19/10/2022
1163 00000071 [1506]                            dw strategy
1164 00000073 [5E06]                            dw dsk_entry
1165
1166                                  ; maximum number of drives
1167
1168 00000075 04                      drvmax:           db 4
1169 00000076 FE                      step_drv:  db 0FEh ; -2           ; last drive accessed
1170 00000077 00                      fhave96:   db 0                   ; flag to indicate presence of
1171                                                                     ; 96tpi support
1172 00000078 00                      single:           db 0                    ; used to detect single drive systems
1173 00000079 00                      fhavek09:  db 0                   ; indicates if this is a k09 or not
1174                                                                     ; used by console driver.
1175 0000007A 00                      fsetowner:  db 0                  ; = 1 if we are setting the owner of a
1176                                                                     ; drive. (examined by checksingle).
1177
1178 0000007B [8D00]                  com1dev:  dw lpt1dev              ; Device Header for device "COM1"
1179 0000007D 7000                              dw 70h
1180 0000007F 0080                              dw 8000h
1181 00000081 [1506]                            dw strategy
1182 00000083 [4106]                            dw aux0_entry
1183 00000085 434F4D3120202020        aCom1:            db 'COM1    '
1184 0000008D [9F00]                  lpt1dev:  dw lpt2dev              ; Device Header for device LPT1
1185 0000008F 7000                              dw 70h
1186 00000091 C0A0                              dw 0A0C0h
1187 00000093 [1506]                            dw strategy
1188 00000095 [2C06]                            dw prn1_entry
1189 00000097 4C5054312020202020      aLpt1:            db 'LPT1    '
1190 0000009F [B800]                  lpt2dev:  dw lpt3dev              ; Device Header for device LPT2
1191 000000A1 7000                              dw 70h
1192 000000A3 C0A0                              dw 0A0C0h
1193 000000A5 [1506]                            dw strategy
1194 000000A7 [3306]                            dw prn2_entry
1195 000000A9 4C5054322020202000-     aLpt2:            db 'LPT2    ',0,0,0
1195 000000B2 0000
1196
1197                                  ;M058; Start of changes
1198                                  ; Orig13 needs to be at offset 0B4h for the CMS floppy driver to work.
1199                                  ;These guys patch Orig13 with their own int 13h hook and so this offset
1200                                  ;cannot change for them to work. Even ProComm does this.
1201
1202 000000B4 00000000                Orig13:           dd 0                    ; to make Orig13 offset 0B4h
1203
1204 000000B8 [CA00]                  lpt3dev:  dw com2dev              ; Device Header for device LPT3
1205 000000BA 7000                              dw 70h
1206 000000BC C0A0                              dw 0A0C0h
1207 000000BE [1506]                            dw strategy
1208 000000C0 [3A06]                            dw prn3_entry
1209 000000C2 4C5054332020202020      aLpt3:            db 'LPT3    '
1210 000000CA [DC00]                  com2dev:  dw com3dev              ; Device Header for device "COM2"
1211 000000CC 7000                              dw 70h
1212 000000CE 0080                              dw 8000h
1213 000000D0 [1506]                            dw strategy
1214 000000D2 [4706]                            dw aux1_entry
1215                                            ; 19/10/2022
1216 000000D4 434F4D3220202020        aCom2:            db 'COM2    '
1217                                  com3dev:  ;dw offset com4dev      ; Device Header for device "COM3"
1218 000000DC [EE00]                            dw com4dev
1219 000000DE 7000                              dw 70h
1220 000000E0 0080                              dw 8000h
1221                                            ;dw offset strategy
1222                                            ;dw offset aux2_entry
1223 000000E2 [1506]                            dw strategy
1224 000000E4 [4D06]                            dw aux2_entry
1225 000000E6 434F4D3320202020        aCom3:            db 'COM3    '
1226 000000EE FFFF                    com4dev:  dw 0FFFFh               ; Device Header for device "COM4"
1227 000000F0 7000                              dw 70h
1228 000000F2 0080                              dw 8000h
1229 000000F4 [1506]                            dw strategy
1230 000000F6 [5306]                            dw aux3_entry
1231 000000F8 434F4D3420202020                  db 'COM4    '
1232
1233                                  ;--------------------------------------------------------------------------
1234
1235 00000100 10                      RomVectors: db 10h
1236 00000101 00000000                Old10:            dd 0
1237 00000105 13                                db 13h
1238 00000106 00000000                Old13:            dd 0
1239 0000010A 15                                db 15h
```

```
1240 0000010B 00000000                Old15:              dd 0
1241 0000010F 19                                  db 19h
1242 00000110 00000000                Old19:              dd 0
1243 00000114 1B                                  db 1Bh
1244 00000115 00000000                Old1B:              dd 0
1245
1246                                  ;EndRomVectors      equ $
1247
1248                                  ;NUMROMVECTORS      equ ((EndRomVectors - RomVectors)/5)
1249
1250                                  ;--------------------------------------------------------------------------------
1251
1252 00000119 [5203]                  start_bds:  dw bds1               ; Start of linked list of BDS's
1253 0000011B 7000                                dw 70h                ; KERNEL_SEGMENT
1254
1255                                  ; (MSDOS 3.3) NOTE:
1256                                  ; Some floppy drives do not have changeline support. The result is a
1257                                  ; large amount of inefficiency in the code. A media-check always returns
1258                                  ; "I don`t know". This cause DOS to reread the FAT on every access and
1259                                  ; always discard any cached data.
1260                                  ;    We get around this inefficiency by implementing a "Logical Door Latch".
1261                                  ; The following three items are used to do this. The logical door latch is
1262                                  ; based on the premise that it is not physically possible to change floppy
1263                                  ; disks in a drive in under two seconds (most people take about 10). The
1264                                  ; logical door latch is implemented by saving the time of the last successful
1265                                  ; disk operation (in the value TIM_DRV). When a new request is made the
1266                                  ; current time is compared to the saved time. If less than two seconds have
1267                                  ; passed then the value "No Change" is returned. If more than two seconds
1268                                  ; have passed the value "Don't Know" is returned.
1269                                  ;    There is one complecation to this algorithm. Some programs change the
1270                                  ; value of the timer. In this unfortunate case we have an invalid timer.
1271                                  ; This possibility is detected by counting the number of disk operations
1272                                  ; which occur without any time passing. If this count exceeds the value of
1273                                  ; "AccessMax" we assume the counter is invalid and always return "Don't
1274                                  ; Know". The variable "AccessCount" is used to keep track of the number
1275                                  ; of disk operation which occur without the time changing.
1276
1277 0000011D 00                      accesscount:        db 0
1278 0000011E FF                      tim_drv:    db 0FFh
1279 0000011F 00                      medbyt:             db 0
1280                                  wrtverify:  ; 15/10/2022
1281 00000120 02                      rflag:              db 2                  ; 2 for read, 3 for write
1282 00000121 00                      verify:             db 0                  ; 1 if verify after write
1283 00000122 0000                    seccnt:             dw 0
1284 00000124 00                                  db 0                  ; -- pad where hardnum was
1285 00000125 01                      dsktnum:    db 1                  ; number of diskette drives
1286
1287                                  ; (MSDOS 3.3) NOTE:
1288                                  ; Some of the older versions of the IBM rom-bios always assumed a seek would
1289                                  ; have to be made to read the diskette. Consequently a large head settle
1290                                  ; time was always used in the I/O operations. To get around this problem
1291                                  ; we need to continually adjust the head settle time. The following
1292                                  ; algorithm is used:
1293                                  ;
1294                                  ;   Get the current head settle value.
1295                                  ;   If it is 1, then
1296                                  ;   set slow = 15
1297                                  ;   else
1298                                  ;   set slow = value
1299                                  ;   ...
1300                                  ;   if we are seeking and writing then
1301                                  ;   use slow
1302                                  ;   else
1303                                  ;   use fast
1304                                  ;   ...
1305                                  ;   restore current head settle value
1306
1307 00000126 00                      motorstartup:       db 0                  ; value from table
1308 00000127 00                      settlecurrent:      db 0                  ; value from table
1309 00000128 00                      settleslow: db 0                  ; slow settle value
1310 00000129 00                      nextspeed:  db 0                  ; value of speed to be used
1311 0000012A 00                      save_head_sttl:     db 0                  ; used by read_sector routine
1312 0000012B 00                      save_eot:   db 0                  ; saved eot from the default DPT
1313 0000012C 09                      eot:        db 9
1314 0000012D 00000000                dpt:        dd 0                  ; pointer to Disk Parameter Table
1315 00000131 00                      cursec:             db 0                  ; current sector
1316 00000132 00                      curhd:              db 0                  ; current head
1317 00000133 0000                    curtrk:             dw 0                  ; current track
1318 00000135 0000                    spsav:              dw 0                  ; save the stack pointer
1319 00000137 08                      formt_eot:  db 8                  ; eot used for format
1320 00000138 00                      hdnum:              db 0                  ; head number
1321 00000139 0000                    trknum:             dw 0                  ; track being manipulated
1322 0000013B 50                      gap_patch:  db 50h                ; format gap patched into dpt
1323
1324                                  ;--------------------------------------------------------------------------------
1325
1326                                  ; disk errors returned from the IBM rom
1327
1328 0000013C CC                      errin:              db 0CCh               ; write fault (hard disk)
1329 0000013D 80                                  db 80h                ; write fault (hard disk)
1330 0000013E 40                                  db 40h                ; seek failed
1331 0000013F 10                                  db 10h                ; uncorrectable CRC or ECC error on read
1332 00000140 08                                  db 8                  ; dma overrun
1333 00000141 06                                  db 6                  ; disk changed (floppy)
1334 00000142 04                                  db 4                  ; sector not found/read error
1335 00000143 03                                  db 3                  ; disk write-protected
1336                                              ; 02/10/2023 - Retro DOS 5.0 (PCDOS 7.1 IBMBIO.COM)
1337 00000144 01                                  db 1                  ; invalid function in AH or invalid parameter
1338 00000145 B2                                  db 0B2h               ; volume not removable
1339                                              ;
1340 00000146 00                      lsterr:             db 0                  ; all other errors
1341
1342                                  ; returned error codes corresponding to above
1343
1344 00000147 0A                      errout:             db 10                 ; write fault error
1345 00000148 02                                  db 2                  ; no response (timeout)
1346 00000149 06                                  db 6                  ; seek failure
1347 0000014A 04                                  db 4                  ; bad crc
1348 0000014B 04                                  db 4                  ; dma overrun
1349 0000014C 0F                                  db 15                 ; invalid media       change
1350 0000014D 08                                  db 8                  ; sector not found
1351 0000014E 00                                  db 0                  ; write attempt to write-protect disk
1352                                              ; 02/10/2023
1353 0000014F 03                                  db 3                  ; unknown command error
1354 00000150 03                                  db 3                  ; unknown command error
1355                                              ;
1356 00000151 0C                                  db 12                 ; general error
1357
1358                                  ;--------------------------------------------------------------------------------
1359                                  ; PCDOS 7.1 IBMBIO.COM - BIOSDATA:0152h
1360
1361                                  ; 02/10/2023 - Retro DOS v5.0 (Modified PCDOS 7.1)
1362                                  %if 1
1363                                  disksector: ;times 174 db 0
```

```
                                NUM174 equ 512-$
00000152 00<rep AEh>                    times NUM174 db 0
                                JB_sign     :       ;dw 424Ah             ; 'BJ' (nasm) ; 'JB' (masm)
00000200 4A                             dec     dx
00000201 42                             inc     dx
00000202 E9FBFD                         jmp     BData_start    ; db 0E9h, 0FBh, 0FDh

00000205 402349424D3A31322E-    IBMBIOCOM$: db '@#IBM:12.01.2003.build_1.32#@ IBMBIO.COM(USA)',0
0000020E 30312E323030332E62-
00000217 75696C645F312E3332-
00000220 23402049424D42494F-
00000229 2E434F4D2855534129-
00000232 00

                                        ;times 287 db 0
00000233 00<rep 11Fh>                   times (disksector+512-$) db 0  ; 287
                                %endif

                                ;--------------------------------------------------------------------------

                                ; 30/12/2018 - Retro DOS v4.0

                                ; read in boot sector here, read done in readboot.
                                ; also read sector for dma check for hard disk.
                                ;
                                ; This buffer is word aligned because certain AMI BIOSs have a bug
                                ; in them which causes the byte after the buffer to be trashed
                                ; on floppy reads to odd-byte boundaries. Although no general effort
                                ; is made to enforce this in the bigger picture, this one small sacrifice
                                ; makes that system more-or-less work.

                                ; 02/10/2023
                                %if 0

                                disksector: ;db 512 dup(0)        ; read in boot sector here
                                            ; 19/10/2022
                                            times 512 db 0
                                %endif

                                ;--------------------------------------------------------------------------

                                ; 02/10/2023 - Retro DOS v5.0
                                ; 30/12/2018 - Retro DOS v4.0
                                ;--------------------------------------------------------------------------
                                ; 25/05/2018 (04/04/2018)
                                ;**************************************************************************
                                ;   "bds" contains information for each drive in the system.
                                ;   various values are patched whenever actions are performed.
                                ;   sectors/alloc. unit in bpb initially set to -1 to signify that
                                ;   the bpb has not been filled. link also set to -1 to signify end
                                ;   of list. # of cylinders in maxparms initialized to -1 to indicate
                                ;   that the parameters have not been set.

                                bds1:               ;dw offset bds2
00000352 [E803]                             dw bds2 ; 19/10/2022
00000354 7000                               dw 70h                  ; dword link to next structure
00000356 00                                 db 0                    ; int 13h drive       number
00000357 00                                 db 0                    ; logical drive       letter
00000358 0002                   fdrive1:    dw 512
                                                                    ; physical sector size in bytes
0000035A FF                                 db 0FFh                 ; sectors/allocation unit
0000035B 0100                               dw 1                    ; reserved sectors for dos
0000035D 02                                 db 2                    ; no of file allocation       tables
0000035E 4000                               dw 64                   ; number of root directory entries
00000360 6801                               dw 360                  ; number sectors (at 512 bytes each)
00000362 00                                 db 0                    ; media descriptor, initially 0
00000363 0200                               dw 2                    ; number of fat       sectors
00000365 0900                               dw 9                    ; sector limit (sectors       per track)
00000367 0100                               dw 1                    ; head limit (number of       heads - 1)
                                            ;
                                            ; 02/10/2023
                                            ; MSDOS 5.0-6.22 (& PCDOS 7.0)
                                            ;dw 0                   ; hidden sector       count (low word)
                                            ;dw 0                   ; hidden sector       (high)
                                            ;dw 0                   ; number sectors (low)
                                            ;dw 0                   ; number sectors (high)
                                            ;db 0                   ; true => large       fats
                                            ; 02/10/2023
                                            ; PCDOS 7.1 (FAT32 support)
00000369 00000000                           dd 0                    ; hidden sector count
0000036D 00000000                           dd 0                    ; number of sectors (32 bit)
00000371 00000000                           dd 0                    ; BPB_FATSz32 ; FAT32 FAT size in sectors ; 4 bytes
                                                                    ;   BS_DrvNum ; FAT INT 13h drive number ; 1 byte
                                                                    ;   BS_Reserved1 ; FAT reserved byte = 0 ; 1 byte
                                                                    ;   BS_BootSig ; FAT Extended boot signature = 29h ; 1 byte
                                                                    ;   BS_VolID ; FAT Volume serial number ; 4 bytes
00000375 0000                               dw 0                    ; BPB_ExtFlags ; FAT32 Extended Flags
00000377 0000                               dw 0                    ; BPB_FSVer ; FAT32 fs/volume version
00000379 00000000                           dd 0                    ; BPB_RootClus ; FAT32 root directory's first cluster number
0000037D FFFF                               dw 0FFFFh               ; BPB_FSInfo ; FAT32 FSINFO sector number = -1 (initial)
0000037F FFFF                               dw 0FFFFh               ; BPB_BkBootSec ; FAT32 backup boot sector number = -1 (initial)
00000381 00<rep Ch>                         times 12 db 0           ; BPB_Reserved  ; FAT32 reserved field = 0, 12 bytes
0000038D 00                                 db 0                    ; true => large       fats
                                            ;
0000038E 0000                               dw 0                    ; open ref. count
00000390 03                                 db 3                    ; form factor
00000391 2000                               dw 20h                  ; various flags
00000393 2800                               dw 40                   ; number of cylinders
00000395 0002                   recommended_bps: dw 512                     ; recommended bps for this drive
00000397 01                                 db 1
00000398 0100                               dw 1
0000039A 02                                 db 2
0000039B E000                               dw 224                  ; number of root directory entries
0000039D 6801                               dw 360
0000039F F0                                 db 0F0h                 ; media descriptor, initially 0F0h
000003A0 0200                               dw 2
000003A2 0900                               dw 9
000003A4 0200                               dw 2
                                            ;
                                            ; 02/10/2023
                                            ;dw 0
                                            ;dw 0
                                            ;dw 0
                                            ;dw 0
                                            ;;db 6 dup(0)
                                            ;times 6 db 0           ; 19/10/2022
000003A6 00000000                           dd 0
000003AA 00000000                           dd 0
000003AE 00000000                           dd 0
000003B2 0000                               dw 0
000003B4 0000                               dw 0
000003B6 00000000                           dd 0
000003BA FFFF                               dw 0FFFFh
000003BC FFFF                               dw 0FFFFh
```

```
                                        ;db 12 dup(0)
000003BE 00<rep Ch>                     times 12 db 0           ; 02/10/2023
                                        ;
000003CA FF                             db 0FFh                 ; last track accessed on this drive
000003CB FFFF                           dw 0FFFFh               ; keep these two contiguous (?)
000003CD FFFF                           dw 0FFFFh
000003CF 4E4F204E414D452020-            db 'NO NAME    ',0      ; volume id for this disk
000003D8 202000
000003DB 00000000                       dd 0                    ; current volume serial      from boot record
000003DF 464154313220202000             db 'FAT12   ',0         ; current file system id from boot record
                                ; ----

                                ; 02/10/2023
                                ; PCDOS 7.1
                                %if 1

                                bds2:               ; 02/10/2023 - Retro DOS v5.0 (Modified PCDOS 7.1)
000003E8 FFFF                           dw 0FFFFh ; -1
000003EA 7000                           dw 70h
000003EC 00                             db 0
000003ED 00                             db 0
000003EE 0002                   fdrive2:    dw 512
000003F0 FF                             db 0FFh
000003F1 0100                           dw 1
000003F3 02                             db 2
000003F4 4000                           dw 64
000003F6 6801                           dw 360
000003F8 00                             db 0
000003F9 0200                           dw 2
000003FB 0900                           dw 9
000003FD 0100                           dw 1
000003FF 00000000<rep 5h>               times 5 dd 0
00000413 FFFFFFFF                       dd 0FFFFFFFFh
00000417 00000000<rep 3h>               times 3 dd 0
00000423 00                             db 0
00000424 0000                           dw 0
00000426 03                             db 3
00000427 2000                           dw 20h
00000429 2800                           dw 40
                                recbpb2:
0000042B 0002                           dw 512
0000042D 01                                 db 1
0000042E 0100                               dw 1
00000430 02                                 db 2
00000431 E000                               dw 224
00000433 6801                               dw 360
00000435 F0                                 db 0F0h
00000436 0200                               dw 2
00000438 0900                               dw 9
0000043A 0200                               dw 2
0000043C 00000000<rep 5h>               times 5 dd 0
00000450 FFFFFFFF                       dd 0FFFFFFFFh
00000454 00000000<rep 3h>                   times 3 dd 0
00000460 FF                                 db 0FFh
00000461 FFFFFFFF                           dd 0FFFFFFFFh
00000465 4E4F204E414D452020-                db 'NO NAME    ',0
0000046E 202000
00000471 00000000                           dd 0
00000475 464154313220202000                 db 'FAT12   ',0
                                %endif

                                ; ----

                                ; 02/10/2023
                                ; MSDOS 5.0 - 6.22 (& PCDOS 7.0)
                                %if 0

                                bds2:               dw bds3
                                            dw 70h
                                            db 0
                                            db 0
                                fdrive2:    dw 512
                                            db 0FFh, 1, 0, 2, 40h, 0, 68h, 1, 0, 2,     0, 9, 0, 1, 0
                                            db 0, 0, 0, 0, 0, 0, 0,       0, 0, 0, 0, 3, 20h, 0, 28h, 0
                                            db 0, 2, 1, 1, 0, 2, 0E0h, 0, 68h, 1, 0F0h, 2, 0, 9, 0
                                            db 2, 0, 0, 0, 0, 0, 0,       0, 0, 0, 0, 0, 0, 0, 0,       0, 0FFh
                                            db 0FFh, 0FFh, 0FFh, 0FFh, 4Eh,      4Fh, 20h, 4Eh, 41h, 4Dh
                                            db 45h,20h, 20h, 20h, 20h, 0, 0, 0, 0,      0, 46h,41h, 54h
                                            db 31h,32h, 20h, 20h, 20h, 0

                                bds3:               dw bds4
                                            dw 70h
                                            db 0
                                            db 0
                                fdrive3:    dw 512
                                            db 0FFh, 1, 0, 2, 40h, 0, 68h, 1, 0, 2,     0, 9, 0, 1, 0
                                            db 0, 0, 0, 0, 0, 0, 0,       0, 0, 0, 0, 3, 20h, 0, 28h, 0
                                            db 0, 2, 1, 1, 0, 2, 0E0h, 0, 68h, 1, 0F0h, 2, 0, 9, 0
                                            db 2, 0, 0, 0, 0, 0, 0,       0, 0, 0, 0, 0, 0, 0, 0,       0, 0FFh
                                            db 0FFh, 0FFh, 0FFh, 0FFh, 4Eh,      4Fh, 20h, 4Eh, 41h, 4Dh
                                            db 45h,20h, 20h, 20h, 20h, 0, 0, 0, 0,      0, 46h,41h, 54h
                                            db 31h,32h, 20h, 20h, 20h, 0
                                ; ----

                                bds4:               dw 0FFFFh
                                            dw 70h
                                            db 0
                                            db 0
                                fdrive4:    dw 512
                                            db 0FFh, 1, 0, 2, 40h, 0, 68h, 1, 0, 2,     0, 9, 0, 1, 0
                                            db 0, 0, 0, 0, 0, 0, 0,       0, 0, 0, 0, 3, 20h, 0, 28h, 0
                                            db 0, 2, 1, 1, 0, 2, 0E0h, 0, 68h, 1, 0F0h, 2, 0, 9, 0
                                            db 2, 0, 0, 0, 0, 0, 0,       0, 0, 0, 0, 0, 0, 0, 0,       0, 0FFh
                                            db 0FFh, 0FFh, 0FFh, 0FFh, 4Eh,      4Fh, 20h, 4Eh, 41h, 4Dh
                                            db 45h,20h, 20h, 20h, 20h, 0, 0, 0, 0,      0, 46h,41h, 54h
                                            db 31h,32h, 20h, 20h, 20h, 0

                                ;--------------------------------------------------------------------------

                                sm92:               db 3                    ; .spf
                                            db 9                    ; .spt
                                            db 112  ; 70h           ; .cdire
                                            dw 1440 ; 2*9*80        ; .csec
                                            db 2                    ; .spau
                                            db 2                    ; .chead

                                %endif

0000047E 00                     keyrd_func: db 0
0000047F 01                     keysts_func:    db 1
00000480 00                     printdev:   db 0                    ; printer device index

                                wait_count: ;dw 4 dup(50h)          ; retry counts for printers
00000481 5000<rep 4h>                       times 4 dw 50h          ; 19/10/2022
```

```
1605
1606 00000489 0000                   daycnt:             dw 0
1607 0000048B 00                    t_switch:   db 0                    ; flag for updating daycnt
1608 0000048C 00                    havecmosclock:      db 0
1609 0000048D 13                    base_century:       db 19
1610 0000048E 50                    base_year:  db 80
1611
1612 0000048F 1F                    month_tab:  db 31
1613 00000490 1C                    february:   db 28 ; 08/08/2023
1614 00000491 1F1E1F1E1F1F1E1F1E-               db 31, 30, 31, 30, 31, 31, 30, 31, 30, 31
1614 0000049A 1F
1615
1616                                ; 08/08/2023 - Retro DOS v4.2 IO.SYS (optimization)
1617                                %if 0
1618                                bintobcd:   dw bin_to_bcd           ; points to bin_to_bcd proc in msinit
1619                                            dw 70h ; 17/10/2022
1620                                daycnttoday:        dw daycnt_to_day        ; points to daycnt_to_day in msinit
1621                                            dw 70h ; 17/10/2022
1622                                %endif
1623
1624 0000049B 00                    set_id_flag:        db 0                    ; flag for getbp routine
1625
1626                                ; 02/10/2023 - Retro DOS v5.0 IBMBIO.COM (Modified PCDOS 7.1)
1627                                ;fat_12_id: db 'FAT12   ',0
1628                                ;fat_16_id: db 'FAT16   ',0
1629                                ;vol_no_name:       db 'NO NAME    ',0
1630                                ;temp_h:    dw 0                    ; temporary for       32 bit calculation
1631
1632 0000049C 0000                   start_sec_h:        dw 0                    ; starting sector number high word
1633 0000049E 0000                   saved_word: dw 0                    ; tempory saving place for a word
1634 000004A0 0000                   multrk_flag:        dw 0
1635 000004A2 00                    ec35flag:   db 0                    ; flags for 3.5 inch disk drives
1636 000004A3 0000                   vretry_cnt: dw 0
1637 000004A5 0000                   soft_ecc_cnt:       dw 0
1638 000004A7 00                    multitrk_format_flag: db 0          ; multi track format request flag
1639 000004A8 0000                   xfer_seg:   dw 0                    ; temp for transfer segment
1640
1641                                ; variables for msdioctl.asm module
1642
1643                                ; tracktable contains a 4-tuples (c,h,r,n) for each sector in a track
1644                                ; c = cylinder number,h = head number,r = sector id,n = bytes per sector
1645                                ;   n       bytes per sector
1646                                ;      ---  ----------------
1647                                ;   0             128
1648                                ;   1             256
1649                                ;   2             512
1650                                ;   3            1024
1651
1652                                ;max_sectors_curr_sup equ 63                ; current maximum sec/trk that
1653                                ;                                   ; we support (was 40 in dos 3.2)
1654
1655 000004AA 2400                   sectorspertrack: dw 36
1656 000004AC 00000102               tracktable: db 0, 0, 1, 2
1657 000004B0 00000202                           db 0, 0, 2, 2
1658 000004B4 00000302                           db 0, 0, 3, 2
1659 000004B8 00000402                           db 0, 0, 4, 2
1660 000004BC 00000502                           db 0, 0, 5, 2
1661 000004C0 00000602                           db 0, 0, 6, 2
1662 000004C4 00000702                           db 0, 0, 7, 2
1663 000004C8 00000802                           db 0, 0, 8, 2
1664 000004CC 00000902                           db 0, 0, 9, 2
1665 000004D0 00000A02                           db 0, 0, 10, 2
1666 000004D4 00000B02                           db 0, 0, 11, 2
1667 000004D8 00000C02                           db 0, 0, 12, 2
1668 000004DC 00000D02                           db 0, 0, 13, 2
1669 000004E0 00000E02                           db 0, 0, 14, 2
1670 000004E4 00000F02                           db 0, 0, 15, 2
1671 000004E8 00001002                           db 0, 0, 16, 2
1672 000004EC 00001102                           db 0, 0, 17, 2
1673 000004F0 00001202                           db 0, 0, 18, 2
1674 000004F4 00001302                           db 0, 0, 19, 2
1675 000004F8 00001402                           db 0, 0, 20, 2
1676 000004FC 00001502                           db 0, 0, 21, 2
1677 00000500 00001602                           db 0, 0, 22, 2
1678 00000504 00001702                           db 0, 0, 23, 2
1679 00000508 00001802                           db 0, 0, 24, 2
1680 0000050C 00001902                           db 0, 0, 25, 2
1681 00000510 00001A02                           db 0, 0, 26, 2
1682 00000514 00001B02                           db 0, 0, 27, 2
1683 00000518 00001C02                           db 0, 0, 28, 2
1684 0000051C 00001D02                           db 0, 0, 29, 2
1685 00000520 00001E02                           db 0, 0, 30, 2
1686 00000524 00001F02                           db 0, 0, 31, 2
1687 00000528 00002002                           db 0, 0, 32, 2
1688 0000052C 00002102                           db 0, 0, 33, 2
1689 00000530 00002202                           db 0, 0, 34, 2
1690 00000534 00002302                           db 0, 0, 35, 2
1691 00000538 00002402                           db 0, 0, 36, 2
1692
1693                                ; 02/10/2023 - Retro DOS v5.0 IBMBIO.COM (Modified PCDOS 7.1)
1694                                ; PCDOS 7.1 IBMBIO.COM - BIOSDATA:053Ch
1695
1696                                            ;times 108 db 0             ; 19/10/2022
1697                                            ;;db 108 dup(0)             ; 4*max_sectors_curr_sup - ($ -      tracktable) dup       (0)
1698                                                                ; times ((4*63) - 144) db 0
1699                                dskdrvs:
1700 0000053C [5803]                           dw fdrive1
1701 0000053E [EE03]                           dw fdrive2
1702
1703                                            ;dw 52 dup(0)
1704 00000540 00<rep 68h>                      times 104 db 0          ; times (((4*63)-144)-4) db 0
1705                                                                ; 4*max_sectors_curr_sup-($-tracktable)-4 dup (0)
1706
1707                                ;--------------------------------------------------------------------------
1708
1709                                ; this is a real ugly place to put this
1710                                ; it should really go in the bds
1711
1712 000005A8 00                    mediatype:  db 0
1713 000005A9 00                    media_set_for_format: db 0          ; 1 if we have done an int 13h set media
1714                                                                ; type for format call
1715 000005AA 00                    had_format_error: db 0                  ; 1 if the previous format operation
1716                                                                ; failed.
1717
1718                                ; temp disk base table. it holds the the current dpt which is then replaced by
1719                                ; the one passed by "new roms" before we perform a format operation. the old
1720                                ; dpt is restored in restoreolddpt. the first entry (disk_specify_1) is -1 if
1721                                ; this table does not contain the previously saved dpt.
1722
1723 000005AB FFFFFFFF               tempdpt:    dd 0FFFFFFFFh ; -1      ; temp disk base table
1724 000005AF FF                    model_byte: db 0FFh                 ; model byte set at init time
1725 000005B0 00                    secondary_model_byte: db 0
1726
1727 000005B1 00                    int19sem:   db 0                    ; indicate that all int 19h
```

```
                                                        ; initialization is complete

                        ;; we assume the following remain contiguous and their order doesn't change
                        ;i19_lst:
                        ;   irp    aa,<02,08,09,0a,0b,0c,0d,0e,70,72,73,74,76,77>
                        ;   public int19old&aa
                        ;          db     aa&h    ; store the number as a byte
                        ;int19old&aa       dd     -1      ; original hardware int. vectors for int 19h.
                        ;   endm

                        ; 21/10/2022

000005B2 02             i19_lst:    db 2
                                                        ; Int19old&aa
000005B3 FFFFFFFF       int19old02: dd 0FFFFFFFFh ; -1
000005B7 08                         db 8
000005B8 FFFFFFFF       int19old08: dd 0FFFFFFFFh         ; original hardware int. vectors for int 19h
000005BC 09                         db 9
000005BD FFFFFFFF       int19old09: dd 0FFFFFFFFh
000005C1 0A                         db 0Ah
000005C2 FFFFFFFF       int19old0A: dd 0FFFFFFFFh
000005C6 0B                         db 0Bh
000005C7 FFFFFFFF       int19old0B: dd 0FFFFFFFFh
000005CB 0C                         db 0Ch
000005CC FFFFFFFF       int19old0C: dd 0FFFFFFFFh
000005D0 0D                         db 0Dh
000005D1 FFFFFFFF       int19old0D: dd 0FFFFFFFFh
000005D5 0E                         db 0Eh
000005D6 FFFFFFFF       int19old0E: dd 0FFFFFFFFh
000005DA 70                         db 70h
000005DB FFFFFFFF       int19old70: dd 0FFFFFFFFh
000005DF 72                         db 72h
000005E0 FFFFFFFF       int19old72: dd 0FFFFFFFFh
000005E4 73                         db 73h
000005E5 FFFFFFFF       int19old73: dd 0FFFFFFFFh
000005E9 74                         db 74h
000005EA FFFFFFFF       int19old74: dd 0FFFFFFFFh
000005EE 76                         db 76h
000005EF FFFFFFFF       int19old76: dd 0FFFFFFFFh
000005F3 77                         db 77h
000005F4 FFFFFFFF       int19old77: dd 0FFFFFFFFh

                        ;num_i19    equ ($ - i19_lst)/5  ; 18/03/2019

                        ;--------------------------------------------------------------------------

                        ; 02/10/2023 - Retro DOS v5.0 IBMBIO.COM (Modified PCDOS 7.1)
                        ;
                        ;dskdrvs:   dw fdrive1
                        ;           dw fdrive2
                        ;           dw fdrive3
                        ;           dw fdrive4
                        ;
                        ;;M011 -- made all hard drive stuff variable
                        ;           ;dw 22 dup(0)          ; up to 26 drives for mini disks
                        ;           times 22 dw 0  ; 19/10/2022

                        ;--------------------------------------------------------------------------

                        ; 01/10/2022 - Retro DOS v4.0 (MSDOS v5.0 -actual-)
                        ; 30/12/2018 - Retro DOS v4.0 (MSDOS v6.21 -draft-)
                        ; 01/06/2018 - Retro DOS v3.0 (MSDOS v3.3)

                        ;variables for dynamic relocatable modules
                        ;these should be stay resident.

000005F8 00000000       int6c_ret_addr:    dd 0                 ; return address from int 6Ch
                                                        ; for p12 machine

                        ; data structures for real-time date and time

000005FC 00000000       bin_date_time:     db 0, 0, 0, 0        ; century, year, month,      day

                        ; 08/08/2023 - Retro DOS v4.2 IO.SYS (optimization)
                        %if 0
                        month_table:       dw 0                 ; january
                                   dw 31                ; february
                                   dw 59
                                   dw 90
                                   dw 120
                                   dw 151
                                   dw 181
                                   dw 212
                                   dw 243
                                   dw 273
                                   dw 304
                                   dw 334               ; december
                        %endif

00000600 0000           daycnt2:   dw 0
                        ; 08/08/2023
                        ;feb29:            db 0                 ; february 29 in a leap      year flag

                        ;--------------------------------------------------------------------------
                        ;
                        ; 01/10/2022 - (New/Actual) Retro DOS v4.0 (will run as MSDOS 5.0)
                        ; by Erdogan Tan (Istanbul) ! free source code !
                        ; 31/12/2018 - (old/draft) Retro DOS v4.0 (will/would run as MSDOS 6.21)

                        ; -------------------------------------------------------------------------

                        ;*************************************************************************
                        ;*                                                                       *
                        ;*  Entry points into Bios_Code routines. The segment values   *
                        ;*    are plugged in by seg_reinit.                                      *
                        ;*                                                                       *
                        ;*************************************************************************

                        ; 01/10/2022 - Retro DOS v4.0 - IO.SYS (MSDOS v5.0)
                        ; BIOSCODE_SEGMENT equ 2C7h
                        ; BIOSDATA_SEGMENT equ 70h ; KERNEL_SEGMENT equ 70h

                        ; 01/10/2022 - Erdogan Tan
                        ; (disassembled MSDOS 5.0 IO.SYS code here with fixed function/routine
                        ;  addresses, they will be changed to table labels later)

                        ; 09/12/2022
                        %if 0
                        cdev:              dw 43h,2C7h          ; chardev_entry
                                                        ; at 2C7h:43h =       70h:25B3h
                        ttticks:    dw 396h, 2C7h       ; time_to_ticks
                                                        ; at 2C7h:396h = 70h:2906h
                        bcode_i2f:  dw 1302h, 2C7h      ; i2f_handler
                                                        ; at 2C7h:1302h       = 70h:3872h
```

```
                                i13x:           dw 154Bh, 2C7h          ; i13z
                                                                        ; at 2C7h:154Bh       = 70h:3ABBh
                                %endif

                                ; 30/12/2022
                                ; (IOSYSCODESEG is 2CCh for MSDOS 6.21 IO.SYS)

                                ; 02/10/2023 - Retro DOS v5.0 IBMBIO.COM (Modified PCDOS 7.1)
                                ; PCDOS 7.1 IBMBIO.COM - BIOSDATA:0602h
                                ; (IOSYSCODESEG is 364h for PCDOS 7.1 IBMBIO.COM)

                                ; 09/12/2022
00000602 [4700]0203             cdev:           dw chardev_entry, IOSYSCODESEG
00000606 [A003]0203             ttticks:    dw time_to_ticks, IOSYSCODESEG
                                ; 07/08/2023
                                ;bcode_i2f: dw i2f_handler, IOSYSCODESEG
0000060A [5718]0203             i13x:           dw i13z, IOSYSCODESEG

                                end_BC_entries:    ; 15/10/2022

                                ;****************************************************************************
                                ;*                                                                          *
                                ;*  cbreak - break key handling - simply set altah=3 and iret               *
                                ;*                                                                          *
                                ;****************************************************************************

                                cbreak:
0000060E 2EC606[0C00]03                 mov     byte [cs:altah], 3 ; break key handling
                                                                    ; indicate break key set
                                intret:
00000614 CF                             iret

                                ; ============== S U B   R O U T I N E =======================================


                                ;****************************************************************************
                                ;*                                                                          *
                                ;*  strategy - store es:bx (device driver request packet)                   *
                                ;*             away at [ptrsav] for next driver function call               *
                                ;*                                                                          *
                                ;****************************************************************************

                                strategy:   ; proc far
00000615 2E891E[1200]                   mov     [cs:ptrsav], bx ; store es:bx (device driver request packet)
                                                                        ; away at [ptrsav] for next driver function call
0000061A 2E8C06[1400]                   mov     [cs:ptrsav+2], es
0000061F CB                             retf

                                ; ---------------------------------------------------------------------------

                                ;****************************************************************************
                                ;*                                                                          *
                                ;*  device driver entry points. these are the initial                       *
                                ;*    'interrupt' hooks out of the device driver chain.                     *
                                ;*    in the case of our resident drivers, they'll just                     *
                                ;*    stick a fake return address on the stack which                        *
                                ;*    points to dispatch tables and possibly some unit                      *
                                ;*    numbers, and then call through a common entry point                       *
                                ;*    which can take care of a20 switching                                  *
                                ;*                                                                          *
                                ;****************************************************************************

                                ; 01/10/2022 - Erdogan Tan
                                ; (disassembled MSDOS 5.0 IO.SYS code here with fixed table
                                ;  addresses, they will be changed to table labels later)

                                ; 09/12/2022

                                ; 02/10/2023 - Retro DOS v5.0
                                ; (PCDOS 7.1 IBMBIO.COM - BIOSDATA:0620h, BIOSCODE = 0364h)

                                con_entry:
00000620 E84000                         call    cdev_entry
                                ; ---------------------------------------------------------------------------
                                                                    ;dw 0E4h            ; con_table
00000623 [E400]                         dw con_table                ; 364h:0E4h (PCDOS 7.1)
                                                                    ; 2C7h:0E4h = 70h:2654h
                                ; ---------------------------------------------------------------------------

                                prn0_entry:
00000625 E83B00                         call    cdev_entry
                                ; ---------------------------------------------------------------------------
                                                                    ;dw 0FBh            ; prn_table
00000628 [FB00]                         dw prn_table
                                                                    ; 2C7h:0FBh = 70h:266Bh
0000062A 0000                           db 0, 0
                                ; ---------------------------------------------------------------------------

                                prn1_entry:
0000062C E83400                         call    cdev_entry
                                ; ---------------------------------------------------------------------------
                                                                    ;dw 0FBh            ; prn_table
0000062F [FB00]                         dw prn_table
                                                                    ; 2C7h:0FBh = 70h:266Bh
00000631 0001                           db 0, 1
                                ; ---------------------------------------------------------------------------

                                prn2_entry:
00000633 E82D00                         call    cdev_entry
                                ; ---------------------------------------------------------------------------
                                                                    ;dw 0FBh            ; prn_table
00000636 [FB00]                         dw prn_table
                                                                    ; 2C7h:0FBh = 70h:266Bh
00000638 0102                           db 1, 2
                                ; ---------------------------------------------------------------------------

                                prn3_entry:
0000063A E82600                         call    cdev_entry
                                ; ---------------------------------------------------------------------------
                                                                    ;dw 0FBh            ; prn_table
0000063D [FB00]                         dw prn_table
                                                                    ; 2C7h:0FBh = 70h:266Bh
0000063F 0203                           db 2, 3
                                ; ---------------------------------------------------------------------------

                                aux0_entry:
00000641 E81F00                         call    cdev_entry
                                ; ---------------------------------------------------------------------------
                                                                    ;dw 130h            ; aux_table
00000644 [3001]                         dw aux_table
                                                                    ; 2C7h:130h = 70h:26A0h
00000646 00                             db 0
                                ; ---------------------------------------------------------------------------

```

```
                                aux1_entry:
00000647 E81900                         call    cdev_entry
                                ; ---------------------------------------------------------------------------
                                        ;dw 130h               ; aux_table
0000064A [3001]                         dw aux_table           ; 364h:130h = 70h:3070h (PCDOS 7.1)
                                                               ; 2C7h:130h = 70h:26A0h
0000064C 01                             db 1
                                ; ---------------------------------------------------------------------------

                                aux2_entry:
0000064D E81300                         call    cdev_entry
                                ; ---------------------------------------------------------------------------
                                        ;dw 130h               ; aux_table
00000650 [3001]                         dw aux_table
                                                               ; 2C7h:130h = 70h:26A0h
00000652 02                             db 2
                                ; ---------------------------------------------------------------------------

                                aux3_entry:
00000653 E80D00                         call    cdev_entry
                                ; ---------------------------------------------------------------------------
                                        ;dw 130h               ; aux_table
00000656 [3001]                         dw aux_table
                                                               ; 2C7h:130h = 70h:26A0h
00000658 03                             db 3
                                ; ---------------------------------------------------------------------------

                                tim_entry:
00000659 E80700                         call    cdev_entry
                                ; ---------------------------------------------------------------------------
                                        ;dw 147h               ; tim_table
0000065C [4701]                         dw tim_table           ; 364h:147h = 70h:3087h (PCDOS 7.1)
                                                               ; 2C7h:147h = 70h:26B7h
                                ; ---------------------------------------------------------------------------

                                ; 15/10/2022
                                ;DSKTBL     equ dsktbl - DOSBIOSEG_2C7h   ; dsktbl - 2C70h
                                ; 09/12/2022
                                DSKTBL equ dsktbl

                                dsk_entry:
0000065E E80200                         call    cdev_entry
                                ; ---------------------------------------------------------------------------
                                        ;dw 4A2h               ; dsktbl
00000661 [6F05]                         dw DSKTBL              ; 09/12/2022
                                                               ; 2C7h:4A2h = 70h:2A12h
                                                               ; 02/10/2023 (PCDOS 7.1 IBMBIO.COM)
                                                               ; 364h:579h = 70h:34B9h

                                ; ============== S U B   R O U T I N E =======================================

                                ;****************************************************************************
                                ;*                                                                          *
                                ;*  Ensure A20 is enabled before jumping into code in HMA.                  *
                                ;*  This code assumes that if Segment of Device request packet is           *
                                ;*  DOS DATA segment then the Device request came from DOS & that           *
                                ;*  A20 is already on.                                                      *
                                ;*                                                                          *
                                ;****************************************************************************

                                cdev_entry: ; proc near
00000663 2E803E[0D00]00                 cmp     byte [cs:inHMA],0
00000669 740D                           jz      short ce_enter_codeseg
                                                ; optimized for DOS in HMA
0000066B 50                             push    ax
0000066C 2EA1[0300]                     mov     ax,[cs:DosDataSg]
00000670 2E3906[1400]                   cmp     [cs:ptrsav+2],ax
00000675 58                             pop     ax
00000676 7505                           jnz     short not_from_dos
                                                ; jump is coded this way to fall thru
                                                ; in 99.99% of the cases
                                ce_enter_codeseg:
00000678 2EFF2E[0206]                   jmp     far [cs:cdev]
                                        ;jmp    dword ptr cs:cdev
                                ;---------------------------------------------------------------------------

                                not_from_dos:
0000067D E8AA00                         call    EnsureA20On
00000680 EBF6                           jmp     short ce_enter_codeseg

                                ;****************************************************************************
                                ;*                                                                          *
                                ;*  outchr - this is our int 29h handler. it writes the                     *
                                ;*     character in al on the display using int 10h ttywrite                *
                                ;*                                                                          *
                                ;****************************************************************************

                                    ; 17/07/2024
                                    ; 02/10/2023
                                    ; PCDOS 7.1 IBMBIO.COM - BIOSDATA:0682h
                                outchr:
00000682 50                             push    ax              ; int 29h handler
00000683 56                             push    si
00000684 57                             push    di
00000685 55                             push    bp
00000686 53                             push    bx
                                        ;;;
                                        ; 02/10/2023 - Retro DOS v5.0 (Modified POCDOS 7.1)
                                        ;mov    ah,0Eh
                                        ;mov    bx,7
                                        ;int    10h     ; - VIDEO - WRITE CHARACTER AND      ADVANCE CURSOR (TTY WRITE)
                                        ;               ; AL = character, BH = display page (alpha modes)
                                        ;               ; BL = foreground color       (graphics modes)
                                        ; 17/07/2024
                                        ; 02/10/2023
00000687 1E                             push    ds ; *
00000688 31DB                           xor     bx,bx ; 0
0000068A 8EDB                           mov     ds,bx ; 0
0000068C B40E                           mov     ah,0Eh
0000068E B307                           mov     bl,7
00000690 2E381E[1208]                   cmp     [cs:IsWin386], bl ; (are we in) Windows ?
                                        ; 17/07/2024
                                        ;jnz    short win_outchr ; *
                                        ;push   ds ; *
                                        ;mov    ds,bx ; 0
                                        ;mov    ah,0Eh
                                        ;mov    bl,7
00000695 7508                           jnz     short win_outchr ; Running on Windows
00000697 9C                             pushf                   ; far call (simulate INT)
00000698 FA                             cli     ; disable interrupts
00000699 FF1E4000                       call    far [40h]       ; far call to INT 10h vector
                                        ; 17/07/2024
                                        ;pop    ds ; *
0000069D EB02                           jmp     short outchr_ok
```

```
                                win_outchr:
000006A0 CD10                           int     10h
                                outchr_ok:
                                        ; 17/07/2024
000006A2 1F                             pop     ds ; *
                                        ;;;
000006A3 5B                             pop     bx
000006A4 5D                             pop     bp
000006A5 5F                             pop     di
000006A6 5E                             pop     si
000006A7 58                             pop     ax
000006A8 CF                             iret

                                ;--------------------------------------------------------------------------

                                    ; 02/10/2023 - Retro DOS v5.0
                                    ; PCDOS 7.1 IBMBIO.COM - BIOSDATA:06A8h

000006A9 50                             db 50h ; P              ; 'PCI' signature
000006AA 43                             db 43h ; C
000006AB 49                             db 49h ; I

000006AC 00000000               Orig1A:         dd 0

                                ; =============== S U B R O U T I N E =======================================

                                    ; 02/10/2023 - Retro DOS v5.0
                                    ; PCDOS 7.1 IBMBIO.COM - BIOSDATA:06AFh

                                Int1A:
000006B0 80FC04                         cmp     ah,4            ; (Y2K-fix)
000006B3 7405                           je      short int1a_1   ; Read the date from the computer's real-time clock
000006B5 2EFF2E[AC06]                   jmp     far [cs:Orig1A]
                                int1a_1:
000006BA 55                             push    bp
                                int1a_2:
000006BB 89E5                           mov     bp,sp
000006BD 55                             push    bp
000006BE 9C                             pushf
000006BF 2EFF1E[AC06]                   call    far [cs:Orig1A]
000006C4 7220                           jc      short int1a_4

                                        ;cmp    cl,0            ; Year (BCD)
                                        ; 02/10/2023
000006C6 08C9                           or      cl,cl
000006C8 7515                           jnz     short int1a_3
000006CA 80FD19                         cmp     ch,19h          ; Century (BCD)
000006CD 7510                           jne     short int1a_3
000006CF B520                           mov     ch,20h
000006D1 B405                           mov     ah,5            ; Set the date on the computer's real-time clock
000006D3 51                             push    cx
000006D4 52                             push    dx              ; dh = Month (BCD), dl = Day (BCD)
000006D5 9C                             pushf
000006D6 2EFF1E[AC06]                   call    far [cs:Orig1A]
000006DB 5A                             pop     dx
000006DC 59                             pop     cx
000006DD 7207                           jc      short int1a_4
                                int1a_3:
000006DF 5D                             pop     bp
000006E0 806606FE                       and     byte [bp+6],0FEh ; clear carry flag
000006E4 EB05                               jmp       short int1a_5
                                int1a_4:
000006E6 5D                             pop     bp
000006E7 804E0601                       or      byte [bp+6],1  ; set carry flag
                                int1a_5:
000006EB 5D                             pop     bp
000006EC CF                             iret

                                        ; 02/10/2023
000006ED 90                             nop     ; (not necessary, i have used this 'nop' to locate 'block13:'
                                                ; at BIOSDATA:06EDh, just as in the original PCDOS 7.1 IBMBIO.COM)

                                ;--------------------------------------------------------------------------

                                ;**************************************************************************
                                ;*                                                                        *
                                ;*  block13 - our int13 hooker                                            *
                                ;*                                                                        *
                                ;**************************************************************************

                                    ; 02/10/2023 - Retro DOS v5.0
                                    ; PCDOS 7.1 IBMBIO.COM - BIOSDATA:06EDh

                                block13:
000006EE 2E803E[0D00]00                 cmp     byte [cs:inHMA],0
000006F4 7403                           jz      short skipa20

                                        ;call   IsA20Off        ; A20 Off?
                                        ;jnz    short skipa20
                                        ;call   EnableA20       ; assure a20 enabled
                                        ; 02/10/2023 -  Retro DOS v5.0 (Modified PCDOS 7.1)
000006F6 E83200                         call    EnsureA20On     ; assure a20 enabled
                                skipa20:
000006F9 2E8C1E[1C00]                   mov     [cs:i13_ds],ds ; save caller's ds for call-through
000006FE 9C                             pushf                   ; fake interrupt
000006FF 2EFF1E[0A06]                   call    far [cs:i13x]
                                        ;call   dword ptr cs:i13x
                                                                ; call through Bios_Code entry table
00000704 2E8E1E[1C00]                   mov     ds,[cs:i13_ds]
00000709 CA0200                         retf    2

                                ; =============== S U B   R O U T I N E =======================================

                                ; the int13 hook calls back here to call-through to the ROM
                                ; this is necessary because some people have extended their
                                ; ROM BIOSs to use ds as a parameter/result register and
                                ; our int13 hook relies heavily on ds to access Bios_Data

                                call_orig13:        ; proc far
0000070C 8E1E[1C00]                     mov     ds,[i13_ds]     ; get caller's ds register
00000710 9C                             pushf                   ; simulate an int13
00000711 2EFF1E[B400]                   call    far [cs:Orig13]
                                        ;call   cs:Orig13
00000716 2E8C1E[1C00]                   mov     [cs:i13_ds],ds
0000071B 0E                             push    cs
0000071C 1F                             pop     ds              ; restore ds ->       Bios_Data before return

0000071D 9C                             pushf
                                        ; 10/12/2022
                                        ; ds = cs
0000071E 803E[0D00]00                   cmp     byte [inHMA],0 ; 16/10/2022
                                        ;cmp    byte [cs:inHMA],0
00000723 7403                           jz      short corig13_popf_retf
                                        ; 07/08/2023 - Retro DOS v4.2 IO.SYS (optimization)
```

```
                                        ; PCDOS 7.1 IBMBIO.COM - BIOSDATA:0725h
                                        ;call   IsA20Off
                                        ;jnz    short corig13_popf_retf
                                        ;call   EnableA20
00000724 E80300                         call    EnsureA20On ; 07/08/2023
                                corig13_popf_retf:
                                        popf
                                        ; 20/09/2023
                                re_init:        ; 07/08/2023
00000728 CB                             retf

                                        ; 02/10/2023
00000729 90                             nop     ; (not necessary, i have used this 'nop' to locate 'EnsureA20On:'
                                                ; at BIOSDATA:072Ah, just as in the original PCDOS 7.1 IBMBIO.COM)

                                ;--------------------------------------------------------------------------

                                ; BIOSDATA:07BBh (MSDOS 6.21, IO.SYS)
                                ; BIOSDATA:07BBh (MSDOS 5.0, IO.SYS) ; 16/10/2022

                                ; 07/08/2023 - Retro DOS v4.2 IO.SYS (optimization)
                                ;HiMem:          dd 0FFFF0090h
                                ;LoMem:          dd 80h

                                ; ---------------------------------------------------------------------------

                                ; =============== S U B   R O U T I N E =======================================


                                ;****************************************************************************
                                ;*                                                                          *
                                ;*  EnsureA20On - ensure that a20 is enabled if we're running               *
                                ;*    in the HMA before interrupt entry points into Bios_Code               *
                                ;*                                                                          *
                                ;****************************************************************************

                                EnsureA20On:    ; proc near
0000072A E80E00                         call    IsA20Off
                                        ;jz     short EnableA20
                                        ;retn
                                        ; 18/12/2022
0000072D 750B                           jnz     short A20On_retn

                                ; =============== S U B   R O U T I N E =======================================


                                EnableA20:  ; proc near
0000072F 50                             push    ax
00000730 53                             push    bx
00000731 B405                           mov     ah,5    ; local enable a20
                                        ;call   cs:xms
00000733 2EFF1E[0E00]                   call    far [cs:xms] ; 16/10/2022
00000738 5B                             pop     bx
00000739 58                             pop     ax
                                A20On_retn: ; 18/12/2022
0000073A C3                             retn

                                ; =============== S U B   R O U T I N E =======================================


                                IsA20Off:   ; proc near
0000073B 1E                             push    ds
0000073C 06                             push    es
0000073D 51                             push    cx
0000073E 56                             push    si
0000073F 57                             push    di
                                        ; 07/08/2023 - Retro DOS v4.2 IO.SYS (optimization)
                                        ;lds    si,[cs:HiMem]
                                        ;les    di,[cs:LoMem]
                                        ; PCDOS 7.1 IBMBIO.COM - BIOSDATA:0740h
00000740 31FF                           xor     di,di
00000742 8EC7                           mov     es,di
00000744 4F                             dec     di
00000745 BE9000                         mov     si,90h  ; 0FFFFh:0090h ; HiMem
00000748 8EDF                           mov     ds,di
0000074A BF8000                         mov     di,80h  ; 0000h:0080h ; LoMem
                                        ; 02/10/2023 - Retro DOS v5.0 IBMBIO.COM (PCDOS 7.1)
                                        ; (following cpu instructions will be modified by 'SYSIN'
                                        ; if the cpu is a 386/32bit, for checking A20 line faster)
                                cpu386_cmpsd:
0000074D 90                             nop
0000074E B90800                         mov     cx,8
00000751 F3A7                           repe cmpsw
                                                ; zf = 0 -> A20 line is ON
                                                ; zf = 1 -> A20 line is OFF
00000753 5F                             pop     di
00000754 5E                             pop     si
00000755 59                             pop     cx
00000756 07                             pop     es
00000757 1F                             pop     ds
00000758 C3                             retn

                                ; ---------------------------------------------------------------------------

                                ; 07/08/2023 - Retro DOS v4.2 IO.SYS (optimization)
                                %if 0
                                DisableA20:
                                        push    ax
                                        push    bx
                                        mov     ah,6            ; local disable A20
                                        call    far [cs:xms]
                                        ;call  cs:xms
                                        pop     bx
                                        pop     ax
                                        retn
                                %endif

                                ; ---------------------------------------------------------------------------

                                ;****************************************************************************
                                ;*                                                                          *
                                ;*  int19 - bootstrap interrupt -- we must restore a bunch of the           *
                                ;*    interrupt vectors before resuming the original int19 code             *
                                ;*                                                                          *
                                ;****************************************************************************

                                        ; 02/10/2023 - Retro DOS v5.0
                                        ; PCDOS 7.1 IBMBIO.COM - BIOSDATA:0759h
                                int19:
00000759 0E                             push    cs
0000075A 1F                             pop     ds
                                        ; 07/08/2023 - Retro DOS v4.2 IO.SYS (optimization)
                                        ;mov    es,[zeroseg]    ; 16/10/2022
                                        ;mov    cx,5            ; NUMROMVECTORS
```

```
                                        xor     cx,cx
                                        mov     es,cx
                                        mov     cl,5
                                        ;mov    si,offset RomVectors
00000761 BE[0001]                       mov     si,RomVectors  ; 19/10/2022
                                next_int:
00000764 AC                             lodsb                  ; get int number
00000765 98                             cbw                    ; assume < 128
00000766 D1E0                           shl     ax,1
00000768 D1E0                           shl     ax,1           ; int * 4
                                        ; 07/08/2023
                                        ;mov    di,ax
                                        ;lodsw
                                        ;stosw
                                        ;lodsw
                                        ;stosw                 ; install the saved vector
                                        ; PCDOS 7.1 IBMBIO.COM - BIOSDATA:076Ah
0000076A 97                             xchg    ax,di
0000076B A5                             movsw
0000076C A5                             movsw
0000076D E2F5                           loop    next_int
                                        ;cmp    byte [int19sem], 0 ; 19/10/2022
0000076F 380E[B105]                     cmp     [int19sem], cl ; 0 ; 07/08/2023
00000773 7419                           jz      short doint19
00000775 BE[B205]                       mov     si,i19_lst     ; stacks code has changed these hardware interrupt vectors
                                                               ; stkinit in sysinit1 will initialize int19oldxx values
                                        ;mov    cx,14          ; num_i19
                                        ; 07/08/2023
00000778 B10E                           mov     cl,14
                                i19_restore_loop:
0000077A AC                             lodsb                  ; get interrupt       number
0000077B 98                             cbw                    ; assume < 128
                                        ;mov    di,ax
                                        ;lodsw                 ; get original vector offset
                                        ;mov    bx,ax          ; save it
                                        ;lodsw
                                        ; 07/08/2023
0000077C 97                             xchg    ax,di
0000077D AD                             lodsw
0000077E 93                             xchg    ax,bx
0000077F AD                             lodsw
                                        ;cmp    bx,0FFFFh      ; check for 0ffffh (unlikely segment)
00000780 43                             inc     bx ; 07/08/2023
00000781 7409                           jz      short i19_restor_1 ; opt no need to check selector too
                                        ;cmp    ax,0FFFFh      ; opt 0ffffh is       unlikely offset
                                        ;jz     short i19_restor_1
00000783 4B                             dec     bx ; 07/08/2023
00000784 01FF                           add     di,di
00000786 01FF                           add     di,di
00000788 93                             xchg    ax,bx
00000789 AB                             stosw
0000078A 93                             xchg    ax,bx
0000078B AB                             stosw                  ; put the vector back
                                i19_restor_1:
0000078C E2EC                           loop    i19_restore_loop
                                doint19:
                                        ;cmp    byte [inHMA],0 ; ; Is dos running from      HMA
0000078E 380E[0D00]                     cmp     [inHMA],cl ; 0 ; 07/08/2023
00000792 7403                           jz      short SkipVDisk
00000794 E82A00                         call    EraseVDiskHead ; Then erase our VDISK header at 1MB boundary
                                                               ; Some m/c's (AST 386 & HP QS/16 do not clear
                                                               ; the memory above 1MB during a      warm boot.
                                SkipVDisk:
00000797 CD19                           int     19h            ; DISK BOOT
                                                               ; causes reboot       of disk system

                                ; =============== S U B   R O U T I N E =======================================

                                ;---------------------------------------------------------------------------
                                ;
                                ; procedure : int15
                                ;
                                ;           Int15 handler for recognizing ctrl-alt-del seq
                                ;           If it recognizes ctrl-alt-del and if DOS was
                                ;           is running high, it Erases the VDISK header
                                ;           present at 1MB boundary
                                ;
                                ;---------------------------------------------------------------------------

                                ; 16/10/2022
                                ;DELKEY             equ     53h
                                ;ROMDATASEG        equ     40h
                                KBFLAG              equ     17h
                                ;CTRLSTATE         equ     04h
                                ;ALTSTATE          equ     08h

                                        ; 02/10/2023 - Retro DOS v5.0 (PCDOS 7.1) IBMBIO.COM
                                Int15:
                                        ;cmp    ax,4F00h+DELKEY
00000799 3D534F                         cmp     ax,4F53h       ; del keystroke ?
                                        ; 02/10/2023 - Retro DOS v5.0
                                        ; 07/08/2023
0000079C 7405                           jz      short int15_1
                                        ;jnz    short Old15_j  ; 07/08/2023
                                Old15_j:
0000079E 2EFF2E[0B01]                   jmp     far [cs:Old15] ; 16/10/2022

                                ; ---------------------------------------------------------------------------
                                int15_1:
000007A3 1E                             push    ds
000007A4 50                             push    ax
                                        ; 07/08/2023 - Retro DOS v4.2 IO.SYS (optimization)
                                        ; PCDOS 7.1 IBMBIO.COM - BIOSDATA:07A5h
                                        ;mov    ax,40h         ; ROMDATASEG
                                        ;mov    ds,ax
                                        ;;mov   al,ds:17h      ; [KBFLAG]
                                        ;; 16/10/2022
                                        ;mov    al,[KBFLAG]
000007A5 31C0                           xor     ax,ax
000007A7 8ED8                           mov     ds,ax
000007A9 A01704                         mov     al,[0417h]     ; KBFLAG = 0417h (PCDOS 7.1 IBMBIO.COM)
000007AC 240C                           and     al,0Ch         ; (CTRLSTATE | ALTSTATE)
000007AE 3C0C                           cmp     al,0Ch         ; (CTRLSTATE | ALTSTATE)
000007B0 750A                           jnz     short int15_2
                                        ; 07/08/2023
                                        ;push   cs
                                        ;pop    ds
                                        ;cmp    byte [inHMA],0 ; is DOS running from HMA
000007B2 2E3826[0D00]                   cmp     byte [cs:inHMA],ah ; 0
000007B7 7403                           jz      short int15_2
000007B9 E80500                         call    EraseVDiskHead
                                int15_2:
000007BC 58                             pop     ax
000007BD 1F                             pop     ds
```

```
                                        stc
                                        ; 02/10/2023 - Retro DOS v5.0
000007BF EBDD                           jmp     short Old15_j

                                        ; 02/10/2023
                                ;Old15_j:   ; 07/08/2023
                                ;           jmp     far [cs:Old15] ; 16/10/2022
                                ;           ;jmp    cs:Old15

                                ; =============== S U B   R O U T I N E =======================================

                                ;---------------------------------------------------------------------------
                                ;
                                ; procedure : EraseVDiskHead
                                ;
                                ;           Erases the VDisk Header present in the 1MB boundary
                                ;
                                ;---------------------------------------------------------------------------

                                EraseVDiskHead:     ; proc near
                                        ; 07/08/2023 - Retro DOS v4.2 IO.SYS (optimization)
                                        ; PCDOS 7.1 IBMBIO.COM - BIOSDATA:07C1h
                                        ;push   ax
000007C1 51                             push    cx
000007C2 57                             push    di
000007C3 06                             push    es
000007C4 E863FF                         call    EnsureA20On
                                        ;mov    ax,0FFFFh       ; HMA seg
                                        ;mov    es,ax
                                        ; 03/10/2023 - Retro DOS v5.0
000007C7 6AFF                           push    0FFFFh
000007C9 07                             pop     es
000007CA BF1000                         mov     di,10h          ; point to VDISK header
                                        ; 07/08/2023
                                        ;mov    cx,10h          ; size of vdisk       header
000007CD 89F9                           mov     cx,di ; 16
                                        ; 03/10/2023
000007CF 31C0                           xor     ax,ax
                                        ;inc    ax ; ax = 0
000007D1 F3AB                           rep stosw               ; clear it
000007D3 07                             pop     es
000007D4 5F                             pop     di
000007D5 59                             pop     cx
                                        ;pop    ax ; 07/08/2023
000007D6 C3                             retn

                                ; ---------------------------------------------------------------------------

                                ; 03/10/2023 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMBIO.COM)
                                ; 17/10/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 IO.SYS)

                                ; 09/12/2022
                                ;SYSINITSEG equ 46Dh  ; SYSINIT segment
                                ;DOSLOADSEG equ 83Fh  ; MSDOS.SYS (kernel) loading segment
                                ; (followings are in sysinit segment)
                                ;FTryToMovDOSHi    equ 0A84h ; (procedure in SYSINIT segment)
                                FTRYTOMOVDOSHI     equ FTryToMovDOSHi ; SYSINIT section
                                ;DEVICELIST equ 273h
                                DEVICELIST  equ DEVICE_LIST       ; SYSINIT section
                                ;MEMORYSIZE equ 292h
                                MEMORYSIZE  equ MEMORY_SIZE       ; SYSINIT section
                                ;DEFAULTDRIVE      equ 296h
                                DEFAULTDRIVE       equ DEFAULT_DRIVE ; SYSINIT section
                                ;;currentdoslocation equ 271h
                                ;CURRENTDOSLOCATION equ 271h
                                CURRENTDOSLOCATION equ CURRENT_DOS_LOCATION  ; SYSINIT section
                                ;SYSINITSTART      equ 267h
                                SYSINITSTART       equ SYSINIT  ; SYSINIT section
                                ; 18/10/2022
                                ;toomanydrivesflag equ 3FFh
                                TOOMANYDRIVESFLAG equ toomanydrivesflag ; SYSINIT section

                                ; ---------------------------------------------------------------------------

                                ; 03/10/2023 - Retro DOS v5.0 IBMBIO.COM (Modified PCDOS 7.1)
                                ; PCDOS 7.1 IBMBIO.COM - BIOSDATA:07D7h

                                %if 1

000007D7 FFFF                   FreeHMAPtr: dw 0FFFFh
                                ;MoveDOSIntoHMA: dd 46D0A84h            ; FTryToMovDOSHi
                                                                        ; (procedure in       SYSINIT segment)
                                ; 17/10/2022
000007D9 [E20B]                 MoveDOSIntoHMA:    dw FTRYTOMOVDOSHI     ; 09/12/2022
000007DB D904                           dw SYSINITSEG           ; 08/08/2023
                                                                ; 0544h for PCDOS 7.1 IBMBIO.COM
                                                                ; 0473h for MSDOS 6.21 IO.SYS
                                ;SR;
                                ; A communication block has been setup between the DOS and the BIOS. All
                                ;the data starting from SysinitPresent will be part of the data block.
                                ;Right now, this is the only data being communicated. It can be expanded
                                ;later to add more stuff

000007DD 00                     SysinitPresent:    db 0

                                %endif

                                ; ---------------------------------------------------------------------------

                                ;****************************************************************************
                                ;*                                                                          *
                                ;*  the int2f handler chains up to Bios_Code through here.                  *
                                ;*    it returns through one of the three functions that follow.            *
                                ;*    notice that we'll assume we're being entered from DOS, so             *
                                ;*    that we're guaranteed to be A20 enabled if needed                     *
                                ;*                                                                          *
                                ;****************************************************************************

                                ; 07/08/2023 - Retro DOS v4.2 IO.SYS (optimization)
                                %if 0       ; 20/09/2023
                                int_2f:
                                        jmp     far [cs:bcode_i2f] ; 16/10/2022
                                        ;jmp    dword ptr cs:bcode_i2f ; far [cs:bcode_i2f]

                                ; ---------------------------------------------------------------------------

                                ; re-enter here to transition out of hma mode and jmp to dsk_entry
                                ; note:  is it really necessary to transiton out and then back in?
                                ;    It's not as if this is a really speed critical function.
                                ;    might as well do whatever's most compact.

                                i2f_dskentry:
                                        jmp     dsk_entry
```

```
                                ; ----------------------------------------------------------------------------

                                ;****************************************************************************
                                ;*                                                                          *
                                ;*  re_init - called back by sysinit after a bunch of stuff                 *
                                ;*          is done. presently does nothing. affects no                     *
                                ;*          registers!                                                      *
                                ;*                                                                          *
                                ;****************************************************************************

                                ; 09/12/2022
                                ; re_init_:
                                re_init:                        ; called back by sysinit after
                                        retf                    ; a bunch of stuff is done.
                                                                ; presently does nothing
                                %endif

                                ; ----------------------------------------------------------------------------

                                ;SR; WIN386 support

                                ; WIN386 instance data structure
                                ;
                                ; Here is a Win386 startup info structure which we set up and to which
                                ; we return a pointer when Win386 initializes.

000007DE 0300                   Win386_SI:  db 3,0              ; SI_Version
                                                                ; Startup Info for Win386
000007E0 00000000               SI_Next:    dd 0                ; pointer to next info structure
000007E4 00000000                           dd 0                ; a field we don't need
000007E8 00000000                           dd 0                ; another field       we don't need
000007EC [F007]                 SI_Instance:        dw Instance_Table
000007EE 7000                               dw 70h  ; Bios_Data    ; far pointer to instance table

                                ; This table gives Win386 the instance data in the BIOS and ROM-BIOS data
                                ; areas. Note that the address and size of the hardware stacks must
                                ; be calculated and inserted at boot time.

000007F0 00005000               Instance_Table:     dw 0,50h            ; print screen status...
000007F4 0200                               dw 2                ; ... 2 bytes
000007F6 0E005000                           dw 0Eh,50h          ; ROM Basic data...
000007FA 1400                               dw 14h              ; ... 14H bytes
000007FC [0C00]                             dw altah            ; a con device buffer...
000007FE 7000                               dw 70h              ; Bios_Data segment
00000800 0100                               dw 1                ; ... 1 byte

                                NextStack:

                                ; NOTE:  If stacks are disabled by STACKS=0,0, the following
                                ;   instance items WILL NOT be filled in by SYSINIT.
                                ;   That's just fine as long as these are the last items
                                ;   in the instance list since the first item is initialized
                                ;   to 0000 at load time.

00000802 00000000                           dw 0,0              ; pointer to next stack      to be used...
00000806 0200                               dw 2                ; ... 2 bytes
00000808 00000000               IT_StackLoc:        dd 0                ; location of hardware stacks
0000080C 0000                   IT_StackSize:       dw 0                ; size of hardware stacks
0000080E 00000000                           dd 0                ; terminate the       instance table

                                                                ;SR;
00000812 00                     IsWin386:   db 0                ; Flag to indicate whether
                                                                ; Win386 is running or not
                                ;----------------------------------------------------------------------------

                                ; 09/12/2023 - Retro DOS v5.0 IBMBIO.COM (Modified PCDOS 7.1)
                                ; PCDOS 7.1 IBMBIO.COM - BIOSDATA:0813h

                                ;This routine was originally in BIOS_CODE but this causes a lot of problems
                                ;when we call it including checking of A20. The code being only about
                                ;30 bytes, we might as well put it in BIOS_DATA

                                V86_Crit_SetFocus:
00000813 57                             push    di
00000814 06                             push    es
00000815 53                             push    bx
00000816 50                             push    ax
00000817 31FF                           xor     di,di
00000819 8EC7                           mov     es,di
0000081B BB1500                         mov     bx,15h          ; Device ID of DOSMGR device
0000081E B88416                         mov     ax,1684h        ; Get API entry       point
00000821 CD2F                           int     2Fh             ; - Multiplex -       MS WINDOWS - GET DEVICE       API ENTRY POINT
                                                                ; BX = virtual device (VxD) ID,      ES:DI = 0000h:0000h
                                                                ; Return: ES:DI       -> VxD API entry point,       or 0:0 if the VxD does not
support an API
00000823 8CC0                           mov     ax, es
00000825 09F8                           or      ax, di
00000827 740A                           jz      short Skip      ; Here, es:di is address of API       routine.
                                                                ; Set up stack frame to       simulate a call.
00000829 0E                             push    cs
                                        ;;mov   ax,offset Skip
                                        ;mov    ax,Skip
                                        ;push   ax
                                        ; 03/10/2023 - Retro DOS v5.0
0000082A 68[3308]                       push    Skip
0000082D 06                             push    es
0000082E 57                             push    di              ; API far call address
0000082F B80100                         mov     ax,1            ; SetFocus function number
00000832 CB                             retf                    ; do the call
                                ;----------------------------------------------------------------------------

                                Skip:
00000833 58                             pop     ax
00000834 5B                             pop     bx
00000835 07                             pop     es
00000836 5F                             pop     di
00000837 CB                             retf

                                ;End WIN386 support

                                ; ----------------------------------------------------------------------------

                                ; 03/10/2023 - Retro DOS v5.0 IBMBIO.COM (Modified PCDOS 7.1)
                                %if 0

                                FreeHMAPtr: dw 0FFFFh
                                ;MoveDOSIntoHMA: dd 46D0A84h             ; FTryToMovDOSHi
                                                                ; (procedure in       SYSINIT segment)
                                ; 17/10/2022
                                MoveDOSIntoHMA:     dw FTRYTOMOVDOSHI       ; 09/12/2022
                                        dw SYSINITSEG           ; 08/08/2023
                                                                ; 0544h for PCDOS 7.1 IBMBIO.COM
                                                                ; 0473h for MSDOS 6.21 IO.SYS
                                ;SR;
```

```
                                ; A communication block has been setup between the DOS and the BIOS. All
                                ;the data starting from SysinitPresent will be part of the data block.
                                ;Right now, this is the only data being communicated. It can be expanded
                                ;later to add more stuff

                                SysinitPresent:    db 0

                                endfloppy:  db 0, 0

                                %endif

                                    ; 03/10/2023 - Retro DOS v5.0 IBMBIO.COM

                                    endfloppy:
00000838 00                             db 0

                                    ; 03/10/2023

                                numxdiv     equ ($-BData_start)
                                numxmod     equ (numxdiv % 16)

                                %if (numxmod>0) & (numxmod<16)
00000839 00<rep 7h>                     times (16-numxmod) db 0
                                %endif

                                ; ----------------------------------------------------------------------------

                                ; Bios_Data ends

                                ; Possibly disposable BIOS data
                                ; This data follows the   regular BIOS data,
                                ; and is part of the same group.

                                ; 03/10/2023 - Retro DOS v5.0 IBMBIO.COM
                                ;nul_vid:   db 'NO NAME    ',0    ; null volume id
                                ;tmp_vid:   db 'NO NAME    ',0    ; vid scratch buffer

                                ; 03/10/2023
00000840 4E4F204E414D452020-    tmp_vid:    db 'NO NAME    '
00000849 2020

0000084B 80                     harddrv:    db 80h

                                end96tpi:

                                ; 03/10/2023 - Retro DOS v5.0 IBMBIO.COM (Modified PCDOS 7.1)
                                ; PCDOS 7.1 IBMBIO.COM - BIOSDATA:084Ch ('bdss:' address)

                                ;;**************************************************************************
                                ;;memory allocation for bdss
                                ;;**************************************************************************
                                ;
                                ;;max_mini_dsk_num equ 23 ; max # of mini disk ibmbio can support
                                ;
                                ;;bdss      BDS_STRUC (2+max_mini_dsk_num) dup (<>)     ; currently max. 25
                                ;
                                ;bdss:      times BDS.size*(2+max_mini_dsk_num) db 0


                                ; 09/12/2023
                                %if 1
                                ; Retro DOS v5.0 (PCDOS 7.1) IBMBIO.COM BDS structure (FAT32 adaptation)

0000084C FFFF                   bdss:               dw 0FFFFh               ; ...
                                                                    ; max_mini_dsk_num equ 23
                                                                    ; BDS_STRUC (2+max_mini_dsk_num) dup (<>)
                                                                    ; currently max. 25
                                                                    ; (MSDOS 6 BDS structure size = 100 bytes)
                                                                    ; (PCDOS 7.1 BDS structure size = 150 bytes)
                                                                    ; BDS.link
0000084E 0000                               dw 0
00000850 50                                 db 80                   ; BDS.drivenum
00000851 03                                 db 3                    ; BDS.drivelet
00000852 0002                               dw 512                  ; BDS.BPB (BDS offset 6)
                                                                    ; 53 bytes BPB for FAT32 fs
                                                                    ; 25 bytes BPB for FAT16 and FAT12 fs
                                                                    ; .bytespersec
00000854 01                                 db 1                    ; .secperclus
00000855 0100                               dw 1                    ; .resectors
00000857 02                                 db 2                    ; .fats
00000858 1000                               dw 16                   ; .direntries
0000085A 0000                               dw 0                    ; .totalsec16
0000085C F8                                 db 0F8h                 ; .media
0000085D 0100                               dw 1                    ; .fatsecs16
0000085F 0000                               dw 0                    ; .secpertrack
00000861 0000                               dw 0                    ; .heads
00000863 00000000                           dd 0                    ; .hiddensectors
00000867 00000000                           dd 0                    ; .totalsecs32
                                                                    ; (End of FAT12/FAT16 BPB)
                                                                    ;
                                                                    ; FAT32 extensions to BDS
0000086B 00000000                           dd 0                    ; .fatsecs32 ; BPB_FATSz32 (BDS offset 31)
0000086F 0000                               dw 0                    ; .extflags ; BPB_ExtFlags
00000871 0000                               dw 0                    ; .fsver ; BPB_FSVer
00000873 00000000                           dd 0                    ; .rootdirclust ; BPB_RootClus (BDS offset 39)
00000877 FFFF                               dw 0FFFFh               ; .fsinfo ; BPB_FSInfo ; initialized to -1
00000879 FFFF                               dw 0FFFFh               ; .bkbootsec ; BPB_BkBootSec ; initialized to -1
0000087B 00<rep Ch>                         times 12 db 0           ; .reserved ; BPB_Reserved (12 zero bytes)
00000887 00                                 db 0                    ; BDS.fatsiz (BDS offset 59)
00000888 0000                               dw 0                    ; BDS.opcnt
0000088A 03                                 db 3
0000088B 2000                               dw 20h                  ; BDS.flags (BDS offset 63)
0000088D 2800                               dw 40
0000088F 00<rep 25h>                        times 37 db 0
000008B4 FFFFFFFF                           dd 0FFFFFFFFh
000008B8 00<rep Ch>                         times 12 db 0
000008C4 FF                                 db -1                   ; BDS.track (BDS offset 120)
000008C5 0100                               dw 1                    ; BDS.tim_lo ; BDS.bdsm_ismini
000008C7 0000                               dw 0                    ; BDS.tim_hi
000008C9 4E4F204E414D452020-                db 'NO NAME    ',0     ; BDS.volid
000008D2 202000
000008D5 00000000                           dd 0                    ; BDS.vol_serial (BDS offset 137)
000008D9 464154313220202000                 db 'FAT12   ',0                 ; BDS.filesys_id
000008E2 FFFF                   bds_1:              dw 0FFFFh
000008E4 000050030002010100-                db 0, 0, 50h, 3, 0, 2, 1, 1, 0, 2, 10h, 0, 0, 0, 0F8h
000008ED 0210000000F8
000008F3 010000000000000000-                db 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
000008FC 000000000000000000
00000905 00000000000000000FF-               db 0, 0, 0, 0, 0, 0, 0, 0, 0FFh, 0FFh, 0FFh, 0FFh, 0, 0
0000090E FFFFFF0000
00000913 000000000000000000-                db 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 3, 20h, 0, 28h
0000091C 0000000003200028
00000924 000000000000000000-                db 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
0000092D 000000000000000000
```

```
2836 00000936 000000000000000000-               db 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
2836 0000093F 000000000000000000
2837 00000948 0000FFFFFFFF000000-               db 0, 0, 0FFh, 0FFh, 0FFh, 0FFh, 0, 0, 0, 0, 0, 0, 0, 0
2837 00000951 0000000000
2838 00000956 00000000FF01000000-               db 0, 0, 0, 0, 0FFh, 1, 0, 0, 0, 4Eh, 4Fh, 20h, 4Eh, 41h
2838 0000095F 4E4F204E41
2839 00000964 4D4520202020000000-               db 4Dh, 45h, 20h, 20h, 20h, 20h, 0, 0, 0, 0, 0, 46h, 41h
2839 0000096D 00004641
2840 00000971 54313220202000                    db 54h, 31h, 32h, 20h, 20h, 20h, 0
2841 00000978 FFFF                    bds_2:    dw 0FFFFh
2842 0000097A 000050030002010100-               db 0, 0, 50h, 3, 0, 2, 1, 1, 0, 2, 10h, 0, 0, 0, 0F8h
2842 00000983 0210000000F8
2843 00000989 010000000000000000-               db 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
2843 00000992 000000000000000000
2844 0000099B 0000000000000000FF-               db 0, 0, 0, 0, 0, 0, 0, 0, 0FFh, 0FFh, 0FFh, 0FFh, 0, 0
2844 000009A4 FFFFFF0000
2845 000009A9 000000000000000000-               db 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 3, 20h, 0, 28h
2845 000009B2 0000000003200028
2846 000009BA 000000000000000000-               db 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
2846 000009C3 000000000000000000
2847 000009CC 000000000000000000-               db 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
2847 000009D5 000000000000000000
2848 000009DE 0000FFFFFFFF000000-               db 0, 0, 0FFh, 0FFh, 0FFh, 0FFh, 0, 0, 0, 0, 0, 0, 0, 0
2848 000009E7 0000000000
2849 000009EC 00000000FF01000000-               db 0, 0, 0, 0, 0FFh, 1, 0, 0, 0, 4Eh, 4Fh, 20h, 4Eh, 41h
2849 000009F5 4E4F204E41
2850 000009FA 4D4520202020000000-               db 4Dh, 45h, 20h, 20h, 20h, 20h, 0, 0, 0, 0, 0, 46h, 41h
2850 00000A03 00004641
2851 00000A07 54313220202000                    db 54h, 31h, 32h, 20h, 20h, 20h, 0
2852 00000A0E FFFF                    bds_3:    dw 0FFFFh
2853 00000A10 000050030002010100-               db 0, 0, 50h, 3, 0, 2, 1, 1, 0, 2, 10h, 0, 0, 0, 0F8h
2853 00000A19 0210000000F8
2854 00000A1F 010000000000000000-               db 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
2854 00000A28 000000000000000000
2855 00000A31 0000000000000000FF-               db 0, 0, 0, 0, 0, 0, 0, 0, 0FFh, 0FFh, 0FFh, 0FFh, 0, 0
2855 00000A3A FFFFFF0000
2856 00000A3F 000000000000000000-               db 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 3, 20h, 0, 28h
2856 00000A48 0000000003200028
2857 00000A50 000000000000000000-               db 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
2857 00000A59 000000000000000000
2858 00000A62 000000000000000000-               db 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
2858 00000A6B 000000000000000000
2859 00000A74 0000FFFFFFFF000000-               db 0, 0, 0FFh, 0FFh, 0FFh, 0FFh, 0, 0, 0, 0, 0, 0, 0, 0
2859 00000A7D 0000000000
2860 00000A82 00000000FF01000000-               db 0, 0, 0, 0, 0FFh, 1, 0, 0, 0, 4Eh, 4Fh, 20h, 4Eh, 41h
2860 00000A8B 4E4F204E41
2861 00000A90 4D4520202020000000-               db 4Dh, 45h, 20h, 20h, 20h, 20h, 0, 0, 0, 0, 0, 46h, 41h
2861 00000A99 00004641
2862 00000A9D 54313220202000                    db 54h, 31h, 32h, 20h, 20h, 20h, 0
2863 00000AA4 FFFF                    bds_4:    dw 0FFFFh
2864 00000AA6 000050030002010100-               db 0, 0, 50h, 3, 0, 2, 1, 1, 0, 2, 10h, 0, 0, 0, 0F8h
2864 00000AAF 0210000000F8
2865 00000AB5 010000000000000000-               db 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
2865 00000ABE 000000000000000000
2866 00000AC7 0000000000000000FF-               db 0, 0, 0, 0, 0, 0, 0, 0, 0FFh, 0FFh, 0FFh, 0FFh, 0, 0
2866 00000AD0 FFFFFF0000
2867 00000AD5 000000000000000000-               db 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 3, 20h, 0, 28h
2867 00000ADE 0000000003200028
2868 00000AE6 000000000000000000-               db 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
2868 00000AEF 000000000000000000
2869 00000AF8 000000000000000000-               db 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
2869 00000B01 000000000000000000
2870 00000B0A 0000FFFFFFFF000000-               db 0, 0, 0FFh, 0FFh, 0FFh, 0FFh, 0, 0, 0, 0, 0, 0, 0, 0
2870 00000B13 0000000000
2871 00000B18 00000000FF01000000-               db 0, 0, 0, 0, 0FFh, 1, 0, 0, 0, 4Eh, 4Fh, 20h, 4Eh, 41h
2871 00000B21 4E4F204E41
2872 00000B26 4D4520202020000000-               db 4Dh, 45h, 20h, 20h, 20h, 20h, 0, 0, 0, 0, 0, 46h, 41h
2872 00000B2F 00004641
2873 00000B33 54313220202000                    db 54h, 31h, 32h, 20h, 20h, 20h, 0
2874 00000B3A FFFF                              dw 0FFFFh
2875 00000B3C 000050030002010100-               db 0, 0, 50h, 3, 0, 2, 1, 1, 0, 2, 10h, 0, 0, 0, 0F8h
2875 00000B45 0210000000F8
2876 00000B4B 010000000000000000-               db 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
2876 00000B54 000000000000000000
2877 00000B5D 0000000000000000FF-               db 0, 0, 0, 0, 0, 0, 0, 0, 0FFh, 0FFh, 0FFh, 0FFh, 0, 0
2877 00000B66 FFFFFF0000
2878 00000B6B 000000000000000000-               db 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 3, 20h, 0, 28h
2878 00000B74 0000000003200028
2879 00000B7C 000000000000000000-               db 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
2879 00000B85 000000000000000000
2880 00000B8E 000000000000000000-               db 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
2880 00000B97 000000000000000000
2881 00000BA0 0000FFFFFFFF000000-               db 0, 0, 0FFh, 0FFh, 0FFh, 0FFh, 0, 0, 0, 0, 0, 0, 0, 0
2881 00000BA9 0000000000
2882 00000BAE 00000000FF01000000-               db 0, 0, 0, 0, 0FFh, 1, 0, 0, 0, 4Eh, 4Fh, 20h, 4Eh, 41h
2882 00000BB7 4E4F204E41
2883 00000BBC 4D4520202020000000-               db 4Dh, 45h, 20h, 20h, 20h, 20h, 0, 0, 0, 0, 0, 46h, 41h
2883 00000BC5 00004641
2884 00000BC9 54313220202000                    db 54h, 31h, 32h, 20h, 20h, 20h, 0
2885 00000BD0 FFFF                              dw 0FFFFh
2886 00000BD2 000050030002010100-               db 0, 0, 50h, 3, 0, 2, 1, 1, 0, 2, 10h, 0, 0, 0, 0F8h
2886 00000BDB 0210000000F8
2887 00000BE1 010000000000000000-               db 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
2887 00000BEA 000000000000000000
2888 00000BF3 0000000000000000FF-               db 0, 0, 0, 0, 0, 0, 0, 0, 0FFh, 0FFh, 0FFh, 0FFh, 0, 0
2888 00000BFC FFFFFF0000
2889 00000C01 000000000000000000-               db 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 3, 20h, 0, 28h
2889 00000C0A 0000000003200028
2890 00000C12 000000000000000000-               db 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
2890 00000C1B 000000000000000000
2891 00000C24 000000000000000000-               db 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
2891 00000C2D 000000000000000000
2892 00000C36 0000FFFFFFFF000000-               db 0, 0, 0FFh, 0FFh, 0FFh, 0FFh, 0, 0, 0, 0, 0, 0, 0, 0
2892 00000C3F 0000000000
2893 00000C44 00000000FF01000000-               db 0, 0, 0, 0, 0FFh, 1, 0, 0, 0, 4Eh, 4Fh, 20h, 4Eh, 41h
2893 00000C4D 4E4F204E41
2894 00000C52 4D4520202020000000-               db 4Dh, 45h, 20h, 20h, 20h, 20h, 0, 0, 0, 0, 0, 46h, 41h
2894 00000C5B 00004641
2895 00000C5F 54313220202000                    db 54h, 31h, 32h, 20h, 20h, 20h, 0
2896 00000C66 FFFF                              dw 0FFFFh
2897 00000C68 000050030002010100-               db 0, 0, 50h, 3, 0, 2, 1, 1, 0, 2, 10h, 0, 0, 0, 0F8h
2897 00000C71 0210000000F8
2898 00000C77 010000000000000000-               db 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
2898 00000C80 000000000000000000
2899 00000C89 0000000000000000FF-               db 0, 0, 0, 0, 0, 0, 0, 0, 0FFh, 0FFh, 0FFh, 0FFh, 0, 0
2899 00000C92 FFFFFF0000
2900 00000C97 000000000000000000-               db 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 3, 20h, 0, 28h
2900 00000CA0 0000000003200028
2901 00000CA8 000000000000000000-               db 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
2901 00000CB1 000000000000000000
2902 00000CBA 000000000000000000-               db 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
2902 00000CC3 000000000000000000
2903 00000CCC 0000FFFFFFFF000000-               db 0, 0, 0FFh, 0FFh, 0FFh, 0FFh, 0, 0, 0, 0, 0, 0, 0, 0
2903 00000CD5 0000000000
```

```
2904 00000CDA 00000000FF01000000-               db 0, 0, 0, 0, 0FFh, 1, 0, 0, 0, 4Eh, 4Fh, 20h, 4Eh, 41h
2904 00000CE3 4E4F204E41
2905 00000CE8 4D4520202020000000-               db 4Dh, 45h, 20h, 20h, 20h, 20h, 0, 0, 0, 0, 0, 46h, 41h
2905 00000CF1 00004641
2906 00000CF5 54313220202000                    db 54h, 31h, 32h, 20h, 20h, 20h, 0
2907 00000CFC FFFF                              dw 0FFFFh
2908 00000CFE 000050030002010100-               db 0, 0, 50h, 3, 0, 2, 1, 1, 0, 2, 10h, 0, 0, 0, 0F8h
2908 00000D07 0210000000F8
2909 00000D0D 010000000000000000-               db 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
2909 00000D16 0000000000000000
2910 00000D1F 0000000000000000FF-               db 0, 0, 0, 0, 0, 0, 0, 0, 0FFh, 0FFh, 0FFh, 0FFh, 0, 0
2910 00000D28 FFFFFF0000
2911 00000D2D 000000000000000000-               db 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 3, 20h, 0, 28h
2911 00000D36 0000000003200028
2912 00000D3E 000000000000000000-               db 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
2912 00000D47 0000000000000000
2913 00000D50 000000000000000000-               db 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
2913 00000D59 0000000000000000
2914 00000D62 0000FFFFFFFF000000-               db 0, 0, 0FFh, 0FFh, 0FFh, 0FFh, 0, 0, 0, 0, 0, 0, 0, 0
2914 00000D6B 0000000000
2915 00000D70 00000000FF01000000-               db 0, 0, 0, 0, 0FFh, 1, 0, 0, 0, 4Eh, 4Fh, 20h, 4Eh, 41h
2915 00000D79 4E4F204E41
2916 00000D7E 4D4520202020000000-               db 4Dh, 45h, 20h, 20h, 20h, 20h, 0, 0, 0, 0, 0, 46h, 41h
2916 00000D87 00004641
2917 00000D8B 54313220202000                    db 54h, 31h, 32h, 20h, 20h, 20h, 0
2918 00000D92 FFFF                              dw 0FFFFh
2919 00000D94 000050030002010100-               db 0, 0, 50h, 3, 0, 2, 1, 1, 0, 2, 10h, 0, 0, 0, 0F8h
2919 00000D9D 0210000000F8
2920 00000DA3 010000000000000000-               db 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
2920 00000DAC 0000000000000000
2921 00000DB5 0000000000000000FF-               db 0, 0, 0, 0, 0, 0, 0, 0, 0FFh, 0FFh, 0FFh, 0FFh, 0, 0
2921 00000DBE FFFFFF0000
2922 00000DC3 000000000000000000-               db 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 3, 20h, 0, 28h
2922 00000DCC 0000000003200028
2923 00000DD4 000000000000000000-               db 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
2923 00000DDD 0000000000000000
2924 00000DE6 000000000000000000-               db 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
2924 00000DEF 0000000000000000
2925 00000DF8 0000FFFFFFFF000000-               db 0, 0, 0FFh, 0FFh, 0FFh, 0FFh, 0, 0, 0, 0, 0, 0, 0, 0
2925 00000E01 0000000000
2926 00000E06 00000000FF01000000-               db 0, 0, 0, 0, 0FFh, 1, 0, 0, 0, 4Eh, 4Fh, 20h, 4Eh, 41h
2926 00000E0F 4E4F204E41
2927 00000E14 4D4520202020000000-               db 4Dh, 45h, 20h, 20h, 20h, 20h, 0, 0, 0, 0, 0, 46h, 41h
2927 00000E1D 00004641
2928 00000E21 54313220202000                    db 54h, 31h, 32h, 20h, 20h, 20h, 0
2929 00000E28 FFFF                              dw 0FFFFh
2930 00000E2A 000050030002010100-               db 0, 0, 50h, 3, 0, 2, 1, 1, 0, 2, 10h, 0, 0, 0, 0F8h
2930 00000E33 0210000000F8
2931 00000E39 010000000000000000-               db 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
2931 00000E42 0000000000000000
2932 00000E4B 0000000000000000FF-               db 0, 0, 0, 0, 0, 0, 0, 0, 0FFh, 0FFh, 0FFh, 0FFh, 0, 0
2932 00000E54 FFFFFF0000
2933 00000E59 000000000000000000-               db 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 3, 20h, 0, 28h
2933 00000E62 0000000003200028
2934 00000E6A 000000000000000000-               db 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
2934 00000E73 0000000000000000
2935 00000E7C 000000000000000000-               db 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
2935 00000E85 0000000000000000
2936 00000E8E 0000FFFFFFFF000000-               db 0, 0, 0FFh, 0FFh, 0FFh, 0FFh, 0, 0, 0, 0, 0, 0, 0, 0
2936 00000E97 0000000000
2937 00000E9C 00000000FF01000000-               db 0, 0, 0, 0, 0FFh, 1, 0, 0, 0, 4Eh, 4Fh, 20h, 4Eh, 41h
2937 00000EA5 4E4F204E41
2938 00000EAA 4D4520202020000000-               db 4Dh, 45h, 20h, 20h, 20h, 20h, 0, 0, 0, 0, 0, 46h, 41h
2938 00000EB3 00004641
2939 00000EB7 54313220202000                    db 54h, 31h, 32h, 20h, 20h, 20h, 0
2940 00000EBE FFFF                              dw 0FFFFh
2941 00000EC0 000050030002010100-               db 0, 0, 50h, 3, 0, 2, 1, 1, 0, 2, 10h, 0, 0, 0, 0F8h
2941 00000EC9 0210000000F8
2942 00000ECF 010000000000000000-               db 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
2942 00000ED8 0000000000000000
2943 00000EE1 0000000000000000FF-               db 0, 0, 0, 0, 0, 0, 0, 0, 0FFh, 0FFh, 0FFh, 0FFh, 0, 0
2943 00000EEA FFFFFF0000
2944 00000EEF 000000000000000000-               db 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 3, 20h, 0, 28h
2944 00000EF8 0000000003200028
2945 00000F00 000000000000000000-               db 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
2945 00000F09 0000000000000000
2946 00000F12 000000000000000000-               db 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
2946 00000F1B 0000000000000000
2947 00000F24 0000FFFFFFFF000000-               db 0, 0, 0FFh, 0FFh, 0FFh, 0FFh, 0, 0, 0, 0, 0, 0, 0, 0
2947 00000F2D 0000000000
2948 00000F32 00000000FF01000000-               db 0, 0, 0, 0, 0FFh, 1, 0, 0, 0, 4Eh, 4Fh, 20h, 4Eh, 41h
2948 00000F3B 4E4F204E41
2949 00000F40 4D4520202020000000-               db 4Dh, 45h, 20h, 20h, 20h, 20h, 0, 0, 0, 0, 0, 46h, 41h
2949 00000F49 00004641
2950 00000F4D 54313220202000                    db 54h, 31h, 32h, 20h, 20h, 20h, 0
2951 00000F54 FFFF                              dw 0FFFFh
2952 00000F56 000050030002010100-               db 0, 0, 50h, 3, 0, 2, 1, 1, 0, 2, 10h, 0, 0, 0, 0F8h
2952 00000F5F 0210000000F8
2953 00000F65 010000000000000000-               db 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
2953 00000F6E 0000000000000000
2954 00000F77 0000000000000000FF-               db 0, 0, 0, 0, 0, 0, 0, 0, 0FFh, 0FFh, 0FFh, 0FFh, 0, 0
2954 00000F80 FFFFFF0000
2955 00000F85 000000000000000000-               db 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 3, 20h, 0, 28h
2955 00000F8E 0000000003200028
2956 00000F96 000000000000000000-               db 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
2956 00000F9F 0000000000000000
2957 00000FA8 000000000000000000-               db 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
2957 00000FB1 0000000000000000
2958 00000FBA 0000FFFFFFFF000000-               db 0, 0, 0FFh, 0FFh, 0FFh, 0FFh, 0, 0, 0, 0, 0, 0, 0, 0
2958 00000FC3 0000000000
2959 00000FC8 00000000FF01000000-               db 0, 0, 0, 0, 0FFh, 1, 0, 0, 0, 4Eh, 4Fh, 20h, 4Eh, 41h
2959 00000FD1 4E4F204E41
2960 00000FD6 4D4520202020000000-               db 4Dh, 45h, 20h, 20h, 20h, 20h, 0, 0, 0, 0, 0, 46h, 41h
2960 00000FDF 00004641
2961 00000FE3 54313220202000                    db 54h, 31h, 32h, 20h, 20h, 20h, 0
2962 00000FEA FFFF                              dw 0FFFFh
2963 00000FEC 000050030002010100-               db 0, 0, 50h, 3, 0, 2, 1, 1, 0, 2, 10h, 0, 0, 0, 0F8h
2963 00000FF5 0210000000F8
2964 00000FFB 010000000000000000-               db 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
2964 00001004 0000000000000000
2965 0000100D 0000000000000000FF-               db 0, 0, 0, 0, 0, 0, 0, 0, 0FFh, 0FFh, 0FFh, 0FFh, 0, 0
2965 00001016 FFFFFF0000
2966 0000101B 000000000000000000-               db 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 3, 20h, 0, 28h
2966 00001024 0000000003200028
2967 0000102C 000000000000000000-               db 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
2967 00001035 0000000000000000
2968 0000103E 000000000000000000-               db 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
2968 00001047 0000000000000000
2969 00001050 0000FFFFFFFF000000-               db 0, 0, 0FFh, 0FFh, 0FFh, 0FFh, 0, 0, 0, 0, 0, 0, 0, 0
2969 00001059 0000000000
2970 0000105E 00000000FF01000000-               db 0, 0, 0, 0, 0FFh, 1, 0, 0, 0, 4Eh, 4Fh, 20h, 4Eh, 41h
2970 00001067 4E4F204E41
2971 0000106C 4D4520202020000000-               db 4Dh, 45h, 20h, 20h, 20h, 20h, 0, 0, 0, 0, 0, 46h, 41h
2971 00001075 00004641
```

```
2972 00001079 54313220202000                  db 54h, 31h, 32h, 20h, 20h, 20h, 0
2973 00001080 FFFF                            dw 0FFFFh
2974 00001082 000050030002010100-             db 0, 0, 50h, 3, 0, 2, 1, 1, 0, 2, 10h, 0, 0, 0, 0F8h
2974 0000108B 0210000000F8
2975 00001091 010000000000000000-             db 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
2975 0000109A 000000000000000000
2976 000010A3 00000000000000000FF-            db 0, 0, 0, 0, 0, 0, 0, 0, 0FFh, 0FFh, 0FFh, 0FFh, 0, 0
2976 000010AC FFFFFF0000
2977 000010B1 000000000000000000-             db 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 3, 20h, 0, 28h
2977 000010BA 0000000003200028
2978 000010C2 000000000000000000-             db 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
2978 000010CB 000000000000000000
2979 000010D4 000000000000000000-             db 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
2979 000010DD 000000000000000000
2980 000010E6 0000FFFFFFFF000000-             db 0, 0, 0FFh, 0FFh, 0FFh, 0FFh, 0, 0, 0, 0, 0, 0, 0, 0
2980 000010EF 0000000000
2981 000010F4 00000000FF01000000-             db 0, 0, 0, 0, 0FFh, 1, 0, 0, 0, 4Eh, 4Fh, 20h, 4Eh, 41h
2981 000010FD 4E4F204E41
2982 00001102 4D4520202020000000-             db 4Dh, 45h, 20h, 20h, 20h, 20h, 0, 0, 0, 0, 0, 46h, 41h
2982 0000110B 00004641
2983 0000110F 54313220202000                  db 54h, 31h, 32h, 20h, 20h, 20h, 0
2984 00001116 FFFF                            dw 0FFFFh
2985 00001118 000050030002010100-             db 0, 0, 50h, 3, 0, 2, 1, 1, 0, 2, 10h, 0, 0, 0, 0F8h
2985 00001121 0210000000F8
2986 00001127 010000000000000000-             db 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
2986 00001130 000000000000000000
2987 00001139 00000000000000000FF-            db 0, 0, 0, 0, 0, 0, 0, 0, 0FFh, 0FFh, 0FFh, 0FFh, 0, 0
2987 00001142 FFFFFF0000
2988 00001147 000000000000000000-             db 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 3, 20h, 0, 28h
2988 00001150 0000000003200028
2989 00001158 000000000000000000-             db 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
2989 00001161 000000000000000000
2990 0000116A 000000000000000000-             db 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
2990 00001173 000000000000000000
2991 0000117C 0000FFFFFFFF000000-             db 0, 0, 0FFh, 0FFh, 0FFh, 0FFh, 0, 0, 0, 0, 0, 0, 0, 0
2991 00001185 0000000000
2992 0000118A 00000000FF01000000-             db 0, 0, 0, 0, 0FFh, 1, 0, 0, 0, 4Eh, 4Fh, 20h, 4Eh, 41h
2992 00001193 4E4F204E41
2993 00001198 4D4520202020000000-             db 4Dh, 45h, 20h, 20h, 20h, 20h, 0, 0, 0, 0, 0, 46h, 41h
2993 000011A1 00004641
2994 000011A5 54313220202000                  db 54h, 31h, 32h, 20h, 20h, 20h, 0
2995 000011AC FFFF                            dw 0FFFFh
2996 000011AE 000050030002010100-             db 0, 0, 50h, 3, 0, 2, 1, 1, 0, 2, 10h, 0, 0, 0, 0F8h
2996 000011B7 0210000000F8
2997 000011BD 010000000000000000-             db 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
2997 000011C6 000000000000000000
2998 000011CF 00000000000000000FF-            db 0, 0, 0, 0, 0, 0, 0, 0, 0FFh, 0FFh, 0FFh, 0FFh, 0, 0
2998 000011D8 FFFFFF0000
2999 000011DD 000000000000000000-             db 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 3, 20h, 0, 28h
2999 000011E6 0000000003200028
3000 000011EE 000000000000000000-             db 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
3000 000011F7 000000000000000000
3001 00001200 000000000000000000-             db 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
3001 00001209 000000000000000000
3002 00001212 0000FFFFFFFF000000-             db 0, 0, 0FFh, 0FFh, 0FFh, 0FFh, 0, 0, 0, 0, 0, 0, 0, 0
3002 0000121B 0000000000
3003 00001220 00000000FF01000000-             db 0, 0, 0, 0, 0FFh, 1, 0, 0, 0, 4Eh, 4Fh, 20h, 4Eh, 41h
3003 00001229 4E4F204E41
3004 0000122E 4D4520202020000000-             db 4Dh, 45h, 20h, 20h, 20h, 20h, 0, 0, 0, 0, 0, 46h, 41h
3004 00001237 00004641
3005 0000123B 54313220202000                  db 54h, 31h, 32h, 20h, 20h, 20h, 0
3006 00001242 FFFF                            dw 0FFFFh
3007 00001244 000050030002010100-             db 0, 0, 50h, 3, 0, 2, 1, 1, 0, 2, 10h, 0, 0, 0, 0F8h
3007 0000124D 0210000000F8
3008 00001253 010000000000000000-             db 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
3008 0000125C 000000000000000000
3009 00001265 00000000000000000FF-            db 0, 0, 0, 0, 0, 0, 0, 0, 0FFh, 0FFh, 0FFh, 0FFh, 0, 0
3009 0000126E FFFFFF0000
3010 00001273 000000000000000000-             db 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 3, 20h, 0, 28h
3010 0000127C 0000000003200028
3011 00001284 000000000000000000-             db 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
3011 0000128D 000000000000000000
3012 00001296 000000000000000000-             db 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
3012 0000129F 000000000000000000
3013 000012A8 0000FFFFFFFF000000-             db 0, 0, 0FFh, 0FFh, 0FFh, 0FFh, 0, 0, 0, 0, 0, 0, 0, 0
3013 000012B1 0000000000
3014 000012B6 00000000FF01000000-             db 0, 0, 0, 0, 0FFh, 1, 0, 0, 0, 4Eh, 4Fh, 20h, 4Eh, 41h
3014 000012BF 4E4F204E41
3015 000012C4 4D4520202020000000-             db 4Dh, 45h, 20h, 20h, 20h, 20h, 0, 0, 0, 0, 0, 46h, 41h
3015 000012CD 00004641
3016 000012D1 54313220202000                  db 54h, 31h, 32h, 20h, 20h, 20h, 0
3017 000012D8 FFFF                            dw 0FFFFh
3018 000012DA 000050030002010100-             db 0, 0, 50h, 3, 0, 2, 1, 1, 0, 2, 10h, 0, 0, 0, 0F8h
3018 000012E3 0210000000F8
3019 000012E9 010000000000000000-             db 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
3019 000012F2 000000000000000000
3020 000012FB 00000000000000000FF-            db 0, 0, 0, 0, 0, 0, 0, 0, 0FFh, 0FFh, 0FFh, 0FFh, 0, 0
3020 00001304 FFFFFF0000
3021 00001309 000000000000000000-             db 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 3, 20h, 0, 28h
3021 00001312 0000000003200028
3022 0000131A 000000000000000000-             db 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
3022 00001323 000000000000000000
3023 0000132C 000000000000000000-             db 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
3023 00001335 000000000000000000
3024 0000133E 0000FFFFFFFF000000-             db 0, 0, 0FFh, 0FFh, 0FFh, 0FFh, 0, 0, 0, 0, 0, 0, 0, 0
3024 00001347 0000000000
3025 0000134C 00000000FF01000000-             db 0, 0, 0, 0, 0FFh, 1, 0, 0, 0, 4Eh, 4Fh, 20h, 4Eh, 41h
3025 00001355 4E4F204E41
3026 0000135A 4D4520202020000000-             db 4Dh, 45h, 20h, 20h, 20h, 20h, 0, 0, 0, 0, 0, 46h, 41h
3026 00001363 00004641
3027 00001367 54313220202000                  db 54h, 31h, 32h, 20h, 20h, 20h, 0
3028 0000136E FFFF                            dw 0FFFFh
3029 00001370 000050030002010100-             db 0, 0, 50h, 3, 0, 2, 1, 1, 0, 2, 10h, 0, 0, 0, 0F8h
3029 00001379 0210000000F8
3030 0000137F 010000000000000000-             db 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
3030 00001388 000000000000000000
3031 00001391 00000000000000000FF-            db 0, 0, 0, 0, 0, 0, 0, 0, 0FFh, 0FFh, 0FFh, 0FFh, 0, 0
3031 0000139A FFFFFF0000
3032 0000139F 000000000000000000-             db 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 3, 20h, 0, 28h
3032 000013A8 0000000003200028
3033 000013B0 000000000000000000-             db 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
3033 000013B9 000000000000000000
3034 000013C2 000000000000000000-             db 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
3034 000013CB 000000000000000000
3035 000013D4 0000FFFFFFFF000000-             db 0, 0, 0FFh, 0FFh, 0FFh, 0FFh, 0, 0, 0, 0, 0, 0, 0, 0
3035 000013DD 0000000000
3036 000013E2 00000000FF01000000-             db 0, 0, 0, 0, 0FFh, 1, 0, 0, 0, 4Eh, 4Fh, 20h, 4Eh, 41h
3036 000013EB 4E4F204E41
3037 000013F0 4D4520202020000000-             db 4Dh, 45h, 20h, 20h, 20h, 20h, 0, 0, 0, 0, 0, 46h, 41h
3037 000013F9 00004641
3038 000013FD 54313220202000                  db 54h, 31h, 32h, 20h, 20h, 20h, 0
3039 00001404 FFFF                            dw 0FFFFh
3040 00001406 000050030002010100-             db 0, 0, 50h, 3, 0, 2, 1, 1, 0, 2, 10h, 0, 0, 0, 0F8h
3040 0000140F 0210000000F8
```

```
3041 00001415 010000000000000000-               db 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
3041 0000141E 000000000000000000
3042 00001427 0000000000000000FF-               db 0, 0, 0, 0, 0, 0, 0, 0, 0FFh, 0FFh, 0FFh, 0FFh, 0, 0
3042 00001430 FFFFFF0000
3043 00001435 000000000000000000-               db 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 3, 20h, 0, 28h
3043 0000143E 0000000003200028
3044 00001446 000000000000000000-               db 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
3044 0000144F 000000000000000000
3045 00001458 000000000000000000-               db 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
3045 00001461 000000000000000000
3046 0000146A 0000FFFFFFFF000000-               db 0, 0, 0FFh, 0FFh, 0FFh, 0FFh, 0, 0, 0, 0, 0, 0, 0, 0
3046 00001473 0000000000
3047 00001478 00000000FF01000000-               db 0, 0, 0, 0, 0FFh, 1, 0, 0, 0, 4Eh, 4Fh, 20h, 4Eh, 41h
3047 00001481 4E4F204E41
3048 00001486 4D4520202020000000-               db 4Dh, 45h, 20h, 20h, 20h, 20h, 0, 0, 0, 0, 0, 46h, 41h
3048 0000148F 00004641
3049 00001493 54313220202000                    db 54h, 31h, 32h, 20h, 20h, 20h, 0
3050 0000149A FFFF                              dw 0FFFFh
3051 0000149C 000050030002010100-               db 0, 0, 50h, 3, 0, 2, 1, 1, 0, 2, 10h, 0, 0, 0, 0F8h
3051 000014A5 0210000000F8
3052 000014AB 010000000000000000-               db 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
3052 000014B4 000000000000000000
3053 000014BD 0000000000000000FF-               db 0, 0, 0, 0, 0, 0, 0, 0, 0FFh, 0FFh, 0FFh, 0FFh, 0, 0
3053 000014C6 FFFFFF0000
3054 000014CB 000000000000000000-               db 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 3, 20h, 0, 28h
3054 000014D4 0000000003200028
3055 000014DC 000000000000000000-               db 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
3055 000014E5 000000000000000000
3056 000014EE 000000000000000000-               db 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
3056 000014F7 000000000000000000
3057 00001500 0000FFFFFFFF000000-               db 0, 0, 0FFh, 0FFh, 0FFh, 0FFh, 0, 0, 0, 0, 0, 0, 0, 0
3057 00001509 0000000000
3058 0000150E 00000000FF01000000-               db 0, 0, 0, 0, 0FFh, 1, 0, 0, 0, 4Eh, 4Fh, 20h, 4Eh, 41h
3058 00001517 4E4F204E41
3059 0000151C 4D4520202020000000-               db 4Dh, 45h, 20h, 20h, 20h, 20h, 0, 0, 0, 0, 0, 46h, 41h
3059 00001525 00004641
3060 00001529 54313220202000                    db 54h, 31h, 32h, 20h, 20h, 20h, 0
3061 00001530 FFFF                              dw 0FFFFh
3062 00001532 000050030002010100-               db 0, 0, 50h, 3, 0, 2, 1, 1, 0, 2, 10h, 0, 0, 0, 0F8h
3062 0000153B 0210000000F8
3063 00001541 010000000000000000-               db 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
3063 0000154A 000000000000000000
3064 00001553 0000000000000000FF-               db 0, 0, 0, 0, 0, 0, 0, 0, 0FFh, 0FFh, 0FFh, 0FFh, 0, 0
3064 0000155C FFFFFF0000
3065 00001561 000000000000000000-               db 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 3, 20h, 0, 28h
3065 0000156A 0000000003200028
3066 00001572 000000000000000000-               db 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
3066 0000157B 000000000000000000
3067 00001584 000000000000000000-               db 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
3067 0000158D 000000000000000000
3068 00001596 0000FFFFFFFF000000-               db 0, 0, 0FFh, 0FFh, 0FFh, 0FFh, 0, 0, 0, 0, 0, 0, 0, 0
3068 0000159F 0000000000
3069 000015A4 00000000FF01000000-               db 0, 0, 0, 0, 0FFh, 1, 0, 0, 0, 4Eh, 4Fh, 20h, 4Eh, 41h
3069 000015AD 4E4F204E41
3070 000015B2 4D4520202020000000-               db 4Dh, 45h, 20h, 20h, 20h, 20h, 0, 0, 0, 0, 0, 46h, 41h
3070 000015BB 00004641
3071 000015BF 54313220202000                    db 54h, 31h, 32h, 20h, 20h, 20h, 0
3072 000015C6 FFFF                              dw 0FFFFh
3073 000015C8 000050030002010100-               db 0, 0, 50h, 3, 0, 2, 1, 1, 0, 2, 10h, 0, 0, 0, 0F8h
3073 000015D1 0210000000F8
3074 000015D7 010000000000000000-               db 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
3074 000015E0 000000000000000000
3075 000015E9 0000000000000000FF-               db 0, 0, 0, 0, 0, 0, 0, 0, 0FFh, 0FFh, 0FFh, 0FFh, 0, 0
3075 000015F2 FFFFFF0000
3076 000015F7 000000000000000000-               db 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 3, 20h, 0, 28h
3076 00001600 0000000003200028
3077 00001608 000000000000000000-               db 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
3077 00001611 000000000000000000
3078 0000161A 000000000000000000-               db 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
3078 00001623 000000000000000000
3079 0000162C 0000FFFFFFFF000000-               db 0, 0, 0FFh, 0FFh, 0FFh, 0FFh, 0, 0, 0, 0, 0, 0, 0, 0
3079 00001635 0000000000
3080 0000163A 00000000FF01000000-               db 0, 0, 0, 0, 0FFh, 1, 0, 0, 0, 4Eh, 4Fh, 20h, 4Eh, 41h
3080 00001643 4E4F204E41
3081 00001648 4D4520202020000000-               db 4Dh, 45h, 20h, 20h, 20h, 20h, 0, 0, 0, 0, 0, 46h, 41h
3081 00001651 00004641
3082 00001655 54313220202000                    db 54h, 31h, 32h, 20h, 20h, 20h, 0
3083 0000165C FFFF                  bds_24:     dw 0FFFFh
3084 0000165E 000050030002010100-               db 0, 0, 50h, 3, 0, 2, 1, 1, 0, 2, 10h, 0, 0, 0, 0F8h
3084 00001667 0210000000F8
3085 0000166D 010000000000000000-               db 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
3085 00001676 000000000000000000
3086 0000167F 0000000000000000FF-               db 0, 0, 0, 0, 0, 0, 0, 0, 0FFh, 0FFh, 0FFh, 0FFh, 0, 0
3086 00001688 FFFFFF0000
3087 0000168D 000000000000000000-               db 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 3, 20h, 0, 28h
3087 00001696 0000000003200028
3088 0000169E 000000000000000000-               db 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
3088 000016A7 000000000000000000
3089 000016B0 000000000000000000-               db 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
3089 000016B9 000000000000000000
3090 000016C2 0000FFFFFFFF000000-               db 0, 0, 0FFh, 0FFh, 0FFh, 0FFh, 0, 0, 0, 0, 0, 0, 0, 0
3090 000016CB 0000000000
3091 000016D0 00000000FF01000000-               db 0, 0, 0, 0, 0FFh, 1, 0, 0, 0, 4Eh, 4Fh, 20h, 4Eh, 41h
3091 000016D9 4E4F204E41
3092 000016DE 4D4520202020000000-               db 4Dh, 45h, 20h, 20h, 20h, 20h, 0, 0, 0, 0, 0, 46h, 41h
3092 000016E7 00004641
3093 000016EB 54313220202000                    db 54h, 31h, 32h, 20h, 20h, 20h, 0
3094
3095                                    %endif
3096
3097                                    ; 09/12/2023
3098                                    %if 0
3099                                    ; Retro DOS v4.2 (MSDOS 6.22) IO.SYS BDS structure
3100
3101                                    bdss:               dw 0FFFFh
3102                                                db 0, 0, 50h, 3, 0, 2, 1, 1, 0,      2, 10h, 0, 0, 0, 0F8h
3103                                                db 1, 0, 0, 0, 0, 0, 0,       0, 0, 0, 0, 0, 0, 0, 0,       0, 0, 3
3104                                                db 20h, 0, 28h, 0, 0, 0, 0, 0, 0, 0, 0,       0, 0, 0, 0, 0
3105                                                db 0, 0, 0, 0, 0, 0, 0, 0,       0, 0, 0, 0, 0, 0, 0, 0,       0, 0, 0
3106                                                db 0, 0FFh, 1, 0, 0, 0,      4Eh, 4Fh, 20h, 4Eh, 41h, 4Dh, 45h
3107                                                db 20h, 20h, 20h, 20h, 0, 0, 0,       0, 0, 46h, 41h,       54h, 31h
3108                                                db 32h, 20h, 20h, 20h, 0
3109                                                dw 0FFFFh
3110                                                db 0, 0, 50h, 3, 0, 2, 1, 1, 0,      2, 10h, 0, 0, 0, 0F8h
3111                                                db 1, 0, 0, 0, 0, 0, 0,       0, 0, 0, 0, 0, 0, 0, 0,       0, 0, 3
3112                                                db 20h, 0, 28h, 0, 0, 0, 0, 0, 0, 0, 0,       0, 0, 0, 0, 0
3113                                                db 0, 0, 0, 0, 0, 0, 0,       0, 0, 0, 0, 0, 0, 0, 0,       0, 0, 0
3114                                                db 0, 0FFh, 1, 0, 0, 0,      4Eh, 4Fh, 20h, 4Eh, 41h, 4Dh, 45h
3115                                                db 20h, 20h, 20h, 20h, 0, 0, 0,       0, 0, 46h, 41h,       54h, 31h
3116                                                db 32h, 20h, 20h, 20h, 0
3117                                                dw 0FFFFh
3118                                                db 0, 0, 50h, 3, 0, 2, 1, 1, 0,      2, 10h, 0, 0, 0, 0F8h
3119                                                db 1, 0, 0, 0, 0, 0, 0,       0, 0, 0, 0, 0, 0, 0, 0,       0, 0, 3
3120                                                db 20h, 0, 28h, 0, 0, 0, 0, 0, 0, 0, 0,       0, 0, 0, 0, 0
```

```
3121                    db 0, 0, 0, 0, 0, 0, 0,       0, 0, 0, 0, 0, 0, 0, 0,       0, 0, 0
3122                    db 0, 0FFh, 1, 0, 0, 0,       4Eh, 4Fh, 20h, 4Eh, 41h, 4Dh, 45h
3123                    db 20h,20h, 20h, 20h, 0, 0, 0,       0, 0, 46h, 41h,       54h, 31h
3124                    db 32h,20h, 20h, 20h, 0
3125                    dw 0FFFFh
3126                    db 0, 0, 50h, 3, 0, 2, 1, 1, 0,      2, 10h,0, 0, 0, 0F8h
3127                    db 1, 0, 0, 0, 0, 0, 0, 0,       0, 0, 0, 0, 0, 0, 0, 0,       0, 0, 3
3128                    db 20h,0, 28h,0, 0, 0, 0, 0, 0, 0, 0,       0, 0, 0, 0, 0
3129                    db 0, 0, 0, 0, 0, 0, 0, 0,       0, 0, 0, 0, 0, 0, 0, 0,       0, 0, 0
3130                    db 0, 0FFh, 1, 0, 0, 0,       4Eh, 4Fh, 20h, 4Eh, 41h, 4Dh, 45h
3131                    db 20h,20h, 20h, 20h, 0, 0, 0,       0, 0, 46h, 41h,       54h, 31h
3132                    db 32h,20h, 20h, 20h, 0
3133                    dw 0FFFFh
3134                    db 0, 0, 50h, 3, 0, 2, 1, 1, 0,      2, 10h,0, 0, 0, 0F8h
3135                    db 1, 0, 0, 0, 0, 0, 0, 0,       0, 0, 0, 0, 0, 0, 0, 0,       0, 0, 3
3136                    db 20h,0, 28h,0, 0, 0, 0, 0, 0, 0, 0,       0, 0, 0, 0, 0
3137                    db 0, 0, 0, 0, 0, 0, 0, 0,       0, 0, 0, 0, 0, 0, 0, 0,       0, 0, 0
3138                    db 0, 0FFh, 1, 0, 0, 0,       4Eh, 4Fh, 20h, 4Eh, 41h, 4Dh, 45h
3139                    db 20h,20h, 20h, 20h, 0, 0, 0,       0, 0, 46h, 41h,       54h, 31h
3140                    db 32h,20h, 20h, 20h, 0
3141                    dw 0FFFFh
3142                    db 0, 0, 50h, 3, 0, 2, 1, 1, 0,      2, 10h,0, 0, 0, 0F8h
3143                    db 1, 0, 0, 0, 0, 0, 0, 0,       0, 0, 0, 0, 0, 0, 0, 0,       0, 0, 3
3144                    db 20h,0, 28h,0, 0, 0, 0, 0, 0, 0, 0,       0, 0, 0, 0, 0
3145                    db 0, 0, 0, 0, 0, 0, 0, 0,       0, 0, 0, 0, 0, 0, 0, 0,       0, 0, 0
3146                    db 0, 0FFh, 1, 0, 0, 0,       4Eh, 4Fh, 20h, 4Eh, 41h, 4Dh, 45h
3147                    db 20h,20h, 20h, 20h, 0, 0, 0,       0, 0, 46h, 41h,       54h, 31h
3148                    db 32h,20h, 20h, 20h, 0
3149                    dw 0FFFFh
3150                    db 0, 0, 50h, 3, 0, 2, 1, 1, 0,      2, 10h,0, 0, 0, 0F8h
3151                    db 1, 0, 0, 0, 0, 0, 0, 0,       0, 0, 0, 0, 0, 0, 0, 0,       0, 0, 3
3152                    db 20h,0, 28h,0, 0, 0, 0, 0, 0, 0, 0,       0, 0, 0, 0, 0
3153                    db 0, 0, 0, 0, 0, 0, 0, 0,       0, 0, 0, 0, 0, 0, 0, 0,       0, 0, 0
3154                    db 0, 0FFh, 1, 0, 0, 0,       4Eh, 4Fh, 20h, 4Eh, 41h, 4Dh, 45h
3155                    db 20h,20h, 20h, 20h, 0, 0, 0,       0, 0, 46h, 41h,       54h, 31h
3156                    db 32h,20h, 20h, 20h, 0
3157                    dw 0FFFFh
3158                    db 0, 0, 50h, 3, 0, 2, 1, 1, 0,      2, 10h,0, 0, 0, 0F8h
3159                    db 1, 0, 0, 0, 0, 0, 0, 0,       0, 0, 0, 0, 0, 0, 0, 0,       0, 0, 3
3160                    db 20h,0, 28h,0, 0, 0, 0, 0, 0, 0, 0,       0, 0, 0, 0, 0
3161                    db 0, 0, 0, 0, 0, 0, 0, 0,       0, 0, 0, 0, 0, 0, 0, 0,       0, 0, 0
3162                    db 0, 0FFh, 1, 0, 0, 0,       4Eh, 4Fh, 20h, 4Eh, 41h, 4Dh, 45h
3163                    db 20h,20h, 20h, 20h, 0, 0, 0,       0, 0, 46h, 41h,       54h, 31h
3164                    db 32h,20h, 20h, 20h, 0
3165                    dw 0FFFFh
3166                    db 0, 0, 50h, 3, 0, 2, 1, 1, 0,      2, 10h,0, 0, 0, 0F8h
3167                    db 1, 0, 0, 0, 0, 0, 0, 0,       0, 0, 0, 0, 0, 0, 0, 0,       0, 0, 3
3168                    db 20h,0, 28h,0, 0, 0, 0, 0, 0, 0, 0,       0, 0, 0, 0, 0
3169                    db 0, 0, 0, 0, 0, 0, 0, 0,       0, 0, 0, 0, 0, 0, 0, 0,       0, 0, 0
3170                    db 0, 0FFh, 1, 0, 0, 0,       4Eh, 4Fh, 20h, 4Eh, 41h, 4Dh, 45h
3171                    db 20h,20h, 20h, 20h, 0, 0, 0,       0, 0, 46h, 41h,       54h, 31h
3172                    db 32h,20h, 20h, 20h, 0
3173                    dw 0FFFFh
3174                    db 0, 0, 50h, 3, 0, 2, 1, 1, 0,      2, 10h,0, 0, 0, 0F8h
3175                    db 1, 0, 0, 0, 0, 0, 0, 0,       0, 0, 0, 0, 0, 0, 0, 0,       0, 0, 3
3176                    db 20h,0, 28h,0, 0, 0, 0, 0, 0, 0, 0,       0, 0, 0, 0, 0
3177                    db 0, 0, 0, 0, 0, 0, 0, 0,       0, 0, 0, 0, 0, 0, 0, 0,       0, 0, 0
3178                    db 0, 0FFh, 1, 0, 0, 0,       4Eh, 4Fh, 20h, 4Eh, 41h, 4Dh, 45h
3179                    db 20h,20h, 20h, 20h, 0, 0, 0,       0, 0, 46h, 41h,       54h, 31h
3180                    db 32h,20h, 20h, 20h, 0
3181                    dw 0FFFFh
3182                    db 0, 0, 50h, 3, 0, 2, 1, 1, 0,      2, 10h,0, 0, 0, 0F8h
3183                    db 1, 0, 0, 0, 0, 0, 0, 0,       0, 0, 0, 0, 0, 0, 0, 0,       0, 0, 3
3184                    db 20h,0, 28h,0, 0, 0, 0, 0, 0, 0, 0,       0, 0, 0, 0, 0
3185                    db 0, 0, 0, 0, 0, 0, 0, 0,       0, 0, 0, 0, 0, 0, 0, 0,       0, 0, 0
3186                    db 0, 0FFh, 1, 0, 0, 0,       4Eh, 4Fh, 20h, 4Eh, 41h, 4Dh, 45h
3187                    db 20h,20h, 20h, 20h, 0, 0, 0,       0, 0, 46h, 41h,       54h, 31h
3188                    db 32h,20h, 20h, 20h, 0
3189                    dw 0FFFFh
3190                    db 0, 0, 50h, 3, 0, 2, 1, 1, 0,      2, 10h,0, 0, 0, 0F8h
3191                    db 1, 0, 0, 0, 0, 0, 0, 0,       0, 0, 0, 0, 0, 0, 0, 0,       0, 0, 3
3192                    db 20h,0, 28h,0, 0, 0, 0, 0, 0, 0, 0,       0, 0, 0, 0, 0
3193                    db 0, 0, 0, 0, 0, 0, 0, 0,       0, 0, 0, 0, 0, 0, 0, 0,       0, 0, 0
3194                    db 0, 0FFh, 1, 0, 0, 0,       4Eh, 4Fh, 20h, 4Eh, 41h, 4Dh, 45h
3195                    db 20h,20h, 20h, 20h, 0, 0, 0,       0, 0, 46h, 41h,       54h, 31h
3196                    db 32h,20h, 20h, 20h, 0
3197                    dw 0FFFFh
3198                    db 0, 0, 50h, 3, 0, 2, 1, 1, 0,      2, 10h,0, 0, 0, 0F8h
3199                    db 1, 0, 0, 0, 0, 0, 0, 0,       0, 0, 0, 0, 0, 0, 0, 0,       0, 0, 3
3200                    db 20h,0, 28h,0, 0, 0, 0, 0, 0, 0, 0,       0, 0, 0, 0, 0
3201                    db 0, 0, 0, 0, 0, 0, 0, 0,       0, 0, 0, 0, 0, 0, 0, 0,       0, 0, 0
3202                    db 0, 0FFh, 1, 0, 0, 0,       4Eh, 4Fh, 20h, 4Eh, 41h, 4Dh, 45h
3203                    db 20h,20h, 20h, 20h, 0, 0, 0,       0, 0, 46h, 41h,       54h, 31h
3204                    db 32h,20h, 20h, 20h, 0
3205                    dw 0FFFFh
3206                    db 0, 0, 50h, 3, 0, 2, 1, 1, 0,      2, 10h,0, 0, 0, 0F8h
3207                    db 1, 0, 0, 0, 0, 0, 0, 0,       0, 0, 0, 0, 0, 0, 0, 0,       0, 0, 3
3208                    db 20h,0, 28h,0, 0, 0, 0, 0, 0, 0, 0,       0, 0, 0, 0, 0
3209                    db 0, 0, 0, 0, 0, 0, 0, 0,       0, 0, 0, 0, 0, 0, 0, 0,       0, 0, 0
3210                    db 0, 0FFh, 1, 0, 0, 0,       4Eh, 4Fh, 20h, 4Eh, 41h, 4Dh, 45h
3211                    db 20h,20h, 20h, 20h, 0, 0, 0,       0, 0, 46h, 41h,       54h, 31h
3212                    db 32h,20h, 20h, 20h, 0
3213                    dw 0FFFFh
3214                    db 0, 0, 50h, 3, 0, 2, 1, 1, 0,      2, 10h,0, 0, 0, 0F8h
3215                    db 1, 0, 0, 0, 0, 0, 0, 0,       0, 0, 0, 0, 0, 0, 0, 0,       0, 0, 3
3216                    db 20h,0, 28h,0, 0, 0, 0, 0, 0, 0, 0,       0, 0, 0, 0, 0
3217                    db 0, 0, 0, 0, 0, 0, 0, 0,       0, 0, 0, 0, 0, 0, 0, 0,       0, 0, 0
3218                    db 0, 0FFh, 1, 0, 0, 0,       4Eh, 4Fh, 20h, 4Eh, 41h, 4Dh, 45h
3219                    db 20h,20h, 20h, 20h, 0, 0, 0,       0, 0, 46h, 41h,       54h, 31h
3220                    db 32h,20h, 20h, 20h, 0
3221                    dw 0FFFFh
3222                    db 0, 0, 50h, 3, 0, 2, 1, 1, 0,      2, 10h,0, 0, 0, 0F8h
3223                    db 1, 0, 0, 0, 0, 0, 0, 0,       0, 0, 0, 0, 0, 0, 0, 0,       0, 0, 3
3224                    db 20h,0, 28h,0, 0, 0, 0, 0, 0, 0, 0,       0, 0, 0, 0, 0
3225                    db 0, 0, 0, 0, 0, 0, 0, 0,       0, 0, 0, 0, 0, 0, 0, 0,       0, 0, 0
3226                    db 0, 0FFh, 1, 0, 0, 0,       4Eh, 4Fh, 20h, 4Eh, 41h, 4Dh, 45h
3227                    db 20h,20h, 20h, 20h, 0, 0, 0,       0, 0, 46h, 41h,       54h, 31h
3228                    db 32h,20h, 20h, 20h, 0
3229                    dw 0FFFFh
3230                    db 0, 0, 50h, 3, 0, 2, 1, 1, 0,      2, 10h,0, 0, 0, 0F8h
3231                    db 1, 0, 0, 0, 0, 0, 0, 0,       0, 0, 0, 0, 0, 0, 0, 0,       0, 0, 3
3232                    db 20h,0, 28h,0, 0, 0, 0, 0, 0, 0, 0,       0, 0, 0, 0, 0
3233                    db 0, 0, 0, 0, 0, 0, 0, 0,       0, 0, 0, 0, 0, 0, 0, 0,       0, 0, 0
3234                    db 0, 0FFh, 1, 0, 0, 0,       4Eh, 4Fh, 20h, 4Eh, 41h, 4Dh, 45h
3235                    db 20h,20h, 20h, 20h, 0, 0, 0,       0, 0, 46h, 41h,       54h, 31h
3236                    db 32h,20h, 20h, 20h, 0
3237                    dw 0FFFFh
3238                    db 0, 0, 50h, 3, 0, 2, 1, 1, 0,      2, 10h,0, 0, 0, 0F8h
3239                    db 1, 0, 0, 0, 0, 0, 0, 0,       0, 0, 0, 0, 0, 0, 0, 0,       0, 0, 3
3240                    db 20h,0, 28h,0, 0, 0, 0, 0, 0, 0, 0,       0, 0, 0, 0, 0
3241                    db 0, 0, 0, 0, 0, 0, 0, 0,       0, 0, 0, 0, 0, 0, 0, 0,       0, 0, 0
3242                    db 0, 0FFh, 1, 0, 0, 0,       4Eh, 4Fh, 20h, 4Eh, 41h, 4Dh, 45h
3243                    db 20h,20h, 20h, 20h, 0, 0, 0,       0, 0, 46h, 41h,       54h, 31h
3244                    db 32h,20h, 20h, 20h, 0
```

```
                                            dw 0FFFFh
                                            db 0, 0, 50h, 3, 0, 2, 1, 1, 0,      2, 10h,0, 0, 0, 0F8h
                                            db 1, 0, 0, 0, 0, 0, 0,      0, 0, 0, 0, 0, 0, 0, 0,      0, 0, 3
                                            db 20h,0, 28h,0, 0, 0, 0, 0, 0, 0, 0,      0, 0, 0, 0, 0
                                            db 0, 0, 0, 0, 0, 0, 0,      0, 0, 0, 0, 0, 0, 0, 0,      0, 0, 0
                                            db 0, 0FFh, 1, 0, 0, 0,      4Eh, 4Fh, 20h, 4Eh, 41h, 4Dh, 45h
                                            db 20h,20h, 20h, 20h, 0, 0, 0,      0, 0, 46h, 41h,      54h, 31h
                                            db 32h,20h, 20h, 20h, 0
                                            dw 0FFFFh
                                            db 0, 0, 50h, 3, 0, 2, 1, 1, 0,      2, 10h,0, 0, 0, 0F8h
                                            db 1, 0, 0, 0, 0, 0, 0,      0, 0, 0, 0, 0, 0, 0, 0,      0, 0, 3
                                            db 20h,0, 28h,0, 0, 0, 0, 0, 0, 0, 0,      0, 0, 0, 0, 0
                                            db 0, 0, 0, 0, 0, 0, 0,      0, 0, 0, 0, 0, 0, 0, 0,      0, 0, 0
                                            db 0, 0FFh, 1, 0, 0, 0,      4Eh, 4Fh, 20h, 4Eh, 41h, 4Dh, 45h
                                            db 20h,20h, 20h, 20h, 0, 0, 0,      0, 0, 46h, 41h,      54h, 31h
                                            db 32h,20h, 20h, 20h, 0
                                            dw 0FFFFh
                                            db 0, 0, 50h, 3, 0, 2, 1, 1, 0,      2, 10h,0, 0, 0, 0F8h
                                            db 1, 0, 0, 0, 0, 0, 0,      0, 0, 0, 0, 0, 0, 0, 0,      0, 0, 3
                                            db 20h,0, 28h,0, 0, 0, 0, 0, 0, 0, 0,      0, 0, 0, 0, 0
                                            db 0, 0, 0, 0, 0, 0, 0,      0, 0, 0, 0, 0, 0, 0, 0,      0, 0, 0
                                            db 0, 0FFh, 1, 0, 0, 0,      4Eh, 4Fh, 20h, 4Eh, 41h, 4Dh, 45h
                                            db 20h,20h, 20h, 20h, 0, 0, 0,      0, 0, 46h, 41h,      54h, 31h
                                            db 32h,20h, 20h, 20h, 0
                                            dw 0FFFFh
                                            db 0, 0, 50h, 3, 0, 2, 1, 1, 0,      2, 10h,0, 0, 0, 0F8h
                                            db 1, 0, 0, 0, 0, 0, 0,      0, 0, 0, 0, 0, 0, 0, 0,      0, 0, 3
                                            db 20h,0, 28h,0, 0, 0, 0, 0, 0, 0, 0,      0, 0, 0, 0, 0
                                            db 0, 0, 0, 0, 0, 0, 0,      0, 0, 0, 0, 0, 0, 0, 0,      0, 0, 0
                                            db 0, 0FFh, 1, 0, 0, 0,      4Eh, 4Fh, 20h, 4Eh, 41h, 4Dh, 45h
                                            db 20h,20h, 20h, 20h, 0, 0, 0,      0, 0, 46h, 41h,      54h, 31h
                                            db 32h,20h, 20h, 20h, 0
                                            dw 0FFFFh
                                            db 0, 0, 50h, 3, 0, 2, 1, 1, 0,      2, 10h,0, 0, 0, 0F8h
                                            db 1, 0, 0, 0, 0, 0, 0,      0, 0, 0, 0, 0, 0, 0, 0,      0, 0, 3
                                            db 20h,0, 28h,0, 0, 0, 0, 0, 0, 0, 0,      0, 0, 0, 0, 0
                                            db 0, 0, 0, 0, 0, 0, 0,      0, 0, 0, 0, 0, 0, 0, 0,      0, 0, 0
                                            db 0, 0FFh, 1, 0, 0, 0,      4Eh, 4Fh, 20h, 4Eh, 41h, 4Dh, 45h
                                            db 20h,20h, 20h, 20h, 0, 0, 0,      0, 0, 46h, 41h,      54h, 31h
                                            db 32h,20h, 20h, 20h, 0
                                            dw 0FFFFh
                                            db 0, 0, 50h, 3, 0, 2, 1, 1, 0,      2, 10h,0, 0, 0, 0F8h
                                            db 1, 0, 0, 0, 0, 0, 0,      0, 0, 0, 0, 0, 0, 0, 0,      0, 0, 3
                                            db 20h,0, 28h,0, 0, 0, 0, 0, 0, 0, 0,      0, 0, 0, 0, 0
                                            db 0, 0, 0, 0, 0, 0, 0,      0, 0, 0, 0, 0, 0, 0, 0,      0, 0, 0
                                            db 0, 0FFh, 1, 0, 0, 0,      4Eh, 4Fh, 20h, 4Eh, 41h, 4Dh, 45h
                                            db 20h,20h, 20h, 20h, 0, 0, 0,      0, 0, 46h, 41h,      54h, 31h
                                            db 32h,20h, 20h, 20h, 0
                                            dw 0FFFFh
                                            db 0, 0, 50h, 3, 0, 2, 1, 1, 0,      2, 10h,0, 0, 0, 0F8h
                                            db 1, 0, 0, 0, 0, 0, 0,      0, 0, 0, 0, 0, 0, 0, 0,      0, 0, 3
                                            db 20h,0, 28h,0, 0, 0, 0, 0, 0, 0, 0,      0, 0, 0, 0, 0
                                            db 0, 0, 0, 0, 0, 0, 0,      0, 0, 0, 0, 0, 0, 0, 0,      0, 0, 0
                                            db 0, 0FFh, 1, 0, 0, 0,      4Eh, 4Fh, 20h, 4Eh, 41h, 4Dh, 45h
                                            db 20h,20h, 20h, 20h, 0, 0, 0,      0, 0, 46h, 41h,      54h, 31h
                                            db 32h,20h, 20h, 20h, 0
                                            db 0
                                 %endif

                                 ;--------------------------------------------------------------------------
                                 ; Possibly disposable data, goes at end of data group
                                 ;**************************************************************************

                                 ; Possibly disposable data, goes at end of data group

                                 ;***ibm_disk_io - main routine, fixes at rom bug
                                 ;
                                 ;   entry: (ah) = function, 02 or 0a for read.
                                 ;          (dl) = drive number (80h or 81h).
                                 ;          (dh) = head number.
                                 ;          (ch) = cylinder number.
                                 ;          (cl) = sector number (high 2 bits has cylinder number).
                                 ;          (al) = number of sectors.
                                 ;          (es:bx) = address of read buffer.
                                 ;          for more on register contents see rom bios listing.
                                 ;          stack set up for return by an iret.
                                 ;
                                 ;   exit:  (ah) = status of current operation.
                                 ;          (cy) = 1 if failed, 0 if successful.
                                 ;          for other register contents see rom bios listing.
                                 ;
                                 ;   uses:
                                 ;
                                 ;
                                 ;   warning: uses old13 vector for non-read calls.
                                 ;          does direct calls to the at rom.
                                 ;          does segment arithmatic.
                                 ;
                                 ;   effects: performs disk i/o operation.

                                 ; 16/10/2022
                                 ; 28/05/2019
                                 cmd_block equ 42h ; ROMBIOS DATA segment (40h) offset 42h ; 13/12/2022

                                 ;* offsets into cmd_block for registers

                                 pre_comp equ 0     ;write pre-compensation
                                 sec_cnt      equ 1 ;sector count
                                 sec_num      equ 2 ;sector number
                                 cyl_low      equ 3 ;cylinder number, low part
                                 cyl_high equ 4     ;cylinder number, high part
                                 drv_head equ 5     ;drive/head (bit 7 = ecc mode, bit 5 = 512 byte sectors,
                                            ;           bit 4 = drive number, bits 3-0 have head number)
                                 cmd_reg  equ 6     ;command register

                                 ; 01/10/2022
                                 disk_status1       equ 74h
                                 hf_num             equ 75h
                                 control_byte       equ 76h

                                 ibm_disk_io:
000016F2 80FA80                             cmp    dl, 80h        ; main routine,       fixes at rom bug
000016F5 720A                               jb     short atd1     ; pass through floppy disk calls.
000016F7 80FC02                             cmp    ah, 2
000016FA 740A                               jz     short atd2     ; intercept call 02 (read sectors).
000016FC 80FC0A                             cmp    ah, 0Ah
000016FF 7405                               jz     short atd2     ; and call 0Ah (read long).
                                 atd1:
00001701 2EFF2E[0601]                       jmp    far [cs:Old13]
                                            ;jmp   cs:Old13       ; use rom int 13h handler
                                 ;----------------------------------------------------------------------------

                                 atd2:
00001706 53                                 push   bx
```

```
3369 00001707 51                        push    cx
3370 00001708 52                        push    dx
3371 00001709 57                        push    di
3372 0000170A 1E                        push    ds
3373 0000170B 06                        push    es
3374 0000170C 50                        push    ax
3375 0000170D B84000                    mov     ax, 40h         ; bioseg (rombios data segment)
3376                                                            ; establish bios segment addressing
3377 00001710 8ED8                      mov     ds, ax
3378                                    ; 16/10/2022
3379 00001712 C606740000                mov     byte [disk_status1], 0
3380                                    ;mov    byte ptr ds:74h, 0 ; [disk_status1]
3381                                                            ; initially no error code.
3382 00001717 80E27F                    and     dl, 7Fh         ; mask to hard disk number
3383 0000171A 3A167500                  cmp     dl, [hf_num]
3384                                    ;cmp    dl, ds:75h      ; [hf_num] ; 40h:75h
3385 0000171E 7207                      jb      short atd3      ; disk number in range
3386                                    ;mov    byte ptr ds:74h, 1 ; bad_disk
3387 00001720 C606740001                mov     byte [disk_status1], 1
3388 00001725 EB20                      jmp     short atd4      ; disk number out of range error,
3389                                                            ; return
3390                                ; ---------------------------------------------------------------------------
3391
3392                                atd3:
3393 00001727 53                        push    bx
3394 00001728 8CC0                      mov     ax, es
3395 0000172A C1EB04                    shr     bx, 4           ; make es:bx to seg:000x form.
3396 0000172D 01D8                      add     ax, bx
3397 0000172F 8EC0                      mov     es, ax
3398 00001731 5B                        pop     bx
3399 00001732 83E30F                    and     bx, 0Fh
3400 00001735 0E                        push    cs
3401 00001736 E8DF00                    call    check_dma
3402 00001739 720C                      jb      short atd4      ; abort if dma across segment boundary
3403 0000173B 58                        pop     ax
3404 0000173C 50                        push    ax
3405 0000173D E81A00                    call    setcmd          ; set up command block for disk op
3406 00001740 BAF603                    mov     dx, 3F6h        ; hf_reg_port
3407 00001743 EE                        out     dx, al          ; write out command modifier
3408 00001744 E86B00                    call    docmd           ; carry out command
3409                                ; ---------------------------------------------------------------------------
3410
3411                                atd4:
3412
3413                                ;  new code - let logical or clear carry and then set carry if ah!=0
3414                                ;         and save a couple bytes while were at it.
3415
3416 00001747 58                        pop     ax
3417                                    ;mov    ah, ds:74h      ; [disk_status1]
3418 00001748 8A267400                  mov     ah, [disk_status1]
3419 0000174C 08E4                      or      ah, ah
3420 0000174E 7401                      jz      short atd5
3421 00001750 F9                        stc
3422                                atd5:
3423 00001751 07                        pop     es
3424 00001752 1F                        pop     ds
3425 00001753 5F                        pop     di
3426 00001754 5A                        pop     dx
3427 00001755 59                        pop     cx
3428 00001756 5B                        pop     bx
3429 00001757 CA0200                    retf    2               ; far return, dropping flags
3430
3431                                ; =============== S U B   R O U TI N E =======================================
3432
3433                                ;***setcmd - set up cmd_block for the disk operation
3434                                ;
3435                                ;   entry:  (ds) = bios data segment.
3436                                ;           (es:bx) in seg:000x form.
3437                                ;           other registers as in int 13h call
3438                                ;
3439                                ;   exit:   cmd_block set up for disk read call.
3440                                ;           control_byte set up for disk operation.
3441                                ;           (al) = control byte modifier
3442                                ;
3443                                ;   sets the fields of cmd_block using the register contents
3444                                ;   and the contents of the disk parameter block for the given drive.
3445                                ;
3446                                ;   warning: (ax) destroyed.
3447                                ;           does direct calls to the at rom.
3448
3449                                setcmd:                 ; proc near
3450                                    ;mov    ds:43h,al       ; [cmd_block+sec_cnt]
3451                                    ; 16/10/2022
3452 0000175A A24300                    mov     [cmd_block+sec_cnt], al
3453                                    ;mov    byte ptr ds:48h, 20h ; [cmd_block+cmd_reg]
3454 0000175D C606480020                mov     byte [cmd_block+cmd_reg], 20h ; assume function 02h (read)
3455 00001762 80FC02                    cmp     ah, 2
3456 00001765 7405                      jz      short setc1     ; cmd_reg = 20h       if function 02h       (read)
3457 00001767 C606480022                mov     byte [cmd_block+cmd_reg], 22h
3458                                    ;mov    byte ptr ds:48h, 22h ; [cmd_block+cmd_reg]
3459                                                            ; cmd_reg = 22h       if function 0Ah       (read long)
3460                                setc1:
3461 0000176C 88C8                      mov     al, cl
3462 0000176E 243F                      and     al, 3Fh         ; mask sector number
3463                                    ;mov    ds:44h,al       ; [cmd_block+sec_num]
3464                                    ;mov    ds:45h,ch       ; [cmd_block+cyl_low]
3465 00001770 A24400                    mov     [cmd_block+sec_num], al ; mov [44h],al
3466 00001773 882E4500                  mov     [cmd_block+cyl_low], ch ; mov [45h],ch
3467 00001777 88C8                      mov     al, cl
3468 00001779 C0E806                    shr     al, 6           ; get two high bits of cylinder      number
3469                                    ;mov    ds:46h,al       ; [cmd_block+cyl_high]
3470 0000177C A24600                    mov     [cmd_block+cyl_high], al ; mov [46h],al
3471 0000177F 89D0                      mov     ax, dx
3472 00001781 C0E004                    shl     al, 4           ; drive number
3473 00001784 80E40F                    and     ah, 0Fh
3474 00001787 08E0                      or      al, ah          ; head number
3475 00001789 0CA0                      or      al, 0A0h        ; set ecc and 512 bytes       per sector
3476                                    ;mov    ds:47h,al       ; [cmd_block+drv_head]
3477 0000178B A24700                    mov     [cmd_block+drv_head], al  ; mov [47h],al
3478 0000178E 06                        push    es
3479 0000178F 53                        push    bx
3480 00001790 0E                        push    cs
3481 00001791 E85C00                    call    get_vec
3482 00001794 268B4705                  mov     ax, [es:bx+5]  ; [es:bx+fdp_precomp]
3483                                                            ; write pre-comp from disk parameters
3484 00001798 C1E802                    shr     ax, 2
3485                                    ;mov    ds:42h,al       ; [cmd_block+pre_comp]
3486 0000179B A24200                    mov     [cmd_block+pre_comp], al ; mov [42h],al
3487                                                            ; only use low part
3488 0000179E 268A4708                  mov     al, [es:bx+8]  ; [es:bx+fdp_control]
3489                                                            ; control byte modifier
3490 000017A2 5B                        pop     bx
3491 000017A3 07                        pop     es
3492                                    ;mov    ah, ds:76h      ; [control_byte]
```

```
000017A4 8A267600                       mov     ah, [control_byte] ; mov ah,[76h]
000017A8 80E4C0                         and     ah, 0C0h        ; keep disable retry bits
000017AB 08C4                           or      ah, al
                                        ;mov    ds:76h,ah
000017AD 88267600                       mov     [control_byte], ah ; mov [76h],al
000017B1 C3                             retn

                                ; =============== S U B   R O U T I N E =======================================

                                ;***docmd - carry out read operation to at hard disk
                                ;
                                ;   entry:  (es:bx) = address for read in data.
                                ;           cmd_block set up for disk read.
                                ;
                                ;   exit:   buffer at (es:bx) contains data read.
                                ;           disk_status1 set to error code (0 if success).
                                ;
                                ;
                                ;
                                ;   warning: (ax), (bl), (cx), (dx), (di) destroyed.
                                ;           no check is made for dma boundary overrun.
                                ;
                                ;   effects: programs disk controller.
                                ;           performs disk input.

                                docmd:                  ; proc near
000017B2 89DF                           mov     di, bx
000017B4 0E                             push    cs
000017B5 E84000                         call    command
000017B8 7535                           jnz     short doc3
                                doc1:
000017BA 0E                             push    cs
000017BB E84200                         call    waitt           ; wait for controller to complete read
000017BE 752F                           jnz     short doc3
000017C0 B90001                         mov     cx, 256         ; 256 words per sector
000017C3 BAF001                         mov     dx, 1F0h        ; hf_port
000017C6 FC                             cld                     ; string op goes up
000017C7 FA                             cli                     ; disable interrupts
                                                                ; (bug was forgetting this)

                                ;   M062 -- some of these old machines have intermittent failures
                                ;           when the read is done at full speed. Instead of using
                                ;           a string rep instruction, we'll use a loop. There is
                                ;           a slight performance hit, but it only affects these
                                ;           very old machines with an exact date code match, and
                                ;           it makes said machines more reliable
                                ;
                                ;M062       repz    insw            ;read in sector

                                rsct_loop:
000017C8 6D                             insw
000017C9 E2FD                           loop    rsct_loop
000017CB FB                             sti
                                        ; 16/10/2022
000017CC F606480002                     test    byte [cmd_block+cmd_reg], 02h
                                        ;test   byte ptr ds:48h, 2 ; [cmd_block+cmd_reg]
                                                                ; (ds = 40h)
000017D1 7410                           jz      short doc2      ; no ecc bytes to read.
000017D3 0E                             push    cs
000017D4 E83100                         call    wait_drq        ; wait for controller to complete read
000017D7 7216                           jb      short doc3
000017D9 B90400                         mov     cx, 4           ; 4 bytes of ecc
000017DC BAF001                         mov     dx, 1F0h        ; hf_port
000017DF FA                             cli
000017E0 F36C                           rep insb                ; read in ecc
000017E2 FB                             sti
                                doc2:
000017E3 0E                             push    cs
000017E4 E82900                         call    check_status
000017E7 7506                           jnz     short doc3      ; operation failed
                                        ;dec    byte ptr ds:43h         ; [cmd_block+sec_cnt]
000017E9 FE0E4300                       dec     byte [cmd_block+sec_cnt]
000017ED 75CB                           jnz     short doc1      ; loop while more sectors to read
                                doc3:
000017EF C3                             retn

                                ; =============== S U B   R O U T I N E =======================================

                                ;***define where the rom routines are actually located
                                ;       in the buggy old AT BIOS that we might need to
                                ;       install a special level of int13 handler for

                                ; 16/10/2022

                                romsegment  equ 0F000h  ; segment
                                romcommand  equ 2E1Eh   ; offset in romsegment
                                romwait             equ 2E7Fh   ; offset in romsegment
                                romwait_drq         equ 2EE2h   ; offset in romsegment
                                romcheck_status equ 2EF8h   ; offset in romsegment
                                romcheck_dma        equ 2F69h   ; offset in romsegment
                                romget_vec  equ 2F8Eh   ; offset in romsegment
                                romfret             equ 0FF65h  ; far return in rom

                                ;***get_vec - get pointer to hard disk parameters.
                                ;
                                ;   entry:  (dl) = low bit has hard disk number (0 or 1).
                                ;
                                ;   exit:   (es:bx) = address of disk parameters table.
                                ;
                                ;   uses:   ax for segment computation.
                                ;
                                ;   loads es:bx from interrupt table in low memory, vector 46h (disk 0)
                                ;   or 70h (disk 1).
                                ;
                                ;   warning: (ax) destroyed.
                                ;           this does a direct call to the at rom.

                                get_vec:    ; proc near
                                            ;push   0FF65h          ; romfret ; far       return in rom
                                            ;jmp    far ptr 0F000h:2F8Eh
                                            ; 16/10/2022
000017F0 6865FF                             push    romfret         ; far return in rom
000017F3 EA8E2F00F0                         jmp     romsegment:romget_vec

                                ; =============== S U B   R O U T I N E =======================================

                                ;***command - send contents of cmd_block to disk controller.
                                ;
                                ;   entry:  control_byte
                                ;           cmd_block - set up with values for hard disk controller.
                                ;
                                ;   exit:   disk_status1 = error code.
                                ;           nz if error, zr for no error.
                                ;
```

```
                                ;
                                ;   warning: (ax), (cx), (dx) destroyed.
                                ;           does a direct call to the at rom.
                                ;
                                ;   effects: programs disk controller.

                                command:    ; proc near
                                            ;push   0FF65h          ; romfret ; far       return in rom
                                            ;jmp    far ptr 0F000h:2E1Eh
                                            ; 16/10/2022
000017F8 6865FF                             push    romfret         ; far return in rom
000017FB EA1E2E00F0                         jmp     romsegment:romcommand

                                ; =============== S U B   R O U T I N E =======================================

                                ;***waitt - wait for disk interrupt
                                ;
                                ;   entry:  nothing.
                                ;
                                ;   exit:   disk_status1 = error code.
                                ;           nz if error, zr if no error.
                                ;
                                ;
                                ;   warning: (ax), (bl), (cx) destroyed.
                                ;           does a direct call to the at rom.
                                ;
                                ;   effects: calls int 15h, function 9000h.

                                waitt:              ; proc near
                                            ;push   0FF65h          ; romfret ; far       return in rom
                                            ;jmp    far ptr 0F000h:2E7Fh
                                            ; 16/10/2022
00001800 6865FF                             push    romfret         ; far return in rom
00001803 EA7F2E00F0                         jmp     romsegment:romwait

                                ; =============== S U B   R O U T I N E =======================================

                                ;***wait_drq - wait for data request.
                                ;
                                ;   entry:  nothing.
                                ;
                                ;   exit:   disk_status1 = error code.
                                ;           cy if error, nc if no error.
                                ;
                                ;   warning: (al), (cx), (dx) destroyed.
                                ;           does a direct call to the at rom.

                                wait_drq:   ; proc near
                                            ;push   0FF65h          ; romfret ; far       return in rom
                                            ;jmp    far ptr 0F000h:2EE2h
                                            ; 16/10/2022
00001808 6865FF                             push    romfret         ; far return in rom
0000180B EAE22E00F0                         jmp     romsegment:romwait_drq

                                ; =============== S U B   R O U T I N E =======================================

                                ;***check_status - check hard disk status.
                                ;
                                ;   entry:  nothing.
                                ;
                                ;   exit:   disk_status1 = error code.
                                ;           nz if error, zr if no error.
                                ;
                                ;   warning: (ax), (cx), (dx) destroyed.
                                ;           does a direct call to the at rom.

                                check_status:       ; proc near
                                            ;push   0FF65h          ; romfret ; far       return in rom
                                            ;jmp    far ptr 0F000h:2EF8h
                                            ; 16/10/2022
00001810 6865FF                             push    romfret         ; far return in rom
00001813 EAF82E00F0                         jmp     romsegment:romcheck_status

                                ; =============== S U B   R O U T I N E =======================================

                                ;***check_dma - check for dma overrun 64k segment.
                                ;
                                ;   entry:  (es:bx) = addr. of memory buffer in seg:000x form.
                                ;           cmd_block set up for operation.
                                ;
                                ;   exit:   disk_status1 - error code.
                                ;           cy if error, nc if no error.
                                ;
                                ;   warning: does a direct call to the at rom.

                                check_dma:  ; proc near
                                            ;push   0FF65h          ; romfret ; far       return in rom
                                            ;jmp    far ptr 0F000h:2F69h
                                            ; 16/10/2022
00001818 6865FF                             push    romfret         ; far return in rom
0000181B EA692F00F0                         jmp     romsegment:romcheck_dma

                                ;--------------------------------------------------------------------------

                                endatrom:

                                ; --------------------------------------------------------------------------

                                ;; M015 -- begin changes
                                ;;
                                ;; Certain old COMPAQ '286 machines have a bug in their ROM BIOS.
                                ;; When Int13 is done with AH > 15h and DL >= 80h, they trash
                                ;; the byte at DS:74h, assuming that DS points to ROM_DATA.
                                ;; If our init code detects this error, it will install this
                                ;; special Int13 hook through the same mechanism that was set
                                ;; up for the IBM patch above. This code is also dynamically
                                ;; relocated by MSINIT.

                                compaq_disk_io:
00001820 80FC15                             cmp     ah, 15h         ; compaq_disk_io proc far
                                                                    ;
                                                                    ; the following       label defines the end of the at      rom patch.
                                                                    ; this is used at configuration      time.
                                                                    ;
                                                                    ; warning!!!
                                                                    ; this code will be dynamically      relocated by msinit
00001823 7705                               ja      short mebbe_hookit ; only deal with functions > 15h
                                no_hookit:
                                            ;jmp    cs:Old13
                                            ; 16/10/2022
00001825 2EFF2E[0601]                       jmp     far [cs:Old13]

                                ; --------------------------------------------------------------------------

```

```
                                mebbe_hookit:
0000182A 80FA80                         cmp     dl, 80h
0000182D 72F6                           jb      short no_hookit
0000182F 1E                             push    ds

                                        ; 07/08/2023 - Retro DOS v4.2 IO.SYS (optimization)
                                        ; PCDOS 7.1 IBMBIO.COM - BIOSDATA:1830h
                                        ;push   ax
                                        ;mov    ax, 40h
                                        ;mov    ds, ax
                                        ;pop    ax
00001830 6A40                           push    40h
00001832 1F                             pop     ds

00001833 9C                             pushf
                                        ;call   cs:Old13
                                        ; 16/10/2022
00001834 2EFF1E[0601]                   call    far [cs:Old13]
00001839 1F                             pop     ds
0000183A CA0200                         retf    2

                                ; ---------------------------------------------------------------------------

0000183D 00                     end_compaq_i13hook: db 0

                                ; =============== S U B   R O U T I N E =======================================

                                ; 08/08/2023 - Retro DOS v4.2 IO.SYS (optimization)
                                %if 0

                                ; CMOS Clock setting support routines used by MSCLOCK.
                                ; Warning!!! This code will be dynamically relocated by MSINIT.

                                daycnt_to_day:      ; proc far

                                ; entry: [daycnt] = number of days since 1-1-80
                                ;
                                ; return: ch - century in bcd
                                ;       cl - year in bcd
                                ;       dh - month in bcd
                                ;       dl - day in bcd

                                        ; 16/10/2022
                                        push    word [cs:daycnt] ; save daycnt
                                        cmp     word [cs:daycnt], 7305; (365*20+(20/4))
                                                                ; # days from 1-1-1980 to 1-1-2000
                                        jnb     short century20
                                        mov     byte [cs:base_century], 19
                                        mov     byte [cs:base_year], 80
                                        jmp     short years
                                ; ---------------------------------------------------------------------------

                                century20:
                                        mov     byte [cs:base_century], 20
                                        mov     byte [cs:base_year], 0
                                        sub     word [cs:daycnt], 7305; (365*20+(20/4))
                                                                ; adjust daycnt
                                years:
                                        xor     dx, dx
                                        mov     ax, [cs:daycnt]
                                        mov     bx, 1461        ; (366+365*3)
                                                                ; # of days in a Leap year block
                                        div     bx              ; AX = # of leap block,       DX = daycnt
                                        mov     [cs:daycnt], dx         ; save daycnt left
                                        mov     bl, 4
                                        mul     bl              ; AX = # of years. Less       than 100
                                        add     [cs:base_year], al ; So, ah = 0. Adjust year
                                        inc     word [cs:daycnt]        ; set daycnt to       1 base
                                        cmp     word [cs:daycnt], 366 ; daycnt=remainder of leap year      bk
                                        jbe     short leapyear ; within 366+355+355+355 days.
                                        inc     byte [cs:base_year]     ; if daycnt <= 366, then leap year
                                        sub     word [cs:daycnt], 366 ; else daycnt--, base_year++ ;
                                        mov     cx, 3           ; And next three years are normal
                                regularyear:
                                        cmp     word [cs:daycnt], 365 ; for(i=1; i>3 or daycnt <=365;      i++)
                                        jbe     short yeardone ; {if (daycnt >        365)
                                        inc     byte [cs:base_year]     ;   { daycnt -=       365
                                        sub     word [cs:daycnt], 365 ;   }
                                        loop    regularyear     ; }
                                                                ;
                                                                ; should never fall through loop
                                leapyear:
                                        mov     byte [cs:month_tab+1], 29 ; leap year.
                                                                ; change month table.
                                yeardone:
                                        xor     bx, bx
                                        xor     dx, dx
                                        mov     ax, [cs:daycnt]
                                        ;mov    si, offset month_tab
                                        mov     si, month_tab  ; 19/10/2022
                                        mov     cx, 12
                                months:
                                        inc     bl

                                        ; !!! -- 16/10/2022 -- (if DS=CS, what for CS: prefixes are used !?)
                                        ;mov    dl, [cs:si]
                                        ; !!! -- 16/10/2022 -- (may be to keep code addrs as unchanged/fix!?)
                                        ; ds = cs !? ((ofcourse ds must be same with cs here))
                                        ;mov    dl, [si] ; 20/03/2019 (MSDOS 6.21 IO.SYS, BIOSDATA:14C0h)
                                        ;mov    dl, [si] ; 16/10/2022 (MSDOS 5.0 IO.SYS, BIOSDATA:14C0h)

                                        mov     dl, [si] ; ?    ; mov dl, [cs:si]
                                        cmp     ax, dx          ; cmp daycnt for each month till fit
                                                                ; dh=0
                                        jbe     short month_done
                                        inc     si              ; next month
                                        sub     ax, dx          ; adjust daycnt
                                        loop    months          ;
                                                                ; should never fall through loop
                                month_done:
                                        mov     byte [cs:month_tab+1], 28
                                                                ; restore month table value
                                        mov     dl, bl
                                        mov     dh, [cs:base_year]
                                        mov     cl, [cs:base_century] ; al=day,dl=month,dh=year,cl=cntry
                                        call    far [cs:bintobcd]
                                        ;call   cs:bintobcd     ; convert "day"       to bcd
                                                                ; dl = bcd day,       al = month
                                        xchg    dl, al
                                        call    far [cs:bintobcd]
                                        ;call   cs:bintobcd     ; dh = bcd month, al = year
                                        xchg    dh, al
                                        call    far [cs:bintobcd]
                                        ;call   cs:bintobcd     ; cl = bcd year, al = century
```

```
                                xchg    cl, al
                                call    far [cs:bintobcd]
                                ;call   cs:bintobcd     ; ch = bcd century
                                mov     ch, al
                                pop     word [cs:daycnt] ; restore original value
                                retf

                        enddaycnttoday:

                        %endif

                        ; ============== S U B   R O U T I N E =======================================

                        ; 08/08/2023 - Retro DOS v4.2 IO.SYS (optimization)
                        %if 0

                        bin_to_bcd: ; proc far              ; real time clock support

                        ;convert a binary input in al (less than 63h or 99 decimal)
                        ;into a bcd value in al. ah destroyed.

                                push    cx
                                aam                     ; al=high digit       bcd, ah=low digit bcd
                                mov     cl, 4
                                shl     ah, cl          ; mov the high digit to      high nibble
                                or      al, ah
                                pop     cx
                                retf
                        %endif

                        ; ----------------------------------------------------------------------------

                        ; the k09 requires the routines for reading the clock because of the suspend/
                        ; resume facility. the system clock needs to be reset after resume.

                        ; the following routine is executed at resume time when the system
                        ; powered on after suspension. it reads the real time clock and
                        ; resets the system time and date, and then irets.

                        ; warning!!! this code will be dynamically relocated by msinit.

                            ; 08/08/2023 - Retro DOS v4.2 IO.SYS (optimization)
                            ; PCDOS 7.1 IBMBIO.COM - BIOSDATA:183Eh
                        int_6Ch:
0000183E 0E                             push    cs
0000183F 1F                             pop     ds
                                        ;cmp    byte [cs:inHMA], 0
00001840 803E[0D00]00                   cmp     byte [inHMA], 0
00001845 7405                           jz      short int6c
00001847 BB[2A07]                       mov     bx, EnsureA20On
0000184A FFD3                           call    bx
                        int6c:
                                        ;push   cs
                                        ;pop    ds
0000184C 8F06[F805]                     pop     word [int6c_ret_addr] ; pop off return address
00001850 8F06[FA05]                     pop     word [int6c_ret_addr+2]
00001854 9D                             popf
00001855 E81300                         call    read_real_date ; get the date from the clock
00001858 FA                             cli
00001859 8936[8904]                     mov     [daycnt], si   ; update dos copy of date
0000185D FB                             sti
0000185E E8B900                         call    read_real_time ; get the time from the       rtc
00001861 FA                             cli
00001862 B401                           mov     ah, 1
00001864 CD1A                           int     1Ah             ; CLOCK - SET TIME OF DAY
                                                                ; CX:DX = clock count
                                                                ; Return: time of day set
00001866 FB                             sti
                                        ;jmp    int6c_ret_addr ; long jump
                                        ; 16/10/2022
00001867 FF2E[F805]                     jmp     far [int6c_ret_addr] ; long jump

                        ; ============== S U B   R O U T I N E =======================================

                        ;   read_real_date reads real-time clock for date and returns the number
                        ;   of days elapsed since 1-1-80 in si

                        read_real_date:     ; proc near
0000186B 50                             push    ax
0000186C 51                             push    cx
0000186D 52                             push    dx
0000186E 30E4                           xor     ah, ah          ; throw away clock roll       over
00001870 CD1A                           int     1Ah             ; CLOCK - GET TIME OF DAY
                                                                ; Return: CX:DX       = clock count
                                                                ; AL = 00h if clock was       read or written (via AH=0,1) since the previous
                                                                ; midnight
                                                                ; Otherwise, AL       > 0
00001872 5A                             pop     dx
00001873 59                             pop     cx
00001874 58                             pop     ax
00001875 50                             push    ax
00001876 53                             push    bx
00001877 51                             push    cx
00001878 52                             push    dx
                                        ;mov    word [cs:daycnt2], 1
                                        ; 08/08/2023 - Retro DOS v4.2 IO.SYS (optimization)
                                        ; PCDOS 7.1 IBMBIO.COM - BIOSDATA:187Ah
00001879 C706[0006]0100                 mov     word [daycnt2], 1
                                                                ; REAL TIME CLOCK ERROR       FLAG (+1 DAY)
0000187F B404                           mov     ah, 4
00001881 CD1A                           int     1Ah             ; CLOCK - READ DATE FROM REAL TIME CLOCK (AT,XT286,CONV,PS)
                                                                ; Return: DL = day in BCD
                                                                ; DH = month in       BCD
                                                                ; CL = year in BCD
                                                                ; CH = century (19h or 20h)
00001883 7303                           jnb     short read_ok
00001885 E98300                         jmp     r_d_ret
                        ;---------------------------------------------------------------------------

                        read_ok:
00001888 882E[FC05]                     mov     [bin_date_time], ch
0000188C 880E[FD05]                     mov     [bin_date_time+1], cl
00001890 8836[FE05]                     mov     [bin_date_time+2], dh
00001894 8816[FF05]                     mov     [bin_date_time+3], dl
                                        ;mov    word [cs:daycnt2], 2 ; READ OF R-T CLOCK SUCCESSFUL
                                        ; 08/08/2023
                                        ;mov    byte [daycnt2], 2
00001898 FE06[0006]                     inc     byte [daycnt2] ; 2
0000189C E83401                         call    bcd_verify     ; verify bcd values in range
0000189F 726A                           jb      short r_d_ret  ; some value out of range
                                        ;mov    word [cs:daycnt2], 3
                                        ; 08/08/2023
                                        ;mov    byte [daycnt2], 3
000018A1 FE06[0006]                     inc     byte [daycnt2] ; 3
```

```
000018A5 E8DB00                         call    date_verify
000018A8 7261                           jb      short r_d_ret
                                        ;mov    word [cs:daycnt2], 0
                                        ; 08/08/2023
000018AA C606[0006]00                   mov     byte [daycnt2], 0
000018AF E8A100                         call    in_bin
000018B2 A0[FD05]                       mov     al, [bin_date_time+1]
000018B5 98                             cbw
000018B6 803E[FC05]14                   cmp     byte [bin_date_time], 20 ; 20th century?
000018BB 7503                           jnz     short century_19 ; no
000018BD 83C064                         add     ax, 100         ; add in a century
                                century_19:
000018C0 83E850                         sub     ax, 80          ; subtract off 1-1-80
000018C3 B104                           mov     cl, 4           ; leap year every 4
000018C5 F6F1                           div     cl              ; al= # leap year blocks, ah= remainder
000018C7 88E3                           mov     bl, ah          ; save odd years
000018C9 98                             cbw                     ; zero ah
000018CA B9B505                         mov     cx, 1461        ; 366+(3*365)
                                                                ; # of days in leap year blocks
000018CD F7E1                           mul     cx
                                        ;mov    [cs:daycnt2], ax ; SAVE COUNT OF DAYS
                                        ; 08/08/2023
000018CF A3[0006]                       mov     [daycnt2], ax
000018D2 88D8                           mov     al, bl          ; get odd years       count
000018D4 98                             cbw
000018D5 09C0                           or      ax, ax
000018D7 740B                           jz      short leap_year
000018D9 B96D01                         mov     cx, 365         ; days in year
000018DC F7E1                           mul     cx
                                        ;add    [cs:daycnt2], ax ; ADD ON DAYS IN ODD YEARS
                                        ; 08/08/2023
000018DE 0106[0006]                     add     [daycnt2], ax
000018E2 EB07                           jmp     short leap_adjustment ;      account for leap year
                                                                ; possibly account for a leap day
                                ;-------------------------------------------------------------------------------

                                leap_year:
000018E4 803E[FE05]02                   cmp     byte [bin_date_time+2], 2 ; is       month february?
000018E9 7604                           jbe     short no_leap_adjustment ; jan or feb. no leap day yet.
                                leap_adjustment:
                                        ;inc    word [cs:daycnt2] ; account for leap day
                                        ; 08/08/2023
000018EB FF06[0006]                     inc     word [daycnt2]
                                no_leap_adjustment:
000018EF 8A0E[FF05]                     mov     cl, [bin_date_time+3] ; get days of month
000018F3 30ED                           xor     ch, ch
000018F5 49                             dec     cx              ; because of offset from day 1,      not day 0
                                        ;add    [cs:daycnt2], cx ; GET DAYS IN MONTHS PRECEEDING
                                        ; 08/08/2023
000018F6 010E[0006]                     add     [daycnt2], cx
000018FA 8A0E[FE05]                     mov     cl, [bin_date_time+2] ; get month
                                        ; 08/08/2023
                                        ;xor    ch, ch
000018FE 49                             dec     cx              ; january starts at offset 0

                                        ; 08/08/2023
                                        ;shl    cx, 1           ; word offset
                                        ;;mov   si, month_table
                                        ;add    si, cx
                                        ;; 16/10/2022
                                        ;; ds must be same with cs here, if so..
                                        ;; what for cs: prefixes are used !?)
                                        ;; mov  ax, [cs:si]
                                        ;; mov  ax, [si] ; 16/10/2022 (MSDOS 5.0 IO.SYS - BIOSDATA:15D5h)
                                        ;mov    ax, [si]        ; mov ax, [cs:si]
                                        ;                       ; get # days in previous months
                                        ;add    [cs:daycnt2], ax

                                        ; 08/08/2023 - Retro DOS v4.2 IO.SYS (optimization)
                                        ; PCDOS 7.1 IBMBIO.COM - BIOSDATA:1907h
000018FF B400                           mov     ah, 0
00001901 BE[8F04]                       mov     si, month_tab
                                r_d_sum_loop:
00001904 AC                             lodsb
00001905 0106[0006]                     add     [daycnt2], ax
00001909 E2F9                           loop    r_d_sum_loop
                                r_d_ret:
                                        ;mov    si, [cs:daycnt2]
                                        ; 08/08/2023
0000190B 8B36[0006]                     mov     si, [daycnt2]
0000190F 5A                             pop     dx
00001910 59                             pop     cx
00001911 5B                             pop     bx
00001912 58                             pop     ax
00001913 C3                             retn

                                ;-------------------------------------------------------------------------------

                                r_t_retj:
00001914 31C9                           xor     cx, cx
00001916 31D2                           xor     dx, dx
00001918 EB38                           jmp     short r_t_ret

                                ; =============== S U B   R O U T I N E =======================================

                                ; read_real_time reads the time from the rtc. on exit, it has the number of
                                ; ticks (at 18.2 ticks per sec.) in cx:dx.

                                read_real_time:    ; proc near
0000191A B402                           mov     ah, 2
0000191C CD1A                           int     1Ah             ; CLOCK - READ REAL TIME CLOCK (AT,XT286,CONV,PS)
                                                                ; Return: CH = hours in       BCD
                                                                ; CL = minutes in BCD
                                                                ; DH = seconds in BCD
0000191E 72F4                           jb      short r_t_retj
00001920 882E[FC05]                     mov     [bin_date_time], ch ; hours
00001924 880E[FD05]                     mov     [bin_date_time+1], cl ; minutes
00001928 8836[FE05]                     mov     [bin_date_time+2], dh ; seconds
0000192C C606[FF05]00                   mov     byte [bin_date_time+3], 0 ; unused for time
00001931 E89F00                         call    bcd_verify
00001934 72DE                           jb      short r_t_retj
00001936 E88500                         call    time_verify
00001939 72D9                           jb      short r_t_retj
0000193B E81500                         call    in_bin          ; from bcd to bin
0000193E 8A2E[FC05]                     mov     ch, [bin_date_time]
00001942 8A0E[FD05]                     mov     cl, [bin_date_time+1]
00001946 8A36[FE05]                     mov     dh, [bin_date_time+2]
0000194A 8A16[FF05]                     mov     dl, [bin_date_time+3]
                                        ; 16/10/2022
                                        ; 17/09/2022
                                        ; 31/05/2019
0000194E FF1E[0606]                     call    far [ttticks]
                                        ;call   dword ptr ttticks ; note: indirect far call
                                                                ; cx:dx = number of ticks
```

```
                                                        ; (at 18.2 ticks per sec.)
                                r_t_ret:
00001952 C3                             retn

                                ; =============== S U B   R O U TI N E =======================================

                                ;   in_bin converts bin_date_time values from bcd to bin

                                in_bin:             ; proc near
00001953 A0[FC05]                       mov     al, [bin_date_time] ; century or hours
00001956 E81F00                         call    bcd_to_bin
00001959 A2[FC05]                       mov     [bin_date_time], al
0000195C A0[FD05]                       mov     al, [bin_date_time+1] ; years or minutes
0000195F E81600                         call    bcd_to_bin
00001962 A2[FD05]                       mov     [bin_date_time+1], al
00001965 A0[FE05]                       mov     al, [bin_date_time+2] ; months or seconds
00001968 E80D00                         call    bcd_to_bin
0000196B A2[FE05]                       mov     [bin_date_time+2], al
0000196E A0[FF05]                       mov     al, [bin_date_time+3] ; days (not used for time)
00001971 E80400                         call    bcd_to_bin
00001974 A2[FF05]                       mov     [bin_date_time+3], al
00001977 C3                             retn

                                ; =============== S U B   R O U TI N E =======================================

                                ;   bcd_to_bin converts two bcd nibbles in al (value <= 99.) to
                                ;   a binary representation in al
                                ;   ah is destroyed

                                bcd_to_bin: ; proc near
00001978 88C4                           mov     ah, al
0000197A 240F                           and     al, 0Fh
0000197C B104                           mov     cl, 4
0000197E D2EC                           shr     ah, cl
00001980 D50A                           aad
00001982 C3                             retn

                                ; =============== S U B   R O U TI N E =======================================

                                ;   date_verify loosely checks bcd date values to be in range
                                ;   in bin_date_time

                                date_verify:        ; proc near
00001983 803E[FC05]20                   cmp     byte [bin_date_time], 20h ; century check
00001988 7732                           ja      short date_error
0000198A 740E                           jz      short century_20 ; jmp in 21th century
0000198C 803E[FC05]19                   cmp     byte [bin_date_time], 19h ; century check
                                        ;jb     short date_error
                                        ; 12/12/2022
00001991 722A                           jb      short date_err2
00001993 803E[FD05]80                   cmp     byte [bin_date_time+1], 80h ; year check
                                        ;jb     short date_error
                                        ; 12/12/2022
00001998 7223                           jb      short date_err2
                                century_20:
0000199A 803E[FD05]99                   cmp     byte [bin_date_time+1], 99h ; year check
0000199F 771B                           ja      short date_error
000019A1 803E[FE05]12                   cmp     byte [bin_date_time+2], 12h ; month check
000019A6 7714                           ja      short date_error
000019A8 803E[FE05]00                   cmp     byte [bin_date_time+2], 0
                                        ;jbe    short date_error
000019AD 760D                           jna     short date_error
000019AF 803E[FF05]31                   cmp     byte [bin_date_time+3], 31h ; day check
000019B4 7706                           ja      short date_error
                                        ;cmp    byte [bin_date_time+3], 0 ; day check
                                        ;;jbe   short date_error
                                        ;jna    short date_error
                                        ; 12/12/2022
                                        ; cf=0
                                        ;clc
                                        ; 12/12/2022
000019B6 803E[FF05]01                   cmp     byte [bin_date_time+3], 1 ; day check
000019BB C3                             retn
                                ;---------------------------------------------------------------------------

                                date_error:
000019BC F9                             stc
                                date_err2:
000019BD C3                             retn

                                ; =============== S U B   R O U TI N E =======================================

                                ; time_verify very loosely checks bcd date values to be in range
                                ; in bin_date_time

                                time_verify:        ; proc near
000019BE 803E[FC05]24                   cmp     byte [bin_date_time], 24h ; hour check
000019C3 770C                           ja      short time_error
000019C5 803E[FD05]59                   cmp     byte [bin_date_time+1], 59h ; minute check
000019CA 7705                           ja      short time_error
                                        ; 12/12/2022h
                                        ;cmp    byte [bin_date_time+2], 59h ; second check
                                        ;ja     short time_error
                                        ;clc
                                        ;retn
                                        ; 12/12/2022
000019CC 803E[FE05]5A                   cmp     byte  [bin_date_time+2], 5Ah
                                time_error:
                                bv_error:
000019D1 F5                             cmc     ; cf=0 -> cf=1, cf=1 -> cf=0
000019D2 C3                             retn

                                ; ---------------------------------------------------------------------------

                                ;time_error:
                                        ;stc
                                        ;retn

                                ; =============== S U B   R O U TI N E =======================================

                                ;   bcd_verify checks values in bin_date_time to be valid
                                ;   bcd numerals.  carry set if any nibble out of range

                                bcd_verify: ; proc near
000019D3 B90400                         mov     cx, 4           ; 4 bytes to check
000019D6 BB[FC05]                       mov     bx, bin_date_time
                                bv_loop:
000019D9 8A07                           mov     al, [bx]        ; get a bcd number (0..99)
000019DB 88C4                           mov     ah, al
000019DD 250FF0                         and     ax, 0F00Fh      ; 10's place in high ah, 1's in al
                                                                ; is 1's place in range?
000019E0 3C0A                           cmp     al, 10
000019E2 77ED                           ja      short bv_error ; jmp out of range
000019E4 D0EC                           shr     ah, 1
```

```
000019E6 D0EC                           shr     ah, 1
000019E8 D0EC                           shr     ah, 1
000019EA D0EC                           shr     ah, 1
000019EC 80E40F                         and     ah, 0Fh         ; get rid of any erroneous bits
000019EF 80FC0A                         cmp     ah, 10          ; is 10's place in range
000019F2 77DD                           ja      short bv_error ; jmp out of range
000019F4 43                             inc     bx              ; next byte
000019F5 49                             dec     cx
000019F6 75E1                           jnz     short bv_loop
000019F8 F8                             clc                     ; set success flag
000019F9 C3                             retn
                                ; ---------------------------------------------------------------------------

                                        ; 12/12/2022
                                ;bv_error:
                                        ;stc                    ; set error flag
                                        ;retn

                                ; ---------------------------------------------------------------------------

                                endk09:

                                ; ---------------------------------------------------------------------------

                                ;--------------------------------------------------------------------
                                ;                                                                   :
                                ;   System initialization                                          :
                                ;                                                                   :
                                ;   The entry conditions are established by the bootstrap               :
                                ;   loader and are considered unknown. The following jobs               :
                                ;   will be performed by this module:                               :
                                ;                                                                   :
                                ;   1.      All device initialization is performed                      :
                                ;   2.      A local stack is set up and DS:SI are set                   :
                                ;           to point to an initialization table. Then                   :
                                ;           an inter-segment call is made to the first                  :
                                ;           byte of the dos                                             :
                                ;   3.      Once the dos returns from this call the ds                  :
                                ;           register has been set up to point to the start              :
                                ;           of free memory. The initialization will then                :
                                ;           load the command program into this area                     :
                                ;           beginning at 100 hex and transfer control to                :
                                ;           this program.                                               :
                                ;                                                                   :
                                ;--------------------------------------------------------------------

                                ; 01/10/2022
                                ; 08/01/2018 - Retro DOS v4.0

                                ; drvfat must be the first location of freeable space!

                                align 2
                                        ;db 90h

                                ; 09/12/2023 - Retro DOS v5.0 IO.SYS (IBMBIO.COM 7.1)
                                ; (PCDOS 7.1 IBMBIO.COM, BIOSDATA:1A0Ch)

                                ; 30/12/2022
                                ; (MSDOS 6.21 IO.SYS, BIOSDATA:16D6h)

000019FA 0000                   drvfat:         dw 0                    ; drive and fat id of dos
                                ; 09/12/2023
                                ;bios_l:    dw 0                ; first sector of data (low word)
                                ;bios_h:    dw 0                ; first sector of data (high word)
                                First_Data_Sector:
000019FC 0000                           dw 0
000019FE 0000                           dw 0
00001A00 0000                   doscnt:         dw 0                    ; how many sectors to read
                                ;fbigfat:   db 0                ; flags for drive
00001A02 0000                   fatloc:         dw 0                    ; seg addr of fat sector
00001A04 0000                   init_bootseg:   dw 0                    ; seg addr of buffer for reading boot record
                                ; 09/12/2023
00001A06 00                     fbigfat:    db 0                ; flags for drive
00001A07 80                     rom_drv_num:    db 80h                  ; rom drive number
00001A08 0002                   md_sectorsize:  dw 200h                 ; used by get_fat_sector proc.
                                ; 12/12/2023
                                ;temp_cluster:  dw 0                    ; used by get_fat_sector proc.
00001A0A FFFF                   last_fat_sec_num: dw 0FFFFh             ; used by get_fat_sector proc.

                                ; the following two bytes are used to save the info returned by int 13, ah = 8
                                ; call to determine drive parameters.

00001A0C 02                     num_heads:  db 2    ; dw 2          ; number of heads returned by rom
00001A0D 00                             db 0    ; 09/12/2023
                                ;sec_trk:   db 9                    ; sec/trk returned by rom
00001A0E 28                     num_cyln:   db 40   ; dw 40         ; number of cylinders returned by rom
00001A0F 00                             db 0    ; 09/12/2023
                                ; 09/12/2023
00001A10 09                     sec_trk:    db 9                    ; sec/trk returned by rom
00001A11 00                     fakefloppydrv:  db 0                    ; if 1, then no diskette drives       in the system.

                                ; 09/12/2023 - Retro DOS v5.0 IO.SYS/IBMBIO.COM
                                Orig_Int1Eh_Table:
00001A12 0000                           dw 0
00001A14 0000                           dw 0

                                ; ---------------------------------------------------------------------------

                                ; 09/12/2023
                                %if 0

                                disktable:  dw 512,0100h, 64, 0   ; warning !!! old values
                                            dw 2048, 0201h, 112, 0
                                            dw 8192, 0402h, 256, 0
                                            dw 32680, 0803h, 512, 0     ; warning !!! old values
                                            dw 65535, 1004h, 1024, 0
                                                                    ; default disktable under
                                                                    ; the assumption of total fat size <= 128 kb,
                                                                    ; and the maximum size of fat entry = 16 bit.
                                %endif

                                ; 09/12/2023 - Retro DOS v5.0 IO.SYS (IBMBIO.COM 7.1)
                                ; (PCDOS 7.1 IBMBIO.COM, BIOSDATA:1A2Ah)

                                            ; 09/12/2023
                                            ; 08/08/2023
                                            ; disktable.totalsectors: resw 1 ; high word
                                            ;                         resw 1 ; low word
                                            ; disktable.shiftcount:   resb 1
                                            ; disktable.secperclus:   resb 1
                                            ; disktable.rdirentries:  resw 1
                                            ; disktable.bigflag:      resw 1
00001A16 0000A87F0308000200-    disktable2: dw 0, 32680, 0803h, 512, 0 ; for compatibility.
00001A1F 00
```

```
                                                        ; (32680 sectors, 16340 KB)
00001A20 040000000204000240-            dw 4, 0, 0402h, 512, 40h   ; covers upto 134 mb media.
00001A29 00
                                                        ; fbig = 40h  ; (40000h sectors = 128 MB)
00001A2A 080000000308000240-            dw 8, 0, 0803h, 512, 40h   ; upto 268 mb ; (80000h sectors = 256 MB)
00001A33 00
00001A34 100000000410000240-            dw 16, 0, 1004h, 512, 40h  ; upto 536 mb ; (100000h sectors = 512 MB)
00001A3D 00
00001A3E 200000000520000240-            dw 32, 0, 2005h, 512, 40h  ; upto 1072 mb ; (200000h sectors = 1024 MB)
00001A47 00
00001A48 400000000640000240-            dw 64, 0, 4006h, 512, 40h  ; upto 2144 mb ; (400000h sectors = 2048 MB)
00001A51 00
                                        ; 09/12/2023
                                        ;dw 128, 0, 8007h, 512, 40h ; upto 4288 mb ; (800000h sectors = 4096 MB)
00001A52 FFFFFFFF0308000060-            dw 0FFFFh, 0FFFFh, 0803h, 0, 60h ; FAT32 (> 2144MB)
00001A5B 00
                                                        ; (fbig and fbigbig flags are set)

                                ; ---------------------------------------------------------------------------

                                ;*******************************************************
                                ;variables for mini disk initialization
                                ;*******************************************************

                                ; 01/10/2022
                                ; [ Note: Minidisk == logical dos drive (in extended dos partition) ]

00001A5C 00                     rom_minidisk_num: db 0                  ; temp variable       for phys unit
00001A5D 00                     hnum:             db 0                  ; real number of hardfiles
00001A5E [3C05]                 last_dskdrv_table: dw dskdrvs           ; index into dskdrv table
00001A60 [4C08]                 end_of_bdss:      dw bdss               ; offset value of the ending address
                                                                ; of bds table. needed to figure out
                                                                ; the dosdatasg address.
00001A62 0000                   mini_hdlim: dw 0
00001A64 0000                   mini_seclim:      dw 0

                                ; 19/12/2023
                                ; 09/12/2023
                                ; (PCDOS 7.1 IBMBIO.COM, BIOSDATA:1A7Ah)
                                ;ld_p_number:      dw 2BADh              ; (for 'find_mini_partition' proc)

                                ;end of mini disk init variables **********************

                                ; ---------------------------------------------------------------------------

00001A66 30312F31302F383400     bios_date:  db '01/10/84',0       ; used for checking at rom bios      date.

                                ; 13/12/2022
                                %if 0

                                ;align 2
                                            db  90h

                                ; the following are the recommended bpbs for the media that we know of so far.

                                ;struc bpbx
                                ;   resw 1 ; 512
                                ;   resb 1
                                ;   resw 1 ; 1
                                ;   resb 1 ; 2
                                ;   resw 1
                                ;   resw 1
                                ;   resb 1
                                ;   resw 1
                                ;   resw 1
                                ;   resw 1 ; 2
                                ;   resw 1
                                ;   resw 1 ; hidden sector high
                                ;   resd 1 ; extended total sectors
                                ;.size:
                                ;endstruc

                                ; 08/01/2019 - Retro DOS v4.0

                                ; 20/04/2019

                                ; 01/10/2022 - Retro DOS v4.0 (MSDOS 5.0) IO.SYS

                                ; 09/12/2023 - Retro DOS v5.0 (PCDOS 7.1) IBMBIO.COM
                                ; (PCDOS 7.1 IBMBIO.COM, BIOSDATA:1A86h)

                                ; 09/12/2022
                                BPB48T:
                                ;bpb48t:    ; bpbx <512, 2, 1, 2, 112, 720, 0FDh, 2, 9, 2, 0, 0, 0, 0>
                                            ; 48 tpi diskettes    ;
                                            dw     512            ; physical sector size in bytes
                                            db     2              ; sectors/allocation unit
                                            dw     1              ; reserved sectors for dos
                                            db     2              ; number of allocation tables
                                            dw     112            ; number of directory entries
                                            dw     720 ; 2*9*40   ; number of sectors (at 512 bytes each)
                                            db     0FDh           ; media descriptor
                                            dw     2              ; number of fat sectors
                                            dw     9              ; sectors per track
                                            dw     2              ; heads
                                            dw     0              ; hidden sector count (low word)
                                            dw     0              ; hidden sector (high)
                                            dw     0              ; number of sectors (low)
                                            dw     0              ; number of sectors (high)
                                            ; 09/12/2023
                                            ; FAT32 extensions (to BDS)
                                            times  28 db 0
                                            ;
                                            db 90h
                                ;align 2
                                BPB96T:
                                ;bpb96t:    ; bpbx <512, 1, 1, 2, 224, 2400, 0F9h, 7, 15, 2, 0, 0, 0, 0>
                                            ; 96 tpi diskettes    ;
                                            dw     512            ; physical sector size in bytes
                                            db     1              ; sectors/allocation unit
                                            dw     1              ; reserved sectors for dos
                                            db     2              ; number of allocation tables
                                            dw     224            ; number of directory entries
                                            dw     2400 ; 2*15*80 ; number of sectors (at 512 bytes each)
                                            db     0F9h           ; media descriptor
                                            dw     7              ; number of fat sectors
                                            dw     15             ; sectors per track
                                            dw     2              ; heads
                                            dw     0              ; hidden sector count (low word)
                                            dw     0              ; hidden sector (high)
                                            dw     0              ; number of sectors (low)
                                            dw     0              ; number of sectors (high)
                                            ; 09/12/2023
                                            ; FAT32 extensions (to BDS)
```

```
                                        times  28 db 0
                                        ;
                                        db 90h
                                ;align 2
                                BPB35:
                                ;bpb35:         ; bpbx <512, 2, 1, 2, 112, 1440, 0F9h, 3, 9, 2, 0, 0, 0, 0>
                                        ; 3.5" diskettes - 720 KB ;
                                        dw      512             ; physical sector size in bytes
                                        db      2               ; sectors/allocation unit
                                        dw      1               ; reserved sectors for dos
                                        db      2               ; number of allocation tables
                                        dw      112             ; number of directory entries
                                        dw      1440 ; 2*9*80   ; number of sectors (at 512 bytes each)
                                        db      0F9h            ; media descriptor
                                        dw      3               ; number of fat sectors
                                        dw      9               ; sectors per track
                                        dw      2               ; heads
                                        dw      0               ; hidden sector count (low word)
                                        dw      0               ; hidden sector (high)
                                        dw      0               ; number of sectors (low)
                                        dw      0               ; number of sectors (high)
                                        ; 09/12/2023
                                        ; FAT32 extensions (to BDS)
                                        times  28 db 0
                                        ;
                                        db 90h
                                ;align 2

                                ;align 2
                                ;BPB144:
                                ;bpb144:    ; Retro DOS v4.0 feature only !      ; 1.44MB diskettes
                                ;
                                ;       dw      512             ; physical sector size in bytes
                                ;       db      1               ; sectors/allocation unit
                                ;       dw      1               ; reserved sectors for dos
                                ;       db      2               ; number of allocation tables
                                ;       dw      224             ; number of directory entries
                                ;       dw      2880 ; 2*18*80  ; number of sectors (at 512 bytes each)
                                ;       db      0F0h            ; media descriptor
                                ;       dw      9               ; number of fat sectors
                                ;       dw      18              ; sectors per track
                                ;       dw      2               ; heads
                                ;       dw      0               ; hidden sector count (low word)
                                ;       dw      0               ; hidden sector (high)
                                ;       dw      0               ; number of sectors (low)
                                ;       dw      0               ; number of sectors (high)
                                ;
                                ;       db 90h
                                ;align 2

                                BPB288:
                                ;bpb288:    ; bpbx <512, 2, 1, 2, 240, 5760, 0F0h, 9, 36, 2, 0, 0, 0, 0>
                                        ; 3.5" diskettes - 2.88 MB ;
                                        dw      512             ; physical sector size in bytes
                                        db      2               ; sectors/allocation unit
                                        dw      1               ; reserved sectors for dos
                                        db      2               ; number of allocation tables
                                        dw      240             ; number of directory entries
                                        dw      5760 ; 2*36*80  ; number of sectors (at 512 bytes each)
                                        db      0F0h            ; media descriptor
                                        dw      3               ; number of fat sectors
                                        dw      9               ; sectors per track
                                        dw      2               ; heads
                                        dw      0               ; hidden sector count (low word)
                                        dw      0               ; hidden sector (high)
                                        dw      0               ; number of sectors (low)
                                        dw      0               ; number of sectors (high)
                                        ; 09/12/2023
                                        ; FAT32 extensions (to BDS)
                                        times  28 db 0
                                        ;
                                        db 90h
                                ;align 2

                                %endif

                                ; ----------------------------------------------------------------------------
                                                                ; align 2
                                ; 09/12/2022
                                %if 0
                                bpbtable:   dw bpb48t           ; 48tpi drives
                                            dw bpb96t           ; 96tpi drives
                                            dw bpb35            ; 3.5" drives
                                            dw bpb35            ; unused 8" diskette
                                            dw bpb35            ; unused 8" diskette
                                            dw bpb35            ; used for hard       disk
                                            dw bpb35            ; used for tape       drive
                                            dw bpb35            ; FFOTHER
                                            dw bpb35            ; ERIMO
                                            dw bpb288           ; 2.88MB drive
                                            ;
                                            ;dw bpb144          ; 1.44MB drive - Retro DOS v4.0 feature !
                                %endif

                                ; 13/12/2022
                                %if 0
                                BPBTABLE:   dw BPB48T           ; 48tpi drives
                                            dw BPB96T           ; 96tpi drives
                                            dw BPB35            ; 3.5" drives
                                            dw BPB35            ; unused 8" diskette
                                            dw BPB35            ; unused 8" diskette
                                            dw BPB35            ; used for hard       disk
                                            dw BPB35            ; used for tape       drive
                                            dw BPB35            ; FFOTHER
                                            dw BPB35            ; ERIMO
                                            dw BPB288           ; 2.88MB drive
                                            ;
                                            ;dw BPB144          ; 1.44MB drive - Retro DOS v4.0 feature !

                                %endif

                                ; ----------------------------------------------------------------------------

                                ;   entry point to call utility functions in Bios_Code. At this time,
                                ;     we aren't doing any A20 switching. During MSINIT time Bios_Code
                                ;     will not yet be moved to its final resting place, so we know
                                ;     it'll be low.
                                ;
                                ;   to use this function, do a "push cs" and load bp with the offset of
                                ;     the function you want to call in Bios_Code. This routine will
                                ;     push the address of a retf in Bios_Code onto the stack which
                                ;     will get executed when the utility function finishes. It will
                                ;     then transfer control to Bios_Code:bp using a couple of pushes
                                ;     and a retf
```

```
                                ; 16/10/2022
                                ;BC_RETF equ bc_retf - DOSBIOSEG_2C7h
                                ; 09/12/2022
                                BC_RETF equ bc_retf

                                ; 09/12/2023
                                ;PCDOS 7.1 IBMBIO.COM bc_retf offset = 0CAh (in BIOSCODE segment = 364h)

                                addr_of_bcretf:    ;dw 0C8h              ; dw bc_retf
                                                                         ; 2C7h:0C8h = 70h:2638h
                                                                         ; 09/12/2023
                                                                         ; 364h:0CAh = 70h:300Ah ; PCDOS 7.1
00001A6F [CA00]                                 dw BC_RETF               ; dw 0CAh

                                ; ---------------------------------------------------------------------------

                                call_bios_code:    ; proc far
00001A71 2EFF36[6F1A]                           push   word [cs:addr_of_bcretf]
                                                                         ; set up near return to far return
00001A76 2EFF36[0406]                           push   word [cs:cdev+2] ; push Bios_Code segment
00001A7B 55                                     push   bp                ; save offset of utility function
00001A7C CB                                     retf                     ; far jump to (DOS)BIOS code

                                ; ---------------------------------------------------------------------------

                                                ; 09/12/2023 - Retro DOS v5.0 IBMBIO.COM/IO.SYS
                                                ; 20/12/2022
00001A7D 00                     flp_drvs:   db 0
                                                ; 11/12/2023
                                                ; (PCDOS 7.1 IBMBIO.COM - BIOSDATA:1B81h)
                                firstcluster_hw:
00001A7E 0000                                   dw 0    ; 06/04/2024
00001A80 00                     Boot_Drv:   db 0

                                ; ---------------------------------------------------------------------------

                                ; 09/12/2023 - Retro DOS v5.0 (PCDOS 7.1) IBMBIO.COM
                                ;---------------------------------------------------------------------------
                                ; PCDOS 7.1 CD BOOT option code
                                ;---------------------------------------------------------------------------
                                ; (PCDOS 7.1 IBMBIO.COM, BIOSDATA:1B84h)

                                cd_boot_option:
00001A81 50                                     push   ax
00001A82 1E                                     push   ds
00001A83 06                                     push   es
00001A84 52                                     push   dx
                                cdbo_1:
00001A85 B401                                   mov    ah, 1
00001A87 CD16                                   int    16h                ; KEYBOARD - status
00001A89 7406                                   jz     short cdbo_2
00001A8B 30E4                                   xor    ah, ah
00001A8D CD16                                   int    16h                ; KEYBOARD - READ CHAR FROM BUFFER, WAIT IF EMPTY
                                                                          ; Return: AH = scan code, AL = character
00001A8F EBF4                                   jmp    short cdbo_1
                                cdbo_2:
00001A91 0E                                     push   cs
00001A92 1F                                     pop    ds
00001A93 BE[6B1B]                               mov    si, cd_boot_msg           ; "Press the ENTER key to boot from CD"...
00001A96 AC                                     lodsb
                                cdbo_3:
00001A97 BB0700                                 mov    bx, 7
00001A9A B40E                                   mov    ah, 0Eh
00001A9C CD10                                   int    10h                ; - VIDEO - WRITE CHARACTER AND ADVANCE CURSOR (TTY WRITE)
                                                                          ; AL = character, BH = display page (alpha modes)
                                                                          ; BL = foreground color (graphics modes)
00001A9E AC                                     lodsb
00001A9F 08C0                                   or     al, al
00001AA1 75F4                                   jnz    short cdbo_3
00001AA3 B84000                                 mov    ax, 40h
00001AA6 8ED8                                   mov    ds, ax
                                                ;mov   bx, [6Ch]          ; 0:46Ch = Daily timer counter (4 bytes)
                                                ; 09/12/2023
00001AA8 8B166C00                               mov    dx, [6Ch]
00001AAC 8B366E00                               mov    si, [6Eh]
                                wait_for_key:
                                                ;push  bx
                                                ;mov   bx, 7
                                                ; bx = 7
00001AB0 B8080E                                 mov    ax, 0E08h
00001AB3 CD10                                   int    10h                ; - VIDEO - WRITE CHARACTER AND ADVANCE CURSOR (TTY WRITE)
                                                                          ; AL = character, BH = display page (alpha modes)
                                                                          ; BL = foreground color (graphics modes)
00001AB5 B8200E                                 mov    ax, 0E20h
00001AB8 CD10                                   int    10h                ; - VIDEO - WRITE CHARACTER AND ADVANCE CURSOR (TTY WRITE)
                                                                          ; AL = character, BH = display page (alpha modes)
                                                                          ; BL = foreground color (graphics modes)
00001ABA B8080E                                 mov    ax, 0E08h
00001ABD CD10                                   int    10h                ; - VIDEO - WRITE CHARACTER AND ADVANCE CURSOR (TTY WRITE)
                                                                          ; AL = character, BH = display page (alpha modes)
                                                                          ; BL = foreground color (graphics modes)
                                                ;pop   bx
                                                ;add   bx, 18             ; 18.2 ticks per second
                                                ; 09/12/2023
00001ABF 83C212                                 add    dx, 18
00001AC2 83D600                                 adc    si, 0              ; next second (if carry flag is 1)
                                continue_to_wait:
00001AC5 B401                                   mov    ah, 1
00001AC7 CD16                                   int    16h                ; KEYBOARD - status
00001AC9 741B                                   jz     short cdbo_5
00001ACB B400                                   mov    ah, 0
00001ACD CD16                                   int    16h                ; KEYBOARD - READ CHAR FROM BUFFER, WAIT IF EMPTY
                                                                          ; Return: AH = scan code, AL = character

                                                ; 09/12/2023
                                                ;cmp   ax, 11Bh ; ESC key
                                                ;jz    short cdb0_7
                                ;cdbo_4:
                                                ;push  ax ; *
00001ACF 89C2                                   mov    dx, ax ; *

                                                ; CRLF (next line)
                                                ;mov   bx, 7
                                                ; bx = 7
00001AD1 B80D0E                                 mov    ax, 0E0Dh
00001AD4 CD10                                   int    10h                ; - VIDEO - WRITE CHARACTER AND ADVANCE CURSOR (TTY WRITE)
                                                                          ; AL = character, BH = display page (alpha modes)
                                                                          ; BL = foreground color (graphics modes)
00001AD6 B80A0E                                 mov    ax, 0E0Ah
00001AD9 CD10                                   int    10h                ; - VIDEO - WRITE CHARACTER AND ADVANCE CURSOR (TTY WRITE)
                                                                          ; AL = character, BH = display page (alpha modes)
                                                                          ; BL = foreground color (graphics modes)
                                                ; 09/12/2023
```

```
                                        ;pop    ax ; *
                                        cmp     dx, 11Bh
00001ADB 81FA1B01                       ;cmp    ax, 11Bh ; ESC key (to cancel CD/DVD boot)
00001ADF 7418                           je      short cdbo_7
                                cdbo_4:         ; 10/12/2023
00001AE1 5A                             pop     dx
00001AE2 07                             pop     es
00001AE3 1F                             pop     ds
00001AE4 58                             pop     ax
00001AE5 C3                             retn
                                cdbo_5:
00001AE6 3B366E00                       cmp     si, [6Eh]
00001AEA 7504                           jnz     short cdbo_6
                                        ; 09/12/2023
00001AEC 3B166C00                       cmp     dx, [6Ch]
                                        ;cmp    bx, [6Ch]
                                cdbo_6:
00001AF0 73D3                           jnb     short continue_to_wait
00001AF2 2EFE0E[6A1B]                   dec     byte [cs:time_counter]
00001AF7 75B7                           jnz     short wait_for_key
                                cdbo_7:
                                        ; 09/12/2023
                                        ; CRLF (next line)
                                        ;
                                        ;mov    bx, 7
                                        ;mov    ax, 0E0Dh
                                        ;int    10h                     ; - VIDEO - WRITE CHARACTER AND ADVANCE CURSOR (TTY WRITE)
                                        ;                               ; AL = character, BH = display page (alpha modes)
                                        ;                               ; BL = foreground color (graphics modes)
                                        ;mov    ax, 0E0Ah
                                        ;int    10h                     ; - VIDEO - WRITE CHARACTER AND ADVANCE CURSOR (TTY WRITE)
                                        ;                               ; AL = character, BH = display page (alpha modes)
                                        ;                               ; BL = foreground color (graphics modes)
00001AF9 0E                             push    cs
00001AFA 1F                             pop     ds
                                        ; 09/12/2023
00001AFB 1E                             push    ds
00001AFC 07                             pop     es
                                        ; es = ds = cs
00001AFD B8004B                         mov     ax, 4B00h
                                        ;xor    dl, dl
                                        ; 09/12/2023
00001B00 31D2                           xor     dx, dx
                                        ; dl = disk drive = 0  ; fd
                                        ;mov    si, 1C93h
00001B02 BE[571B]                       mov     si, empty_dap_buff
00001B05 CD13                           int     13h                     ; DISK - Bootable CD-ROM - AL = TERMINATE DISK EMULATION
                                        ; DS:SI = Specification packet filled
                                        ;mov    dx, 80h
                                        ;xor    ax, ax
                                        ; 09/12/2023
00001B07 B81300                         mov     ax, 19
00001B0A 89F7                           mov     di, si
                                        ;mov    byte [si], 13h
                                        ;mov    [si+1], al
00001B0C AB                             stosw
                                        ;mov    [si+2], dx
00001B0D B080                           mov     al, 80h
00001B0F AB                             stosw
00001B10 89C2                           mov     dx, ax
                                        ;mov    [si+4], ax
                                        ;mov    [si+6], ax
                                        ;mov    [si+8], ax
                                        ;mov    [si+0Ah], ax
                                        ;mov    [si+0Ch], ax
                                        ;mov    [si+0Eh], ax
                                        ;mov    [si+10h], al
                                        ;mov    [si+11h], al
                                        ;mov    [si+12h], al
00001B12 B90F00                         mov     cx, 15
00001B15 F3AA                           rep     stosb
                                        ; dl = disk drive = 80h ; hd
00001B17 B8004B                         mov     ax, 4B00h
00001B1A CD13                           int     13h                     ; DISK - Bootable CD-ROM - AL = TERMINATE DISK EMULATION
00001B1C 31C0                           xor     ax, ax
                                        ; 09/12/2023
                                        ;mov    dx, 80h
                                        ; dx = 80h
00001B1E CD13                           int     13h                     ; DISK - RESET DISK SYSTEM
                                                                        ; DL = drive (if bit 7 is set both hard disks and floppy disks reset)
                                        ; 09/12/2023
                                        ;push   cs
                                        ;pop    es
                                        ; es = ds = cs
00001B20 B80102                         mov     ax, 201h
                                        ;mov    bx, 152h
00001B23 BB[5201]                       mov     bx, disksector
                                        ;mov    cx, 1
                                        ; 09/12/2023
00001B26 41                             inc     cx ; cx = 1
                                        ;mov    dx, 80h
                                        ; dx = 80h
00001B27 CD13                           int     13h                     ; DISK - READ SECTORS INTO MEMORY
                                                                        ; AL = number of sectors to read, CH = track, CL = sector
                                                                        ; DH = head, DL = drive, ES:BX -> buffer to fill
                                                                        ; Return: CF set on error, AH = status, AL = number of sectors read
                                        ;jc     short cdbo_8
                                        ; 10/12/2023
00001B29 72B6                           jc      short cdbo_4

00001B2B 2681BFFE0155AA                 cmp     word [es:bx+1FEh], 0AA55h
                                        ;jz     short cdbo_9
                                        ; 10/12/2023
00001B32 75AD                           jnz     short cdbo_4
                                ;cdbo_8:
                                        ;jmp    short cdbo_4
                                ;cdbo_9:
                                        ; 10/12/2023
                                        ; (stack clearing -pop- is not necessary here,
                                        ;  PCDOS 7.1 boot sector will set stack pointer again)
                                        ;pop    ax ; near call return address
                                        ;pop    cx ; +++ ; ch = [MediaByte]

                                        ; 09/12/2023
                                        ;push   cs
                                        ;pop    ds
                                        ; ds = cs
00001B34 31C0                           xor     ax, ax  ; 0
```

```
4850 00001B36 BF007C                            mov     di, 7C00h
4851 00001B39 8EC0                              mov     es, ax
4852 00001B3B 89DE                              mov     si, bx
4853 00001B3D 06                                push    es
4854 00001B3E 57                                push    di
4855 00001B3F B90001                            mov     cx, 100h ; 256
4856                                            ; 10/12/2023
4857                                            ;cld    ; not necessary (direction flag is already cleared)
4858 00001B42 F3A5                              rep movsw
4859 00001B44 8ED8                              mov     ds, ax
4860 00001B46 BE7800                            mov     si, 78h
4861 00001B49 2EA1[121A]                        mov     ax, [cs:Orig_Int1Eh_Table]
4862 00001B4D 8904                              mov     [si], ax
4863 00001B4F 2EA1[141A]                        mov     ax, [cs:Orig_Int1Eh_Table+2]
4864 00001B53 894402                            mov     [si+2], ax
4865 00001B56 CB                                retf
4866
4867                                    ; ---------------------------------------------------------------------------
4868                                    dap_buffer: ; 16/12/2023
4869
4870 00001B57 13                        empty_dap_buff:     db 19
4871                                                ;db 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0  ; db 18 dup(0)
4872 00001B58 00<rep 12h>                           times 18 db 0
4873 00001B6A 05                        time_counter:       db 5    ; 5 seconds
4874 00001B6B 0D0A                      cd_boot_msg:        db 0Dh,0Ah
4875                                                ;db 'Press the ENTER key to boot from CD or DVD......',0
4876                                                ; 09/12/2023
4877 00001B6D 507265737320616E79-                   db 'Press any key to boot from CD or DVD ...',0
4877 00001B76 206B657920746F2062-
4877 00001B7F 6F6F742066726F6D20-
4877 00001B88 4344206F7220445644-
4877 00001B91 202E2E2E00
4878
4879                                    ; ---------------------------------------------------------------------------
4880
4881                                    ; 10/12/2023 - Retro DOS v5.0 (PCDOS 7.1)
4882
4883                                    ; 01/10/2022 - Retro DOS v4.0 (MSDOS 5.0, classic/old MICROSOFT DOS method)
4884                                    ; 20/12/2022 - Retro DOS v4.0 (combined kernel files, original/new method) (*)
4885                                    ;      (*) (for using Retro DOS kernel 'MSDOS.SYS' with Retro DOS boot sector)
4886
4887                                    ;--------------------------------------------------------------------------
4888                                    ; entry point from boot sector
4889                                    ;--------------------------------------------------------------------------
4890
4891                                    init:               ; 27/12/2018
4892                                                ; MSDOS 6.0 (MSINIT.ASM)
4893                                                ;============================================================
4894                                                ;
4895                                                ; entry from boot sector. the register contents are:
4896                                                ;
4897                                                ;   dl = int 13 drive number we booted from
4898                                                ;   ch = media byte
4899                                                ;   bx = first data sector on disk.
4900                                                ;   ax = first data sector (high)
4901                                                ;   di = sectors/fat for the boot media.
4902
4903                                                ; 10/12/2023
4904                                                ; Retro DOS v5.0 (IBMBIO.COM)
4905                                                ;============================================================
4906                                                ; PCDOS 7.1 IBMBIO.COM - registers from MSLOAD section
4907                                                   ; DL = [BootDrive]
4908                                                ; CH = [MediaByte]
4909                                                ; AX:BX = First data Sector
4910                                                ; DS:SI = Original INT 1Eh table address
4911                                                ;
4912                                                ; Stack: INT 1Eh vector (0:78h) !not used! (dword [sp])
4913                                                ;        INT 1Eh table address !not used! (dword [sp+4])
4914                                                ; DI = 78h !not used!
4915
4916                                                ; 07/04/2018
4917                                                ;============================================================
4918                                                ; Retro DOS v2.0 - registers from FD Boot Sector
4919                                                   ; DL = [bsDriveNumber]
4920                                                ; DH = [bsMedia]
4921                                                ; AX = [bsSectors] ; Total sectors
4922                                                ; DS = 0, SS = 0
4923                                                ; BP = 7C00h
4924
4925                                                ; 29/09/2023
4926                                                ; SP = 0FFFEh (for Retro DOS v2&v3 boot sector)
4927                                                ;    = 07C00h (for MSDOS 5.0 boot sector)
4928
4929                                                ; 10/12/2023 - Retro DOS v5.0
4930                                                ; ------------------------------------------------------------
4931                                                ; INPUT (registers from Retro DOS v4-v5 boot sector):
4932                                                ;  DL = [bsDriveNumber]
4933                                                ;  DH = [bsMedia]
4934                                                ;  SS = 0
4935                                                ;  BP = 7C00h (boot sector address)
4936                                                ;
4937                                                ; If the boot drive is a CD (CDROM) or DVD
4938                                                ;    and CD boot option is enabled/requested:
4939                                                ;    AX = 'CD'
4940                                                ; If the boot drive is a FD or HD
4941                                                ;    or CD boot option is not enabled/requested:
4942                                                ;    AX <> 'CD'
4943
4944                                    ; 20/12/2022
4945                                    ; Changing original MSDOS 5.0 IO.SYS init code with Retro DOS v4.0 init code.
4946                                    %if 0
4947                                                cli
4948
4949                                                push    ax
4950                                                xor     ax, ax
4951                                                mov     ds, ax
4952                                                pop     ax
4953                                    %endif
4954
4955                                    ; 20/12/2022 - Retro DOS v4.0 (combined kernel)
4956                                    ; 10/12/2023 - Retro DOS v5.0 (combined kernel)
4957
4958                                    KERNEL_SEGMENT equ 70h     ; (DOS BIOSDATA SEGMENT)
4959                                    BSSECPERTRACK equ 18h      ; boot sector offset 18h (for Retro DOS & MSDOS)
4960
4961                                    ;--------------------------------------------------------------------------
4962                                    ; initialization - stage 1
4963                                    ;--------------------------------------------------------------------------
4964                                    ; 02/06/2018 - Retro DOS v3.0
4965
4966                                                ; 10/12/2023
4967 00001B96 FC                                   cld     ; may not be necessary
4968
4969                                                ; 21/12/2022
```

```
                                        ; Move Retro DOS v2.0 boot sector parameters to 0060h:0
                                        ;mov    bx, 60h
                                        ;mov    es, bx
                                        ;mov    si, bp
                                        ;sub    di, di
                                        ;mov    cx, 35 ; 70 bytes, 35 words
                                        ;;mov   cl, 35
                                        ;rep    movsw

                                        ; 10/12/2023 - Retro DOS v5.0
00001B97 3D4344                         cmp     ax, 'CD' ; is CD boot option enabled or not ?
00001B9A 7503                           jne     short init0

00001B9C E8E2FE                         call    cd_boot_option
                                init0:
00001B9F 0E                             push    cs
00001BA0 1F                             pop     ds

                                        ; 10/12/2023 - Retro DOS v5.0 IO.SYS/IBMBIO.COM
                                        ;mov    [Boot_Drv], dl

                                        ; 20/03/2019 - Retro DOS v4.0
                                        ;cli            ; turn interrupts off while manupulating stack
                                        ;mov    ss, cx  ; set stack segment register

00001BA1 BC0007                         mov     sp, 0700h ; move stack pointer to safe place

                                        ;sti            ; turn interrupts on

                                        ; 27/03/2018
                                        ;mov    cx, KERNEL_SIZE       ; words !

                                        ; 20/03/2019
00001BA4 B90080                         mov     cx, 32768 ; 65536 bytes

                                        ; 21/12/2022
                                        ; 07/04/2018
00001BA7 BB7000                         mov     bx, KERNEL_SEGMENT ; 0070h
                                        ;mov    bl, KERNEL_SEGMENT
00001BAA 8EC3                           mov     es, bx
00001BAC 31FF                           xor     di, di
00001BAE 89FE                           mov     si, di

                                        ; Move KERNEL file from 1000h:0 to 0070h:0
                                        ; (Retro DOS v2 BS loads 'MSDOS.SYS' at 1000h:0000h)
00001BB0 F3A5                           rep     movsw

                                        ; 20/03/2019 - Retro DOS v4.0
00001BB2 53                             push    bx
                                        ;push   init0
00001BB3 68[B71B]                       push    init1   ; 10/12/2023
00001BB6 CB                             retf
                                ;init0:
                                        ; 10/12/2023 - Retro DOS 5.0
                                init1:
                                        ; 20/12/2022
                                        ; (combined kernel file > 64KB)

                                        ; 20/03/2019
00001BB7 B520                           mov     ch, 20h
00001BB9 8ED9                           mov     ds, cx ; 2000h
                                        ;mov    cx, 1070h
00001BBB B97010                         mov     cx, KERNEL_SEGMENT+1000h ; 20/12/2022
00001BBE 8EC1                           mov     es, cx

                                        ; 21/12/2022
                                        ;KERNEL_SIZE equ END_OF_KERNEL - BData_start
                                        ; 28/09/2023
                                        NXWORDCOUNT equ ((KERNEL_SIZE+1)>>1)-32768

                                        ;mov    cx, KERNEL_SIZE - 32768
                                        ; 28/09/2023 (BugFix)
00001BC0 B9E31E                         mov     cx, NXWORDCOUNT
                                        ;mov    cx, NXBYTECOUNT
                                        ;shr    cx, 1 ; 28/09/2023
                                        ;xor    si, si
                                        ;xor    di, di
00001BC3 F3A5                           rep     movsw

                                        ; 28/09/2023
                                        ;; 17/06/2018
                                        ;mov    ds, bx
                                        ;; 21/03/2019
                                        ;mov    es, bx
                                ;init0:
                                ;               ;push   es
                                ;               push    bx ; 20/03/2019
                                ;               push    init1 ; 07/04/2018
                                ;               retf    ; jump to 0070h:init1
                                ;init:
                                ;init1:
                                        ; 10/12/2023
                                init2:
                                        ; 20/12/2022
                                        ; Change INT 1Eh diskette parameters table and INT 1Eh address
                                        ; for full MSDOS compatibility.

                                        ; 10/12/2023
                                        ;cli    ; not necessary for INT 1Eh

00001BC5 8EC1                           mov     es, cx ; 0
00001BC7 8ED9                           mov     ds, cx ; 0

00001BC9 B82205                         mov     ax, SEC9

                                        ;mov    bx, 1Eh*4  ; [0078h] ; INT 1Eh vector/pointer
00001BCC B378                           mov     bl, 1Eh*4
                                                ; INT 1Eh points to diskette parms table

                                        ; check if the table is already at 0:SEC9 (0:0522h)
                                        ; (do not move the DPT if is not original ROMBIOS table)

                                        ;;or    [bx+2],cx [(1Eh*4)+2] ; [007Ah] ; segment
                                        ;;jnz   short mov_dpt

                                        ;cmp    ax, [bx]  ; [1Eh*4] = 0522h ?
                                        ;je     short dont_mov_dpt

                                        ;mov    si, [bx] ; [1Eh*4]
                                ;mov_dpt:
                                        ;mov    ds, [bx+2] ; [(1Eh*4)+2] ; [007Ah] ; segment
00001BCE C537                           lds     si, [bx]

                                        ; 10/12/2023 - Retro DOS v5.0
```

```
                                        ;mov    [cs:Orig_Int1Eh_Table+2], ds
                                        ;mov    [cs:Orig_Int1Eh_Table], si

00001BD0 89C7                           mov     di, ax  ; SEC9
00001BD2 B10B                           mov     cl, 11
                                        ;cld
00001BD4 F3A4                           rep     movsb

                                        ; Set INT 1Eh vector/pointer to the new DPT address
00001BD6 8ED9                           mov     ds, cx ; 0
00001BD8 8907                           mov     [bx], ax ; SEC9       ; [007Eh] ; 1Eh*4  ; offset
00001BDA 894F02                         mov     [bx+2], cx ; 0  ; [007Ah] ; 1Eh*4+2 ; segment
                                ;dont_mov_dpt:

                                ; 20/12/2022 - Retro DOS v4.0
                                %if 0
                                        ; 27/12/2018 - Retro DOS v4.0
                                        ; 'Starting MS-DOS...' message
                                        ; (MSDOS 6.21, IO.SYS Segment: 423h, Offset: 5673h)
                                        ; (0070h:96A3h)

                                        mov     si, SYSINIT_START+StartMsg ; 18/03/2019
                                        mov     ah, 0Eh
                                        ;bh = 0
                                        mov     bl, 7         ; "normal" attribute and page
                                startmsg_nxt_chr:
                                        lodsb
                                        or      al, al
                                        jz      short startmsg_ok

                                        int     10h           ; video write
                                        jmp     short startmsg_nxt_chr

                                ;flp_drvs:  db  0   ; 27/12/2018 - Retro DOS v4.0

                                startmsg_ok:

                                %endif

                                ;--------------------------------------------------------------------------
                                ; initialization - stage 2
                                ;--------------------------------------------------------------------------
                                ; 20/12/2022 - Retro DOS v4.0 (combined kernel)


                                ; 19/03/2018
                                ; Retro DOS v2.0 (24/02/2018)
                                ; [REF: MSDOS 3.3, MSBIO, "MSINIT.ASM"  (24/07/1987)]

                                ;----------------------------------------------------------------------
                                ;                                                      :
                                ;   System initialization                              :
                                ;                                                      :
                                ;   The entry conditions are established by the bootstrap            :
                                ;   loader and are considered unknown. The following jobs            :
                                ;   will be performed by this module:                  :
                                ;                                                      :
                                ;   1.      All device initialization is performed                   :
                                ;   2.      A local stack is set up and DS:SI are set  :
                                ;           to point to an initialization table. Then  :
                                ;           an inter-segment call is made to the first :
                                ;           byte of the dos                            :
                                ;   3.      Once the dos returns from this call the ds :
                                ;           register has been set up to point to the start           :
                                ;           of free memory. The initialization will then       :
                                ;           load the command program into this area    :
                                ;           beginning at 100 hex and transfer control to       :
                                ;           this program.                              :
                                ;                                                      :
                                ;----------------------------------------------------------------------

                                        ; 20/12/2022
                                        ; ----------------------
                                        ; Registers
                                        ; ----------------------
                                        ; DL = [bsDriveNumber]
                                        ; DH = [bsMedia]
                                        ; DS = 0, ES = 0, SS = 0
                                        ; BP = 7C00h
                                        ; SP = 700h
                                        ; ----------------------
                                        ; CX = 0

                                ; 02/10/2022 - 20/12/2022
                                ; ------------------------------------------------------------------------------
                                ; Note: Retro DOS v4.0 Kernel does not use/contain MSLOAD part of IO.SYS (5.0)
                                ;   Because, Retro DOS v2 boot sector loads complete/entire MSDOS.SYS
                                ;   (RETRODOS.SYS) Kernel file (IO.SYS & MSDOS.SYS together).
                                ;   As result of boot sector ve init differences, Retro DOS init code (here)
                                ;   moves kernel to segment 70h at first, then sets diskette parameters
                                ;   at segment 50h (while MSDOS 5.0 boot sector and then MSLOAD sets this).
                                ; ------------------------------------------------------------------------------

                                ; msload will check the extended boot record and set ax, bx accordingly.
                                ;
                                ;;  msload passes a 32 bit sector number hi word in ax and low in bx
                                ;;  save this in cs:bios_h and cs:bios_l. this is for the start of
                                ;;  data sector of the bios.
                                ;
                                ;           mov     [cs:bios_h], ax       ; (start of) dos bios (IO.SYS) data sector
                                ;           mov     [cs:bios_l], bx

                                ; with the following information from msload, we don't need the
                                ;     boot sector any more.-> this will solve the problem of 29 kb size
                                ;     limitation of msbio.com file.

                                        ; 10/12/2023
                                        ; 21/12/2022
                                        ;cli

00001BDD 0E                             push    cs              ; Save a peck of interrupt vectors...
00001BDE 07                             pop     es
                                        ;push   cx
                                        ;push   di

                                        ; 20/12/2022
00001BDF B105                           mov     cl, 5
                                        ;mov    cx, 5           ; NUMROMVECTORS
                                                                ; no. of rom vectors to      be saved
                                        ;mov    si, offset RomVectors ; point to list of int vectors
00001BE1 BE[0001]                       mov     si, RomVectors

                                        ; 10/12/2023
00001BE4 FA                             cli
```

```
                                next_int_:
00001BE5 2E                             cs      ; 16/10/2022
00001BE6 AC                             lodsb
                                        ;lods   byte ptr cs:[si] ; cs lodsb
00001BE7 98                             cbw                     ; ax = interrupt number
00001BE8 D1E0                           shl     ax, 1
00001BEA D1E0                           shl     ax, 1           ; int no * 4
00001BEC 89C7                           mov     di, ax          ; interrupt vector address
00001BEE 87F7                           xchg    si, di          ; rombios interrupt vector address in si
                                                                ; saving address in di
                                        ;lodsw                  ; movsw
                                        ;stosw
                                        ;lodsw                  ; movsw
                                        ;stosw                  ; save the vector
                                        ; 20/12/2022
00001BF0 A5                             movsw
00001BF1 A5                             movsw

00001BF2 87F7                           xchg    si, di
00001BF4 E2EF                           loop    next_int_

                                        ;pop    di
                                        ;pop    cx

                                ; we need to save int13 in two places in case we are running on an at.
                                ; on ats we install the ibm supplied rom_bios patch which hooks
                                ; int13 ahead of orig13. since int19 must unhook int13 to point to the
                                ; rom int13 routine, we must have that rom address also stored away.

                                        ; 20/12/2022
                                        ;mov    ax, [cs:Old13] ; save old13 in orig13 also
                                        ;mov    [cs:Orig13], ax
                                        ;mov    ax, [cs:Old13+2]
                                        ;mov    [cs:Orig13+2], ax

                                        ; 10/12/2023
                                        ;cli
                                        ; 16/10/2022
00001BF6 C7064C00[ED06]                 mov     word [13h*4], block13
                                        ;mov    word ptr ds:4Ch, offset      block13 ; 13h*4
                                                                ; set up int 13       for new action
00001BFC 8C0E4E00                       mov     [13h*4+2], cs
                                        ;mov    word ptr ds:4Eh, cs ; 13h*4+2
00001C00 C7065400[9907]                 mov     word [15h*4], Int15
                                        ;mov    word ptr ds:54h, offset      Int15 ; 15h*4
                                                                ; set up int 15       for new action
00001C06 8C0E5600                       mov     [15h*4+2], cs
                                        ;mov    word ptr ds:56h, cs ; 15h*4+2
00001C0A C7066400[5907]                 mov     word [19h*4], int19
                                        ;mov    word ptr ds:64h, offset      int19 ; 19h*4
                                                                ; set up int 19       for new action
00001C10 8C0E6600                       mov     [19h*4+2], cs
                                        ;mov    word ptr ds:66h, cs ; 19h*4+2

                                        ; 10/12/2023 - Retro DOS v5.0 (PCDOS 7.1 IBMBIO.COM)
00001C14 A16800                         mov     ax, [68h]       ; 1Ah*4
00001C17 8B3E6A00                       mov     di, [6Ah]       ; 1Ah*4+2
00001C1B C7066800[AF06]                 mov     word [68h], Int1A
00001C21 8C0E6A00                       mov     [6Ah], cs

                                        ; 20/12/2022
00001C25 0E                             push    cs
00001C26 1F                             pop     ds

                                        ; 10/12/2023
00001C27 A3[AB06]                       mov     [Orig1A], ax
00001C2A 893E[AD06]                     mov     [Orig1A+2], di

00001C2E A1[0601]                       mov     ax, [Old13]     ; save old13 in orig13 also
00001C31 A3[B400]                       mov     [Orig13], ax
00001C34 A1[0801]                       mov     ax, [Old13+2]
00001C37 A3[B600]                       mov     [Orig13+2], ax
                                                                ; ;
00001C3A FB                             sti
00001C3B CD11                           int     11h             ; EQUIPMENT DETERMINATION
                                                                ; Return: AX = equipment flag bits
                                        ; 10/12/2023
                                        ;jmp    short chk_fd_count
                                        ; (PCDOS 7.1 IBMBIO.COM, BIOSDATA:1DF7h) ; *!!*
                                        ; ((signature))
                                        ;push   dx              ; 52h ; 'R'
                                        ;push   ax              ; 50h ; 'P'
                                        ;push   bx              ; 53h ; 'S'

                                ; we have to support a system that does not have any diskette
                                ; drives but only hardfiles. this system will ipl from the hardfile.
                                ; if the equipment flag bit 0 is 1, then the system has diskette drive(s).
                                ; otherwise, the system has only hardfiles.
                                ;
                                ; important thing is that still, for compatibility reason, the drive letter
                                ; for the hardfiles start from "c".  so, we still need to allocate dummy bds
                                ; drive a and drive b. at sysinit time, we are going to set cds table entry
                                ; of dpb pointer for these drives to 0, so any user attempt to access this
                                ; drives will get "invalid drive letter ..." message. we are going to
                                ; establish "fakefloppydrv" flag. ***sysinit module should call int 11h to
                                ; determine whether there are any diskette drivers in the system or not.!!!***

                                ; check the register returned by the equipment determination interrupt
                                ; we have to handle the case of no diskettes in the system by faking
                                ; two dummy drives.
                                ;
                                ; if the register indicates that we do have floppy drives we don't need
                                ; to do anything special.
                                ;
                                ; if the register indicates that we don't have any floppy drives then
                                ; what we need to do is set the fakefloppydrv variable, change the
                                ; register to say that we do have floppy drives and then go to execute
                                ; the code which starts at notsingle. this is because we can skip the
                                ; code given below which tries to find if there are one or two drives
                                ; since we already know about this.

                                chk_fd_count:   ; 10/12/2023
                                        ;or     ax, 1   ; *!!*

                                        ; 06/05/2019 - Retro DOS v4.0
00001C3D 88C1                           mov     cl, al

                                        ; 12/12/2022
00001C3F A801                           test    al, 1
                                        ;test   ax, 1           ; floppy drives       present?
00001C41 751E                           jnz     short normalfloppydrv ;       yes.

                                ; Some ROM BIOSs lie that there are no floppy drives. Lets find out
```

```
                                ; whether it is an old ROM BIOS or a new one
                                ;
                                ; WARNING !!!
                                ;
                                ; This sequence of code is present in SYSINIT1.ASM also. Any modification
                                ; here will require an equivalent modification in SYSINIT1.ASM also

                                        ; 20/12/2022
                                        ;push   ax
                                        ;push   bx
                                        ;push   cx
00001C43 52                             push    dx
                                        ;push   di
00001C44 06                             push    es

00001C45 B408                           mov     ah, 8
00001C47 B200                           mov     dl, 0
00001C49 CD13                           int     13h             ; DISK - DISK -       GET CURRENT DRIVE PARAMETERS (XT,AT,XT286,CONV,PS)
                                                                ; DL = drive number
                                                                ; Return: CF set on error, AH =     status code, BL       = drive type
                                                                ; DL = number of consecutive drives
                                                                ; DH = maximum value for head number, ES:DI -> drive parameter
00001C4B 7202                           jc      short _gdskp_error
                                        ;;mov   [cs:flp_drvs], dl
                                        ; 20/12/2022
                                        ; ds = cs
                                        ;mov    [flp_drvs], dl
00001C4D 88D1                           mov     cl, dl
                                _gdskp_error:
                                        ; 20/12/2022
00001C4F 07                             pop     es
                                        ;pop    di
00001C50 5A                             pop     dx
                                        ;pop    cx
                                        ;pop    bx
                                        ;pop    ax

00001C51 720E                           jc      short normalfloppydrv
                                                                ; if error it is an old ROM BIOS
                                                                ; so, lets assume that ROM BIOS lied
                                        ; 20/12/2022
                                        ; ds = cs
                                        ;cmp    byte [flp_drvs], 0
                                        ;;cmp   byte [cs:flp_drvs], 0 ; number of drvs == 0?
                                        ;jz     short _set_fake_flpdrv
                                        ;;mov   al, [cs:flp_drvs]
                                        ;mov    al, [flp_drvs]
                                        ;;dec   al              ; make it zero based
                                        ;; 18/12/2022
                                        ;dec    ax
                                        ;jmp    short got_num_flp_drvs

                                        ; 20/12/2022
00001C53 08C9                           or      cl, cl ; [flp_drvs]
00001C55 7403                           jz      short _set_fake_flpdrv
00001C57 49                             dec     cx
00001C58 EB0B                           jmp     short got_num_flp_drvs
                                ; ---------------------------------------------------------------------------

                                _set_fake_flpdrv:
                                        ; 20/12/2022
                                        ; ds = cs
                                        ;inc    cl      ; cl = 1
                                        ; 10/12/2023
00001C5A 41                             inc     cx      ; cl = 1
00001C5B 880E[111A]                     mov     [fakefloppydrv], cl ; 1
                                        ;mov    byte [fakefloppydrv], 1
                                        ;;mov   byte [cs:fakefloppydrv], 1
                                                                ; we don't have any floppy drives.
                                        ; 20/12/2022
                                        ;mov    ax, 1
00001C5F EB0A                           jmp     short settwodrive ; well then set it for two drives!
                                ; ---------------------------------------------------------------------------

                                normalfloppydrv:                ; yes, bit 0 is 1.
                                        ; 20/12/2022
                                        ;rol    al, 1           ; there exist floppy drives.
                                        ;rol    al, 1           ; put bits 6 & 7 into bits 0 & 1
00001C61 D0C1                           rol     cl, 1
00001C63 D0C1                           rol     cl, 1
                                got_num_flp_drvs:
                                        ;;and   ax, 3           ; only look at bits 0 &       1
                                        ; 18/12/2022
                                        ;and    al, 3
                                        ; 20/12/2022
00001C65 80E103                         and     cl, 3
00001C68 7505                           jnz     short notsingle       ; zero means single drive system
                                        ; 20/12/2022
00001C6A 41                             inc     cx
                                        ;inc    ax              ; pretend it's a two drive system
                                settwodrive:                            ; set this to two fakedrives
                                        ; 20/12/2022
                                        ; ds = cs
00001C6B FE06[7800]                     inc     byte [single]
                                        ;inc    byte [cs:single] ; remember this
                                notsingle:
                                        ; 20/12/2022
                                        ;inc    ax              ; ax has number       of drives, 2-4
                                        ;                       ; is also 0 indexed boot drive if we
                                        ;                       ; booted off hard file
                                        ;mov    cl, al          ; ch is fat id, cl # floppies

                                        ; 20/12/2022
                                        ;inc    cl      ; cl >= 2
                                        ; 10/12/2023
00001C6F 41                             inc     cx      ; cl >= 2

                                ; 16/10/2022
                                ; MSDOS 3.3 - "MSEQU.INC" (24/07/1987)
                                INITSPOT EQU        534h    ; IBM wants 4 zeros here
                                BRKADR       EQU    1BH * 4 ; 6CH, 1BH break vector address
                                TIMADR      EQU     1CH * 4 ; 70H, 1CH timer interrupt
                                DSKADR      EQU     1EH * 4 ; address of ptr to disk parameters
                                SEC9EQU     522h    ; address of disk parameters
                                CHROUT      EQU     29h
                                LSTDRV      EQU      504h

                                ; determine whether we booted from floppy or hard disk...

                                        ; 20/12/2022
00001C70 88C8                           mov     al, cl  ; 26/05/2019

00001C72 F6C280                         test    dl, 80h         ; boot from floppy ?
00001C75 7502                           jnz     short gothrd    ; no.
```

```
00001C77 31C0                           xor     ax, ax          ; indicate boot       from drive a
                                        ; 10/12/2023
                                        ;mov    [Boot_Drv], al
                                gothrd:

                                ; MSDOS 6.0
                                ;   ax = 0-based drive we booted from
                                ;   bios_l, bios_h set.
                                ;   cl = number of floppies including fake one
                                ;   ch = media byte

                                ; Retro DOS 4.0 - 27/12/2018
                                ;  (from Retro DOS v2.0 boot sector)
                                ;   dl = int 13 drive number we booted from
                                ;   dh = media byte

                                        ; 20/12/2022
00001C79 88F5                           mov     ch, dh          ; 01/07/2018

                                        ; cl = number of floppies
                                        ; ch = media byte

                                        ; set up local stack

                                        ; 20/12/2022
                                        ;xor    dx, dx          ; ax = 0-based drive we       booted from
                                                                ; bios_l, bios_h set.
                                                                ; cl = number of floppies including fake one
                                                                ; ch = media byte
                                        ; 20/12/2022
                                        ; es = ds = cs
                                        ; ss = 0
                                        ; sp = 700h

                                        ; 20/12/2022
                                        ;cli
                                        ;mov    ss, dx          ; set stack segment and stack pointer
                                        ;mov    sp, 700h
                                        ;sti

00001C7B 51                             push    cx ; (***)      ; save number of floppies and media byte

00001C7C 88EC                           mov     ah, ch          ; FAT ID to AH
00001C7E 50                             push    ax ; (**)       ; save boot drive number and media byte

                                ; let model_byte, secondary_model_byte be set here!!!

00001C7F B4C0                           mov     ah, 0C0h
00001C81 CD15                           int     15h     ; SYSTEM - GET CONFIGURATION (XT after 1/10/86,AT mdl 3x9,CONV,XT286,PS)
00001C83 7215                           jb      short no_rom_system_conf ; just       use Model_Byte
00001C85 80FC00                         cmp     ah, 0
00001C88 7510                           jnz     short no_rom_system_conf

                                ;       ; 20/12/2022
                                ;       ; (Programmer's Guide to the AMIBIOS, page 268)
                                ;       ; (https://stanislavs.org/helppc/int_15-c0.html)
                                ;       ;
                                ;       INT 15h, ah = C0h - Return System Configuration Parameters (PS/2 only)
                                ;       ;
                                ;       on return:
                                ;       CF = 0 if successful
                                ;          = 1 if error
                                ;       AH = when CF set, 80h for PC & PCjr, 86h for XT
                                ;           (BIOS after 11/8/82) and AT (BIOS after 1/10/84)
                                ;       ;
                                ;       ES:BX = pointer to system descriptor table in ROM of the format:
                                ;       ;
                                ;       Offset Size         Description
                                ;       ;
                                ;         00   word   length of descriptor (8 minimum)
                                ;         02   byte   model byte (same as F000:FFFE, not reliable)
                                ;         03   byte   secondary model byte
                                ;         04   byte   BIOS revision level (zero based)
                                ;         05   byte   feature information, see below
                                ;         06   dword  reserved

                                        ; 20/12/2022
                                        ; ds = cs
00001C8A 268A4702                       mov     al, [es:bx+2]  ; [es:bx+ROMBIOS_DESC.bios_sd_modelbyte]
00001C8E A2[AF05]                       mov     [model_byte], al
                                        ;mov    [cs:model_byte], al
                                                                ; get/save model byte
00001C91 268A4703                       mov     al, [es:bx+3]  ; [es:bx+ROMBIOS_DESC.bios_sd_scnd_modelbyte]
00001C95 A2[B005]                       mov     [secondary_model_byte], al
                                        ;mov    [cs:secondary_model_byte], al
                                                                ; get/save secondary model byte
00001C98 EB0C                           jmp     short turn_timer_on
                                ;-----------------------------------------------------------------------------

                                no_rom_system_conf:
00001C9A BEFFFF                                 mov     si, 0FFFFh
00001C9D 8EC6                                   mov     es, si
                                                ; 20/12/2022
00001C9F 26A00E00                               mov     al, [es:0Eh]   ; get model byte (from 0FFFFh:0Eh)
00001CA3 A2[AF05]                               mov     [model_byte], al
                                                ;mov    [cs:model_byte], al ; save model byte
                                turn_timer_on:
00001CA6 B020                                   mov     al, 20h ; ' '   ; turn on the timer
00001CA8 E620                                   out     20h, al         ; Interrupt controller,       8259A.
                                                                        ; AKPORT

                                ; some olivetti m24 machines have an 8530 serial communications
                                ; chip installed at io address 50h and 52h. if we're running
                                ; on one of those, we must inhibit the normal aux port initialization

                                                ; 20/12/2022
                                                ; ds = cs
00001CAA 803E[AF05]00                           cmp     byte [model_byte], 0
                                                ;cmp    byte [cs:model_byte], 0 ; next to last      byte in rom bios
00001CAF 7510                                   jnz     short not_olivetti_m24 ; skip for all other machines
                                                                        ; (except olivetti m24)
00001CB1 E466                                   in      al, 66h         ; is 8530 installed?
00001CB3 A820                                   test    al, 20h
00001CB5 740A                                   jz      short not_olivetti_m24 ; we're done if not
00001CB7 B00F                                   mov     al, 0Fh         ; double check
00001CB9 E650                                   out     50h, al
00001CBB E450                                   in      al, 50h
00001CBD A801                                   test    al, 1           ; this test was       copied from olivetti
00001CBF 7414                                   jz      short skip_aux_port_init ; take      this branch if 8530 installed

                                not_olivetti_m24:
00001CC1 B003                                   mov     al, 3           ; init com4
00001CC3 E8DB09                                 call    aux_init
00001CC6 B002                                   mov     al, 2           ; init com3
```

```
5590 00001CC8 E8D609                            call    aux_init
5591 00001CCB B001                              mov     al, 1           ; init com2
5592 00001CCD E8D109                            call    aux_init
5593 00001CD0 30C0                              xor     al, al          ; init com1
5594 00001CD2 E8CC09                            call    aux_init
5595
5596                                    skip_aux_port_init:
5597 00001CD5 B002                              mov     al, 2           ; init lpt3
5598 00001CD7 E8BF09                            call    print_init
5599 00001CDA B001                              mov     al, 1           ; init lpt2
5600 00001CDC E8BA09                            call    print_init
5601 00001CDF 30C0                              xor     al, al          ; init lpt1
5602 00001CE1 E8B509                            call    print_init
5603
5604                                            ; 11/12/2023 - Retro DOS v5.0 (PCDOS 7.1 IBMBIO.COM)
5605                                            ;mov    di, 534h        ; offset INITSPOT
5606                                            ;;mov   di, INITSPOT    ; 0534h
5607                                            ;                       ; IBMDOS.COM's first cluster - high word
5608                                            ;                       ; 520h (the 2nd entry of root dir) + 14h
5609                                            ;mov    ax, [di]
5610                                            ;mov    [firstcluster_hw], ax
5611
5612 00001CE4 31D2                              xor     dx, dx  ; 0
5613 00001CE6 8EDA                              mov     ds, dx          ; to initialize       print screen vector
5614 00001CE8 8EC2                              mov     es, dx
5615 00001CEA 31C0                              xor     ax, ax
5616                                            ; 16/10/2022
5617 00001CEC BF3405                            mov     di, INITSPOT    ; 0534h
5618                                            ;mov    di, 534h        ; INITSPOT (0000h:0534h)
5619                                                                    ; IBM wants 4 zeros here
5620 00001CEF AB                                stosw
5621 00001CF0 AB                                stosw
5622 00001CF1 8CC8                              mov     ax, cs          ; fetch segment
5623 00001CF3 C7066C00[0E06]                    mov     word [BRKADR], cbreak
5624                                            ;mov    word ptr ds:6Ch, offset       cbreak ; [BRKADR]
5625                                                                    ; break entry point
5626 00001CF9 A36E00                            mov     [BRKADR+2], ax
5627                                            ;mov    ds:6Eh, ax      ; vector for break
5628 00001CFC C706A400[8206]                    mov     word [CHROUT*4], outchr
5629                                            ;mov    word ptr ds:0A4h, offset outchr       ; [CHROUT*4]
5630 00001D02 A3A600                            mov     [CHROUT*4+2], ax
5631                                            ;mov    ds:0A6h, ax     ; [CHROUT*4+2]
5632
5633 00001D05 BF0400                            mov     di, 4
5634 00001D08 BB[1406]                          mov     bx, intret ; 19/10/2022
5635                                            ;mov    bx, offset intret ; intret (cs:intret)
5636                                                                    ; will initialize rest of interrupts
5637 00001D0B 93                                xchg    ax, bx
5638 00001D0C AB                                stosw                   ; location 4
5639 00001D0D 93                                xchg    ax, bx          ; cs:
5640 00001D0E AB                                stosw                   ; int 1 ; location 6
5641 00001D0F 83C704                            add     di, 4
5642 00001D12 93                                xchg    ax, bx
5643 00001D13 AB                                stosw                   ; location 12
5644 00001D14 93                                xchg    ax, bx          ; cs:
5645 00001D15 AB                                stosw                   ; int 3 ; location 14
5646 00001D16 93                                xchg    ax, bx
5647 00001D17 AB                                stosw                   ; location 16
5648 00001D18 93                                xchg    ax, bx          ; cs:
5649 00001D19 AB                                stosw                   ; int 4 ; location 18
5650
5651                                    ;           ; 20/12/2022
5652                                    ;           ; (https://stanislavs.org/helppc/bios_data_area.html)
5653                                    ;           Address Size   Description    (BIOS/DOS Data Area)
5654                                    ;
5655                                    ;           50:00   byte    Print screen status byte
5656                                    ;                           00 = PrtSc not active,
5657                                    ;                           01 = PrtSc in progress
5658                                    ;                           FF = error
5659                                    ;           50:01  3 bytes Used by BASIC
5660                                    ;           50:04   byte    DOS single diskette mode flag, 0=A:, 1=B:
5661                                    ;           50:05  10bytes POST work area
5662                                    ;           50:0F   byte    BASIC shell flag; set to 2 if current shell
5663                                    ;           50:10   word    BASICs default DS value (DEF SEG)
5664                                    ;           50:12   dword   Pointer to BASIC INT 1C interrupt handler
5665                                    ;           50:16   dword   Pointer to BASIC INT 23 interrupt handler
5666                                    ;           50:1A   dword   Pointer to BASIC INT 24 disk error handler
5667                                    ;           50:20   word    DOS dynamic storage
5668                                    ;           50:22  14bytes DOS diskette initialization table (INT 1E)
5669                                    ;           50:30   4bytes  MODE command
5670                                    ;           70:00           I/O drivers from IO.SYS/IBMBIO.COM
5671
5672 00001D1A 89160005                          mov     [0500h], dx ; 0
5673                                            ;mov    ds:500h, dx     ; set print screen & break = 0
5674 00001D1E 89160405                          mov     [LSTDRV], dx    ; [0504h]
5675                                            ;mov    ds:504h, dx     ; clean out last drive spec
5676
5677                                    ; we need to initialize the cs:motorstartup variable from the disk
5678                                    ; parameter table at sec9. the offsets in this table are defined in
5679                                    ; the disk_parms struc in msdskprm.inc. 2 locs
5680
5681 00001D22 A02C05                                  mov     al, [SEC9+0Ah] ; 16/10/2022
5682                                                  ;mov    al, ds:52Ch     ; [SEC9+DISK_PARMS.DISK_MOTOR_STRT]
5683                                                                          ; [522h+0Ah]
5684                                                  ; 20/12/2022
5685                                                  ; ds = 0
5686
5687 00001D25 2EA2[2601]                            mov     [cs:motorstartup], al
5688 00001D29 2E803E[AF05]FD                       cmp     byte [cs:model_byte], 0FDh ; is this an old rom?
5689 00001D2F 720B                                 jb      short no_diddle         ; no
5690 00001D31 C7062B050F02                         mov     word [SEC9+09h], 20Fh
5691                                                ;mov    word ptr ds:52Bh, 20Fh ; [SEC9+DISK_PARMS.DISK_HEAD_STTL], 0200h+NORMSETTLE
5692                                                                          ; set head settle and motor start on pc-1 pc-2 pc-xt hal0
5693 00001D37 C6062205DF                           mov     byte [SEC9+0], 0DFh
5694                                                ;mov    byte ptr ds:522h, 0DFh ; [SEC9+DISK_PARMS.DISK_SPECIFY_1]
5695                                                                          ;  set 1st specify byte       on pc-1 pc-2 pc-xt hal0
5696                                    no_diddle:
5697 00001D3C CD12                                 int     12h             ; MEMORY SIZE -
5698                                                                        ; Return: AX = number of contiguous 1K blocks of memory
5699 00001D3E B106                                 mov     cl, 6
5700 00001D40 D3E0                                 shl     ax, cl          ; convert memory size to 16-byte blocks     (segment no.)
5701
5702                                                ; 20/12/2022
5703                                                ; 03/07/2018 - 27/12/2018
5704                                                ;pop    cx ; (**)
5705                                                ;mov    [cs:drvfat], cx
5706
5707 00001D42 50                                   push    ax ; (*)        ; save real top       of memory
5708
5709                                                ; 27/12/2018 - (MSDOS 6.0, 6.21)
5710
5711                                    ;M068 - BEGIN
5712                                    ;------ Check if an RPL program is present at TOM and do not tromp over it
5713
```

```
                                        ; 20/12/2022
                                        ; ds = 0

                                        ;push  ds
                                        ;push  bx             ; pushes not required but since this
                                                              ; happens to be a last minute change
                                                              ; & since it is only init code.
                                        ;xor   bx, bx
                                        ;mov   ds, bx

                                        ;;mov  bx, ds:0BCh    ; [2Fh*4]
                                        ;mov   bx, [2Fh*4]
                                        ;;mov  ds, word ptr ds:0BEh ; [2Fh*4+2]
                                        ;mov   ds, [2Fh*4+2]
                                        ; 29/09/2023
00001D43 C51EBC00                       lds    bx, [2Fh*4]

00001D47 817F035250                     cmp    word [bx+3], 'RP' ; 'RPL'
                                        ;cmp   word ptr [bx+3], 'PR' ; 'RPL'
00001D4C 750F                           jnz    short SkipRPL
00001D4E 807F054C                       cmp    byte [bx+5], 'L'
                                        ;cmp   byte ptr [bx+5], 'L'
00001D52 7509                           jnz    short SkipRPL
00001D54 89C2                           mov    dx, ax         ; get TOM into DX
00001D56 B8064A                         mov    ax, 4A06h      ; (multMULT shl       8) + multMULTRPLTOM
00001D59 CD2F                           int    2Fh            ; Get new TOM from any RPL
00001D5B 89D0                           mov    ax, dx
                                SkipRPL:
                                        ; 20/12/2022
                                        ;pop   bx
                                        ;pop   ds

                                ;M068 - END
                                        ; 20/12/2022
                                        ; 27/12/2018
00001D5D 0E                             push   cs
00001D5E 1F                             pop    ds
                                        ; 18/03/2019 - Retro DOS v4.0
                                        ;sub   ax, 64         ; room for fatloc segment. (1 kb buffer)
                                        ;mov   [cs:fatloc], ax       ; location to read fat

                                        ; 01/07/2018
                                        ; 08/04/2018
                                        ; 28/03/2018
                                        ; MSDOS 6.0 - MSINIT.ASM, 1991
00001D5F 83E840                         sub    ax, 64
00001D62 A3[041A]                       mov    [init_bootseg], ax ; 20/12/2022
                                        ;mov   [cs:init_bootseg], ax

                                        ; 27/12/2018 - Retro DOS v4.0
                                        ;;pop  ax ; (*)       ; get back real top of memory
                                        ;pop   dx ; (*)
                                        ; 29/09/2023 - Retro DOS v4.2 (BugFix)
00001D65 58                             pop    ax ; (*)       ; get back real top of memory


                                        ; 20/12/2022
                                        ; 27/12/2018
00001D66 59                             pop    cx ; (**)
00001D67 890E[FA19]                     mov    [drvfat], cx   ; save drive to load dos, and fat id

                                        ; 20/12/2022

                                        ;mov   dx, 46Dh       ; SYSINIT segment
                                        ;mov   dx, 544h       ; 10/12/2023 (PCDOS 7.1 IBMBIO.COM)
00001D6B BAD904                         mov    dx, SYSINITSEG ; 17/10/2022
00001D6E 8EDA                           mov    ds, dx

                                ; set pointer to resident device driver chain

                                        ; 17/10/2022
00001D70 C706[7502][2300]               mov    word [DEVICELIST], res_dev_list
                                        ;mov   word [273h], res_dev_list
                                        ;;mov  word ptr ds:273h, offset res_dev_list
                                                              ; [SYSINIT+DEVICE_LIST]
00001D76 8C0E[7702]                     mov    [DEVICELIST+2], cs
                                        ;mov   [275h], cs
                                        ;;mov  word ptr ds:275h, cs ; [SYSINIT+DEVICE_LIST+2]

00001D7A A3[9402]                       mov    [MEMORYSIZE], ax
                                        ;mov   [292h], ax
                                        ;;mov  ds:292h, ax    ; [SYSINIT+MEMORY_SIZE]

00001D7D FEC1                           inc    cl
00001D7F 880E[9802]                     mov    [DEFAULTDRIVE], cl
                                        ;mov   [296h], cl
                                        ;;mov  ds:296h, cl    ; [SYSINIT+DEFAULT_DRIVE]

                                        ;mov   word [CURRENTDOSLOCATION], 0AF8h ; 10/12/2023
00001D83 C706[7302]1A0A                 mov    word [CURRENTDOSLOCATION], DOSLOADSEG
                                        ;mov   word [271h], 83Fh ; (MSDOS.SYS segment)
                                        ;;mov  word ptr ds:271h, 83Fh ; [SYSINIT+CURRENT_DOS_LOCATION]
                                                              ; dos_load_seg

                                ; important: some old ibm hardware generates spurious int 0F's due to bogus
                                ; printer cards. we initialize this value to point to an iret only if
                                ;
                                ; 1) the original segment points to storage inside valid ram.
                                ;
                                ; 2) the original segment is 0F000:xxxx

                                        ;;;mov  ax, 46Dh       ; SYSINIT segment
                                        ;;mov  ax, 544h       ; 10/12/2023
                                        ;mov   ax, SYSINITSEG ; 17/10/2022
                                        ;mov   es, ax
                                        ; 20/12/2022
                                        ;push  ds ; SYSINITSEG
                                        ;pop   es
00001D89 8EC2                           mov    es, dx ; SYSINITSEG
00001D8B 31C0                           xor    ax, ax ; 0
00001D8D 8ED8                           mov    ds, ax         ; segment 0
                                        ;mov   ax, ds:3Eh     ; [0Fh*4+2]
00001D8F A13E00                         mov    ax, [0Fh*4+2]  ; segment for INT 0Fh
                                        ; 18/10/2022
00001D92 263B06[9402]                   cmp    ax, [es:MEMORYSIZE] ; es:292h
                                        ;cmp   ax, es:292h    ; [ES:MEMORY_SIZE]  ; (condition 1)
00001D97 7605                           jbe    short resetintf
00001D99 3D00F0                         cmp    ax, 0F000h     ; (condition 2)
00001D9C 750A                           jnz    short keepintf
                                resetintf:
00001D9E C7063C00[1406]                 mov    word [0Fh*4], intret
                                        ;mov   word ptr ds:3Ch, offset      intret ; [0Fh*4]
00001DA4 8C0E3E00                       mov    word [0Fh*4+2], cs
```

```
                                        ;mov   word ptr ds:3Eh, cs ; [0Fh*4+2]
                                keepintf:
                                ; end important

                                ; 17/10/2022
                                ; 28/12/2018 - Retro DOS v4.0

                                ; (MSDOS 6.0, MSINIT.ASM, 1991)
                                ;
                                ; we will check if the system has ibm extended keyboard by
                                ; looking at a byte at 40:96. if bit 4 is set, then extended keyboard
                                ; is installed, and we are going to set keyrd_func to 10h, keysts_func to 11h
                                ; for the extended keyboard function. use cx as the temporary register.

                                        ; 20/12/2022
                                        ; ds = 0
                                        ;xor   cx, cx
                                        ;mov   ds, cx

00001DA8 8A0E9604                       mov    cl, [496h]     ; get keyboard flag

                                        ; 20/12/2022
                                        ; 20/03/2019
00001DAC 0E                             push   cs
00001DAD 1F                             pop    ds

                                        ;test  cl, 00010000b ; 10h
00001DAE F6C110                         test   cl, 10h        ; extended keyboard ?
00001DB1 740A                           jz     short org_key  ; no, original keyboard

                                        ; 20/12/2022
                                        ;  ds = cs
00001DB3 C606[7E04]10                   mov    byte [keyrd_func], 10h ; extended keyboard
00001DB8 C606[7F04]11                   mov    byte [keysts_func], 11h
                                        ;mov   byte [cs:keyrd_func], 10h ; extended keyboard
                                        ;mov   byte [cs:keysts_func], 11h
                                                       ; change for extended keyboard functions
                                org_key:

                                ; 02/06/2018 - Retro DOS v3.0

                                ;*************************************************************
                                ;   will initialize the number of drives
                                ;   after the equipment call (int 11h) bits 6&7 will tell
                                ;   the indications are as follows:
                                ;
                                ;   bits    7       6       drives
                                ;           0       0       1
                                ;           0       1       2
                                ;           1       0       3
                                ;           1       1       4
                                ;*************************************************************

                                        ; 20/12/2022
                                        ; ds = cs
                                        ;push  cs
                                        ;pop   ds
                                        ; 21/12/2022
                                        ;push  cs
                                        ;pop   es

00001DBD E8C20A                         call   cmos_clock_read       ; If cmos clock       exists,
                                                               ; then set the system time according to     that.
                                                               ; also, reset the cmos clock rate.
                                        ; 18/10/2022
                                        ;mov   word ptr BData_start, offset harddrv ;
                                                               ; set up pointer to hdrive
                                        ; 02/10/2022
00001DC0 C706[0000][4B08]               mov    word [hdrv_pat], harddrv

                                        ; 20/12/2022
                                        ; 28/12/2018 - Retro DOS v4.0 (MSDOS 6.21)
00001DC6 58                             pop    ax ; (***)     ; number of floppies and FAT ID

00001DC7 30E4                           xor    ah, ah         ; chuck fat id byte
00001DC9 A2[7500]                       mov    [drvmax], al   ; remember which drive is hard disk
00001DCC A2[2501]                       mov    [dsktnum], al  ; and set initial number of drives
00001DCF D1E0                           shl    ax, 1
00001DD1 0106[5E1A]                     add    [last_dskdrv_table], ax

                                        ; 10/12/2023 - Retro DOS v5.0 IO.SYS (IBMBIO.COM)
                                        ; ((MSDOS 6.22 IO.SYS & PCDOS 7.1 IBMBIO.COM))
                                        ; ........
00001DD5 1E                             push    ds
00001DD6 B800F0                         mov     ax, 0F000h
00001DD9 8ED8                           mov     ds, ax

00001DDB 813EEAFF434F                   cmp    word [0FFEAh], 'CO' ; 'COMPAQ'
00001DE1 751F                           jne    short skip_mode2
00001DE3 813EECFF4D50                   cmp    word [0FFECh], 'MP'
00001DE9 7517                           jne    short skip_mode2
00001DEB 813EEEFF4151                   cmp    word [0FFEEh], 'AQ'
00001DF1 750F                           jne    short skip_mode2

00001DF3 B800E4                         mov    ax, 0E400h     ; get advanced system info (COMPAQ ROMBIOS)
00001DF6 CD15                           int    15h
00001DF8 7208                           jc     short skip_mode2
                                        ; 10/12/2023
                                        ; PCDOS 7.1 IBMBIO.COM
                                        ;or    bx, 0           ; or bx,40h ; enable mode 2
                                                               ; (MSDOS 6.0)
                                        ; MSDOS 6.22 IO.SYS
00001DFA 83CB40                         or     bx, 40h        ; enable mode 2 (dual harddisk controller)
00001DFD B880E4                         mov    ax, 0E480h      ; set advanced system info (COMPAQ ROMBIOS)
00001E00 CD15                           int    15h
                                skip_mode2:
00001E02 1F                             pop    ds
                                        ; ........

00001E03 B280                           mov    dl, 80h
00001E05 B408                           mov    ah, 8
00001E07 CD13                           int    13h            ; DISK - DISK -       GET CURRENT DRIVE PARAMETERS (XT,AT,XT286,CONV,PS)
                                                               ; DL = drive number
                                                               ; Return: CF set on error, AH =      status code, BL       = drive type
                                                               ; DL = number of consecutive drives
                                                               ; DH = maximum value for head number, ES:DI -> drive parameter
00001E09 7204                           jc     short enddrv
00001E0B 8816[5D1A]                     mov    [hnum], dl     ; save number of hard disk drives
                                enddrv:
                                        ; 21/12/2022
00001E0F 0E                             push   cs
00001E10 07                             pop    es

                                ; scan the list of drives to determine their type. we have three flavors of
```

```
                                ; diskette drives:
                                ;
                                ;   48tpi drives    we do nothing special for them
                                ;   96tpi drives    mark the fact that they have changeline support.
                                ;   3.5"  drives    mark changeline support and small.
                                ;
                                ; the following code uses registers for certain values:
                                ;
                                ;   dl - physical drive
                                ;   ds:di - points to current bds
                                ;   cx - flag bits for bds
                                ;   dh - form factor for the drive (1 - 48tpi, 2 - 96tpi, 3 - 3.5" medium)
00001E11 30D2                           xor     dl, dl

                                        ; 20/12/2022
                                        ; ds = cs
                                        ; 17/06/2018
                                        ;push   cs
                                        ;pop    ds

00001E13 C606[2C01]09                   mov     byte [eot], 9
00001E18 BF[1901]                       mov     di, start_bds  ; if we are faking floppy drives we need
                                                               ; to set aside two bdss      for the two fake floppy       drives

                                ; 02/10/2022 - Retro DOS v4.0 (MSDOS 5.0 IO.SYS)
                                ; 28/12/2018 - Retro DOS v4.0 (MSDOS 6.0, MSINIT.ASM)

                                ; check to see if we are faking floppy drives. if not we don't
                                ; do anything special. if we are faking floppy drives we need
                                ; to set aside two bdss for the two fake floppy drives. we
                                ; don't need to initalise any fields though. so starting at start_bds
                                ; use the link field in the bds structure to go to the second bds
                                ; in the list and initalise it's link field to -1 to set the end of
                                ; the list. then jump to the routine at dohard to allocate/initialise
                                ; the bds for harddrives.

00001E1B 803E[111A]01                   cmp     byte [fakefloppydrv], 1
00001E20 750B                           jnz     short loop_drive
00001E22 8B3D                           mov     di, [di]       ; [di+BDS.link]
                                                               ; di <- first bds link
00001E24 8B3D                           mov     di, [di]       ; [di+BDS.link]
                                                               ; di <- second bds link
00001E26 C705FFFF                       mov     word [di], 0FFFFh ; -1 ; set end of link
00001E2A E98801                         jmp     dohard         ; allocate/initialise bds for harddrives
                                ;--------------------------------------------------------------------------

                                loop_drive:
00001E2D 3A16[7500]                     cmp     dl, [drvmax]
00001E31 7203                           jb      short got_more
00001E33 E97B01                         jmp     done_drives
                                ;--------------------------------------------------------------------------

                                got_more:
                                        ; 12/12/2023
                                        ;xor    cx, cx         ; zero all flags
00001E36 8B3D                           mov     di, [di]       ; [di+BDS.link]
                                                               ; get next bds
                                        ; .......
                                        ; 10/12/2023 - Retro DOS v5.0
                                        ; (PCDOS 7.1 IBMBIO.COM BIOSDATA:2046h)
00001E38 83FFFF                         cmp     di, 0FFFFh      ; end of link ?
00001E3B 7516                           jne     short not_last_bds
00001E3D 88D0                           mov     al, dl          ; drive number (0 based)
00001E3F 98                             cbw
00001E40 01C0                           add     ax, ax
00001E42 05[3C05]                       add     ax, dskdrvs
00001E45 A3[5E1A]                       mov     [last_dskdrv_table], ax
00001E48 8B3E[601A]                     mov     di, [end_of_bdss]
00001E4C E8FD09                         call    xinstall_bds
00001E4F FE0E[7500]                     dec     byte [drvmax]
                                not_last_bds:
                                        ; ........

00001E53 B600                           mov     dh, 0          ; ff48tpi
                                                               ; set form factor to 48      tpi
00001E55 C606[0E1A]28                   mov     byte [num_cyln], 40 ; 40 tracks per side

                                        ; 20/12/2022
                                        ;push   ds ; 11/05/2019
00001E5A 57                             push    di
00001E5B 52                             push    dx
                                        ;push   cx ; not necessary (10/12/2023)
00001E5C 06                             push    es ; ((*)) ; 20/12/2022

                                        ; ..........
                                        ; 10/12/2023 - Retro DOS v5.0 (PCDOS 7.1 IBMBIO.COM)
                                        ;xor    bx, bx
                                        ;xor    cx, cx
00001E5D 52                             push    dx  ; dl = drive number

00001E5E B408                           mov     ah, 8
00001E60 CD13                           int     13h            ; DISK - DISK -       GET CURRENT DRIVE PARAMETERS (XT,AT,XT286,CONV,PS)
                                                               ; DL = drive number
                                                               ; Return: CF set on error, AH =      status code, BL       = drive type
                                                               ; DL = number of consecutive drives
                                                               ; DH = maximum value for head number, ES:DI -> drive parameter
                                        ;jc     short noparmsfromrom
                                        ; 10/12/2023
00001E62 58                             pop     ax  ; al = drive number
00001E63 7303                           jnc     short chk_drv_type
00001E65 E9E600                         jmp     noparmsfromrom

                                chk_drv_type:
                                        ; 10/12/2023
                                        ; ch = low eight bits of maximum cylinder number
                                        ; cl = maximum sector number (bits 5-0)
                                        ;      high two bits of maximum cylinder number (bits 7-6)
                                        ;
00001E68 80FB10                         cmp     bl, 10h        ; ATAPI Removable Media Device
00001E6B 7554                           jne     short not_atapi_removable

                                        ; save ds:si
00001E6D 1E                             push    ds
                                        ;push   si      ; not necessary (10/12/2023)

00001E6E 88C2                           mov     dl, al
00001E70 83EC1A                         sub     sp, 26
00001E73 31C0                           xor     ax, ax ; 0
00001E75 50                             push    ax
00001E76 B81E00                         mov     ax, 30
00001E79 50                             push    ax
00001E7A 89E6                           mov     si, sp         ; DS:SI = segment:offset pointer to Result Buffer
00001E7C 16                             push    ss
```

```
6086 00001E7D 1F                                pop     ds
6087 00001E7E B448                              mov     ah, 48h
6088 00001E80 CD13                              int     13h             ; DISK - IBM/MS Extension
6089                                                                    ; GET DRIVE PARAMETERS (DL - drive, DS:SI - buffer)
6090 00001E82 7239                              jb      short ext_gdp_err
6091 00001E84 8B4408                            mov     ax, [si+8]      ; physical number of heads
6092 00001E87 A3[0C1A]                          mov     [num_heads], ax
6093 00001E8A 8B4404                            mov     ax, [si+4]      ; physical number of cylinders
6094 00001E8D A3[0E1A]                          mov     [num_cyln], ax
6095 00001E90 8A440C                            mov     al, [si+0Ch]    ; physical number of sectors per track
6096 00001E93 A2[101A]                          mov     [sec_trk], al
6097 00001E96 3A06[2C01]                        cmp     al, [eot]
6098 00001E9A 7603                              jbe     short _eotok
6099 00001E9C A2[2C01]                          mov     [eot], al
6100
6101                                    _eotok:             ; 10/12/2023
6102                                                        ;xor    al, al
6103 00001E9F 31C9                              xor     cx, cx ; 0
6104 00001EA1 F6440210                          test    byte [si+2], 10h ; information flags
6105                                                                    ; bit 4 = Device has change line support
6106 00001EA5 7403                              jz      short not_chgline_sup
6107                                                        ;or     al, 2           ; change line support
6108 00001EA7 80C902                            or      cl, 2
6109                                    not_chgline_sup:
6110 00001EAA 83C41E                            add     sp, 30
6111                                                        ;pop    si      ; (10/12/2023)
6112 00001EAD 1F                                pop     ds
6113                                                        ;
6114 00001EAE 07                                pop     es      ; es=cs=ds (21/12/2022)
6115                                                        ;pop    cx      ; (10/12/2023)
6116 00001EAF 5A                                pop     dx
6117 00001EB0 5F                                pop     di
6118                                                        ;pop    ds      ; (21/12/2022)
6119
6120                                                        ; 10/12/2023
6121 00001EB1 F6C102                            test    cl, 2
6122                                                        ;test   al, 2
6123                                                        ;jz     short gotother_j
6124 00001EB4 7450                              jz      short gotother
6125                                                        ;or     cl, al
6126 00001EB6 C606[7700]01                      mov     byte [fhave96], 1 ; Device has change line support
6127                                    gotother_j:
6128 00001EBB EB49                              jmp     short gotother
6129                                    ext_gdp_err:
6130 00001EBD 83C41E                            add     sp, 30
6131                                                        ;pop    si      ; (10/12/2023)
6132 00001EC0 1F                                pop     ds
6133
6134                                                        ; 10/12/2023
6135                                    not_atapi_removable:
6136                                                        ; ...........
6137
6138
6139                                    ; if cmos is bad, it gives es,ax,bx,cx,dh,di=0. cy=0.
6140                                    ; in this case, we are going to put bogus informations to bds table.
6141                                    ; we are going to set ch=39,cl=9,dh=1 to avoid divide overflow when
6142                                    ; they are calculated at the later time. this is just for the diagnostic
6143                                    ; diskette which need msbio,msdos to boot up before it sets cmos.
6144                                    ; this should only happen with drive b.
6145
6146 00001EC1 80FD00                                cmp     ch, 0           ; if ch=0, then       cl,dh=0 too.
6147 00001EC4 7505                              jnz     short pfr_ok
6148
6149                                                        ;mov    ch, 39          ; rom gave wrong info.
6150                                                        ;mov    cl, 9           ; let's default to 360k.
6151                                                        ; 20/12/2022
6152 00001EC6 B90927                            mov     cx, 2709h
6153
6154 00001EC9 B601                              mov     dh, 1
6155                                    pfr_ok:
6156                                                        ;inc    dh              ; make number of heads 1-based
6157                                                        ;mov    [num_heads], dh         ; save parms returned by rom
6158                                                        ; 10/12/2023
6159 00001ECB 86D6                              xchg    dl, dh
6160 00001ECD 30F6                              xor     dh, dh
6161 00001ECF 42                                inc     dx              ; make number of heads 1-based
6162 00001ED0 8916[0C1A]                        mov     [num_heads], dx
6163
6164                                                        ;inc    ch              ; make number of cylinders 1-based
6165                                                        ;and    cl, 3Fh
6166                                                        ;mov    [sec_trk], cl
6167                                                        ;mov    [num_cyln], ch ; assume less than 256 cylinders!!
6168                                                        ; 10/12/2023
6169 00001ED4 88CA                              mov     dl, cl
6170 00001ED6 80E23F                            and     dl, 3Fh
6171 00001ED9 8816[101A]                        mov     [sec_trk], dl
6172 00001EDD 86CD                              xchg    cl, ch
6173 00001EDF D0C5                              rol     ch, 1
6174 00001EE1 D0C5                              rol     ch, 1
6175 00001EE3 80E503                            and     ch, 3
6176 00001EE6 41                                inc     cx              ; make number of cylinders 1-based
6177 00001EE7 890E[0E1A]                        mov     [num_cyln], cx
6178
6179                                    ; make sure that eot contains the max number of sec/trk in system of floppies
6180
6181                                                        ;mov    cl, [sec_trk] ; 10/12/2023
6182                                                        ;cmp    cl, [eot]       ; may set carry
6183                                                        ;;jbe   short eot_ok
6184                                                        ;; 09/12/2022
6185                                                        ;;jne   short eotok  ; wrong ! 14/08/2023
6186                                                        ;; 14/08/2023
6187                                                        ;jbe    short eotok
6188                                                        ;mov    [eot], cl
6189                                                        ; 10/12/2023
6190 00001EEB 3A16[2C01]                        cmp     dl, [eot] ; dl = [sec_trk]
6191 00001EEF 7604                              jbe     short eotok
6192 00001EF1 8816[2C01]                        mov     [eot], dl
6193                                    ;eot_ok:
6194                                    eotok:
6195                                                        ; 10/12/2023
6196                                                        ; !!!
6197                                                        ; (following pops are moved to 'chk_changeline' procedure)
6198                                                        ;
6199                                                        ; 20/12/2022
6200                                                        ;pop    es ; ((*)) es = cs = ds
6201                                                        ;;pop   cx      ; 10/12/2023
6202                                                        ;pop    dx
6203                                                        ;pop    di
6204
6205                                                        ; 20/12/2022
6206                                                        ;pop    ds
6207
6208                                    ; Check     for presence of       changeline
6209
```

```
                                ; 10/12/2023
                                %if 0
                                        ; 10/12/2023
                                        ;xor    cx, cx  ; 0
                                        ;push   cx
                                        push    dx

                                        mov     ah, 15h
                                        int     13h             ; DISK - DISK -       GET TYPE (AT,XT2,XT286,CONV,PS)
                                                                ; DL = drive ID
                                                                ; Return: CF set on error, AH =      disk type (3 = hard drive)
                                                                ; CX:DX = number of sectors on the media
                                        ; 10/12/2023
                                        pop     dx
                                        ;pop    cx
                                        mov     cx, 0 ; 12/12/2023
                                        jc      short changeline_done
                                        cmp     ah, 2           ; check for presence of       changeline
                                        jnz     short changeline_done

                                ; we have a drive with change line support.

                                        or      cl, 2           ; fchangeline
                                                                ; signal type
                                        mov     byte [fhave96], 1 ; remember that we have 96tpi disks
                                %endif
                                        ; 10/12/2023
00001EF5 E83800                         call    chk_changeline
                                        ;jc     short changeline_done

                                ; we now try to set up the form factor for the types of media that we know
                                ; and can recognise. for the rest, we set the form factor as "other".

                                changeline_done:

                                ; 40 cylinders and 9 or less sec/trk, treat as 48 tpi medium

00001EF8 803E[0E1A]28                   cmp     byte [num_cyln], 40
00001EFD 750B                           jnz     short try_80
00001EFF 803E[101A]09                   cmp     byte [sec_trk], 9
00001F04 765F                           jbe     short nextdrive
                                gotother:
                                        ; 10/12/2023
                                        ; ch = 0, cl = 2 or 0

00001F06 B607                           mov     dh, 7           ; ffOther
                                                                ; we have a "strange" medium
00001F08 EB5B                           jmp     short nextdrive
                                ;-------------------------------------------------------------------------

                                ; 80 cylinders and 9 sectors/track => 720 kb device
                                ; 80 cylinders and 15 sec/trk => 96 tpi medium

                                try_80:
00001F0A 803E[0E1A]50                   cmp     byte [num_cyln], 80
00001F0F 75F5                           jnz     short gotother
00001F11 B609                           mov     dh, 9 ; ff288  ; assume 2.88 MB drive
00001F13 803E[101A]24                   cmp     byte [sec_trk], 36 ; is it ?
00001F18 744B                           jz      short nextdrive       ; yeah, go update

                                        ; 12/05/2019 (ff144 type will not be used -compatibility problem-)
                                        ; 08/01/2018 - Retro DOS v4.0 feature only ! for 1.44MB diskettes
                                        ;mov    dh, ff144
                                        ;cmp    byte [sec_trk], 18
                                        ;je     short nextdrive

00001F1A 803E[101A]0F                   cmp     byte [sec_trk], 15
00001F1F 740B                           jz      short got96

00001F21 803E[101A]09                   cmp     byte [sec_trk], 9
00001F26 75DE                           jnz     short gotother

00001F28 B602                           mov     dh, 2           ; ffSmall
00001F2A EB39                           jmp     short nextdrive
                                ; -------------------------------------------------------------------------

                                got96:
00001F2C B601                           mov     dh, 1           ; ff96tpi
00001F2E EB35                           jmp     short nextdrive

                                ; -------------------------------------------------------------------------

                                        ; 10/12/2023 - Retro DOS v5.0 IO.SYS/IBMBIO.COM
                                        ; check change line feature (and set fhave96 if there is)
                                        ; (common procedure for 'eotok:' and 'noparmsfromrom:')
                                chk_changeline:
00001F30 59                             pop     cx ; near call return address

                                        ; (pop es, dx, di for 'eotok' and 'noparmsfromrom' procs)
00001F31 07                             pop     es ; es=cs=ds ; 21/12/2022
                                        ;pop    cx      ; (10/12/2023)
00001F32 5A                             pop     dx
00001F33 5F                             pop     di ; BDS address/offset

00001F34 51                             push    cx ; near call return address

                                        ;xor    cx, cx ; 0
                                        ;push   cx
00001F35 52                             push    dx

00001F36 B415                           mov     ah, 15h
00001F38 CD13                           int     13h             ; DISK - DISK -       GET TYPE (AT,XT2,XT286,CONV,PS)
                                                                ; DL = drive ID
                                                                ; Return: CF set on error, AH =      disk type (3 = hard drive)
                                                                ; CX:DX = number of sectors on the media
00001F3A 5A                             pop     dx
                                        ;pop    cx
00001F3B B90000                         mov     cx, 0
00001F3E 720D                           jc      short chk_chgl_1

00001F40 80FC02                         cmp     ah, 2           ; is there changeline?
00001F43 7508                           jne     short chk_chgl_2 ; *

00001F45 80C902                         or      cl, 2
                                        ;or     cl, ah ; 2
00001F48 C606[7700]01                   mov     byte [fhave96], 1 ; fchangeline
                                        ; cf = 0
                                chk_chgl_1:
                                chk_chgl_2:
00001F4D C3                             retn

                                ;chk_chgl_2:       ; *
                                ;       ; 10/12/2023
                                ;       ; ah = 1 ; harddisk type (ah = 3) return not possible here for floppies
```

```
6334                                ;           ;stc
6335                                ;           ; cf = 1
6336                                ;           retn
6337                                
6338                                ; ---------------------------------------------------------------------------
6339                                
6340                                ; we have an old rom, so we either have a 48tpi or 96tpi drive. if the drive
6341                                ; has changeline, we assume it is a 96tpi, otherwise we treat it as a 48tpi.
6342                                
6343                                noparmsfromrom:
6344                                            ; 10/12/2023
6345                                            ; !!!
6346                                            ; (following pops are moved to 'chk_changeline' procedure)
6347                                            ;
6348                                            ; 20/12/2022
6349                                            ;pop    es ; ((*))
6350                                            ;;pop   cx      ; (10/12/2023)
6351                                            ;pop    dx
6352                                            ;pop    di
6353                                
6354                                            ; 20/12/2022
6355                                            ;pop    ds
6356                                ; 10/12/2023
6357                                %if 0
6358                                            ; 10/12/2023
6359                                            ;xor    cx, cx ; 0
6360                                            ;push   cx
6361                                            push    dx
6362                                
6363                                
6364                                            mov     ah, 15h
6365                                            int     13h             ; DISK - DISK -       GET TYPE (AT,XT2,XT286,CONV,PS)
6366                                                                    ; DL = drive ID
6367                                                                    ; Return: CF set on error, AH =     disk type (3 = hard drive)
6368                                                                    ; CX:DX= number of sectors on the media
6369                                            ; 10/12/2023
6370                                            pop     dx
6371                                            ;pop    cx
6372                                            mov     cx, 0 ; 12/12/2023
6373                                            jc      short nextdrive
6374                                
6375                                            cmp     ah, 2           ; is there changeline?
6376                                            jnz     short nextdrive
6377                                
6378                                            or      cl, 2
6379                                            mov     byte [fhave96], 1 ; fchangeline
6380                                %endif
6381                                            ; 10/12/2023
6382 00001F4E E8DFFF                           call    chk_changeline
6383 00001F51 7212                             jc      short nextdrive
6384                                
6385                                            ; change line support, [fhave96] = 1
6386                                
6387 00001F53 C606[0E1A]50                     mov     byte [num_cyln], 80
6388 00001F58 B601                             mov     dh, 1           ; ff96tpi
6389 00001F5A B00F                             mov     al, 15
6390 00001F5C 3A06[2C01]                       cmp     al, [eot]
6391 00001F60 7603                             jbe     short nextdrive
6392 00001F62 A2[2C01]                         mov     [eot], al
6393                                ; ---------------------------------------------------------------------------
6394                                
6395                                ;eot_ok2:
6396                                nextdrive:
6397                                            ; 10/12/2023
6398                                            ; ch = 0, cl = 2 or 0
6399                                
6400 00001F65 80C920                           or      cl, 20h ; fi_own_physical
6401                                                                    ; set this true       for all drives
6402 00001F68 88D7                             mov     bh, dl          ; save int13 drive number
6403                                
6404                                ; we need to do special things if we have a single drive system and are setting
6405                                ; up a logical drive. it needs to have the same int13 drive number as its
6406                                ; counterpart, but the next drive letter. also reset ownership flag.
6407                                ; we detect the presence of this situation by examining the flag single for the
6408                                ; value 2.
6409 00001F6A 803E[7800]02                     cmp     byte [single], 2
6410 00001F6F 7505                             jnz     short not_special
6411 00001F71 FECF                             dec     bh              ; int13 drive number same for logical drive
6412 00001F73 80F120                           xor     cl, 20h ; fi_own_physical
6413                                                                    ; reset ownership flag for logical drive
6414                                not_special:
6415                                
6416                                ; the values that we put in for BDS_RBPB.BPB_HEADS and
6417                                ; BDS_RBPB.BPB_SECTORSPERTRACK will only remain if the
6418                                ; form factor is of type "ffother".
6419                                
6420                                            ;xor    ax, ax          ; fill BDS for drive
6421                                            ;mov    al, [num_heads]
6422                                            ; 10/12/2023 - Retro DOS v5.0 (PCDOS 7.1 IBMBIO.COM) ; *
6423 00001F76 A1[0C1A]                         mov     ax, [num_heads]
6424                                            ;mov    [di+36h], ax    ; [di+BDS.rheads]
6425 00001F79 894552                           mov     [di+52h], ax    ; [di+BDS.rheads] ; *
6426 00001F7C 31C0                             xor     ax, ax          ; *
6427 00001F7E A0[101A]                         mov     al, [sec_trk]
6428                                            ;mov    [di+34h], ax    ; [di+BDS.rsecpertrack]
6429 00001F81 894550                           mov     [di+50h], ax    ; [di+BDS.rsecpertrack] ; *
6430                                            ;mov    [di+23h], cx    ; [di+BDS.flags]
6431 00001F84 894D3F                           mov     [di+3Fh], cx    ; [di+BDS.flags] ; *
6432                                            ;mov    [di+22h], dh    ; [di+BDS.formfactor]
6433 00001F87 88753E                           mov     [di+3Eh], dh    ; [di+BDS.formfactor] ; *
6434 00001F8A 885505                           mov     [di+5], dl      ; [di+BDS.drivelet]
6435 00001F8D 887D04                           mov     [di+4], bh      ; [di+BDS.drivenum]
6436                                            ;mov    bl, [num_cyln]
6437                                            ;mov    [di+25h], bl    ; [di+BDS.cylinders]
6438                                            ; 10/12/2023
6439 00001F90 A1[0E1A]                         mov     ax, [num_cyln]
6440 00001F93 894541                           mov     [di+41h], ax    ; [di+BDS.cylinders] ; *
6441                                
6442 00001F96 803E[7800]01                     cmp     byte [single], 1 ; Special case for single drive system
6443 00001F9B 7510                             jnz     short no_single
6444                                            ;mov    byte [single], 2 ; Don't forget we have
6445                                                                    ; single drive system
6446                                            ; 10/12/2023
6447 00001F9D FE06[7800]                       inc     byte [single]  ; [single] = 2
6448                                            ; 18/12/2022
6449 00001FA1 80C910                           or      cl, 10h
6450                                            ;or     cx, 10h         ; fi_am_mult
6451                                                                    ; set that this       is one of several drives
6452                                            ;or     [di+23h], cx    ; [di+BDS.flags]
6453 00001FA4 094D3F                           or      [di+3Fh], cx    ; [di+BDS.flags] ; *
6454                                                                    ; save flags
6455 00001FA7 8B3D                             mov     di, [di]        ; [di+BDS.link]
6456                                                                    ; move to next BDS in list
6457 00001FA9 FEC2                             inc     dl              ; add a number
```

```
6458 00001FAB EBB8                      jmp     short nextdrive       ; Use same info       for BDS as previous
6459                                    ; ---------------------------------------------------------------------------
6460
6461                                    no_single:
6462                                            ;inc    dl
6463                                            ; 18/12/2022
6464 00001FAD 42                                inc     dx
6465 00001FAE E97CFE                            jmp     loop_drive
6466                                    ; ---------------------------------------------------------------------------
6467
6468                                            ; 11/12/2023 - Retro DOS v5.0 IO.SYS/IBMBIO.COM
6469                                            ; (PCDOS 7.1 IBMBIO.COM - BIOSDATA:21E8h)
6470                                    done_drives:
6471                                            ;mov    word [di+BDS.link], -1
6472 00001FB1 C705FFFF                         mov     word [di], -1  ; set link to null
6473
6474                                    ; set up all the hard drives in   the system
6475
6476                                            ; 20/12/2022
6477                                            ; 28/12/2018 - Retro DOS v4.0 (MSDOS 6.21)
6478                                    dohard:
6479 00001FB5 8A36[5D1A]                        mov     dh, [hnum]
6480 00001FB9 08F6                              or      dh, dh          ; done if no hardfiles
6481 00001FBB 7459                              jz      short static_configure
6482 00001FBD B280                              mov     dl, 80h
6483                                    dohard1:
6484 00001FBF 52                                push    dx
6485 00001FC0 8B3E[601A]                        mov     di, [end_of_bdss]
6486 00001FC4 8A1E[7500]                        mov     bl, [drvmax]
6487 00001FC8 B700                              mov     bh, 0           ; first primary partition (or active)
6488 00001FCA E89A01                            call    sethard
6489 00001FCD 7208                              jb      short hardfile_err
6490 00001FCF E86508                            call    dmax_check      ; error if already 26 drives
6491 00001FD2 7303                              jnb     short hardfile_err
6492 00001FD4 E87508                            call    xinstall_bds    ; insert new bds into linked list
6493                                    hardfile_err:
6494 00001FD7 5A                                pop     dx
6495                                            ;inc    dl              ; next hard drive
6496                                            ; 12/12/2023
6497 00001FD8 42                                inc     dx
6498 00001FD9 FECE                              dec     dh
6499 00001FDB 75E2                              jnz     short dohard1
6500
6501                                    ; end of physical drive   initialization
6502
6503                                    ; *** do not change the position of the following statement.
6504                                    ; *** domini routine will use [drvmax] value for the start of the logical
6505                                    ; *** drive number of mini disk(s).
6506
6507 00001FDD E8CA06                            call    domini          ; for setting up mini disks, if found
6508
6509                                    ; -- begin added section
6510
6511 00001FE0 8A36[5D1A]                        mov     dh, [hnum]      ; we already know this is >0
6512 00001FE4 B280                              mov     dl, 80h
6513                                    dohardx1:
6514 00001FE6 B701                              mov     bh, 1           ; do all subsequent primary partitions
6515                                    dohardx2:
6516 00001FE8 52                                push    dx
6517 00001FE9 53                                push    bx
6518 00001FEA 8B3E[601A]                        mov     di, [end_of_bdss]
6519 00001FEE 8A1E[7500]                        mov     bl, [drvmax]
6520 00001FF2 E87201                            call    sethard
6521 00001FF5 720E                              jb      short dohardx4 ; move to next hardfile if error
6522 00001FF7 E83D08                            call    dmax_check      ; make sure <=26 drives
6523 00001FFA 7309                              jnb     short dohardx4 ; skip if error
6524 00001FFC E84D08                            call    xinstall_bds    ; insert new bds into linked list
6525 00001FFF 5B                                pop     bx              ; get partition number
6526 00002000 5A                                pop     dx              ; restore physical drive counts
6527 00002001 FEC7                              inc     bh
6528 00002003 EBE3                              jmp     short dohardx2 ; keep looping until we fail
6529                                    ; ---------------------------------------------------------------------------
6530
6531                                    dohardx4:
6532 00002005 5B                                pop     bx              ; unjunk partition number from stack
6533 00002006 5A                                pop     dx              ; restore physical drive counts
6534                                            ;inc    dl              ; next hard drive
6535                                            ; 12/12/2023
6536 00002007 42                                inc     dx
6537 00002008 FECE                              dec     dh
6538 0000200A 75DA                              jnz     short dohardx1
6539
6540                                    ; -- end changed section
6541
6542                                    ;******************************************************************************
6543                                    ; if more than 2 diskette drives on the system, then it is necessary to remap
6544                                    ; the bds chain to adjust the logical drive num (drive letter) with greater
6545                                    ; than two diskette drives
6546                                    ;
6547                                    ; new scheme:      if more than 2 disktte drives, first map the bds structure
6548                                    ;           as usual and then rescan the bds chain to adjust the  drive
6549                                    ;           letters. to do this, scan for disk drives and assign logical
6550                                    ;           drive number starting from 2 and then rescan diskette drives
6551                                    ;           and assign next to the last logical drive number of last disk
6552                                    ;           drive to the 3rd and 4th diskette drives.
6553                                    ;******************************************************************************
6554
6555 0000200C 803E[2501]02                         cmp     byte [dsktnum], 2 ; >2 diskette drives
6556                                            ;jbe    short static_configure ; no - no need for remapping
6557 00002011 7603                              jbe     short no_remap
6558 00002013 E8D500                            call    remap           ; remap bds chain to adjust driver letters
6559                                    no_remap:
6560
6561                                    ; End of drive initialization.
6562
6563                                    ; ---------------------------------------------------------------------------
6564
6565                                    ;we now decide, based on the configurations available so far, what
6566                                    ;code or data we need to keep as a stay resident code. the following table
6567                                    ;shows the configurations under consideration. they are listed in the order
6568                                    ;of their current position memory.
6569                                    ;
6570                                    ;configuration will be done in two ways:
6571                                    ;
6572                                    ;first, we are going to set "static configuration". static configuration will
6573                                    ;consider from basic configuration to endof96tpi configuration. the result
6574                                    ;of static configuration will be the address the dynamic configuration will
6575                                    ;use to start with.
6576                                    ;
6577                                    ;secondly, "dynamic configuration" will be performed. dynamic configuration
6578                                    ;involves possible relocation of code or data. dynamic configuration routine
6579                                    ;will take care of bdsm tables and at rom fix module thru k09 suspend/resume
6580                                    ;code individually. after these operation, [dosdatasg] will be set.
6581                                    ;this will be the place sysinit routine will relocate msdos module for good.
```

```
                                ; -- begin changed section
                                ;
                                ;   1.      basic configuration for msbio (endfloppy)
                                ;   2.   end96tpi  ; a system that supports "change line error"
                                ;   3.      end of bdss   ; end of bdss for hard disks
                                ;   4.      endatrom      ;some of at rom fix module.
                                ;   5.      endcmosclockset;supporting program for cmos clock write.
                                ;   6.      endk09        ;k09 cmos clock module to handle suspend/resume operation.
                                ;

                                ; 02/10/2022 - Retro DOS v4.0 (MSDOS v5.0 IO.SYS)

                                static_configure:
00002016 8B3E[601A]                             mov     di, [end_of_bdss]
0000201A 81FF[4C08]                             cmp     di, bdss        ; 19/10/2022
                                                ;cmp    di, offset bdss         ; did we allocate any hard drive bdss?
0000201E 750D                                   jnz     short dynamic_configure         ; that's the end, then
                                                ; 18/10/2022
00002020 BF[4C08]                               mov     di, end96tpi
                                                ;mov    di, offset harddrv ; end96tpi
                                                                        ; keep everything up to end96tpi
00002023 803E[7700]00                           cmp     byte [fhave96], 0
00002028 7503                                   jnz     short dynamic_configure

0000202A BF[3808]                               mov     di, endfloppy
                                dynamic_configure:
                                                ; 20/12/2022
                                                ;push   cs
                                                ;pop    es

                                                ; 10/12/2023
0000202D FC                                     cld     ; clear direction flag is not necessary here !?
                                                        ; because there will not be a running program
                                                        ; which will set direction flag as backward (std)

                                ; -- end changed section

                                                ; 20/12/2022
                                                ; ds = cs <> es
                                                ; ss = 0
                                                ; sp = 700h

                                                ; 13/12/2023
0000202E BE00F0                                 mov     si, 0F000h
00002031 8EC6                                   mov     es, si          ; ES -> ROM BIOS segment

00002033 803E[AF05]FC                           cmp     byte [model_byte], 0FCh ; AT ?
                                                ;jnz    short checkcmosclock
                                                ; 10/12/2023
00002038 751E                                   jnz     short checkcompaqbug ; no
0000203A 803E[5D1A]00                           cmp     byte [hnum], 0 ; No hard file?
                                                ;jz     short checkcmosclock
0000203F 7417                                   jz      short checkcompaqbug
00002041 97                                     xchg    ax, di          ; save allocation pointer in ax
                                                ; 13/12/2023
                                                ;mov    si, 0F000h
                                                ;mov    es, si          ; ES -> ROM BIOS segment
00002042 BE[661A]                               mov     si, bios_date   ; "01/10/84"
00002045 BFF5FF                                 mov     di, 0FFF5h      ; ROM BIOS string is at       F000:FFF5
00002048 B90900                                 mov     cx, 9           ; bdate_l
                                                                        ; Only patch ROM for bios 01/10/84
0000204B F3A6                                   repe cmpsb              ; check for date + zero       on end
0000204D 97                                     xchg    ax, di          ; restore allocation pointer

                                ; M015 -- begin changes

                                                ;jnz    short checkcmosclock
                                                ; 02/10/2022
0000204E 7508                                   jnz     short checkcompaqbug

                                ; install at rom fix

                                                ; 19/10/2022
                                                ;mov    cx, offset endatrom
00002050 B9[2018]                               mov     cx, endatrom
                                                ;mov    si, offset ibm_disk_io
00002053 BE[F216]                               mov     si, ibm_disk_io
00002056 EB46                                   jmp     short install_int13_patch
                                ; ---------------------------------------------------------------------------

                                ; M065 -- begin changes
                                ;
                                ; On certain systems with Western Digital disk controllers, the
                                ; following detection scheme caused an unpredictable and serious
                                ; failure. In particular, they've implemented a nonstandard
                                ; Int13(ah=16h) which reconfigures the hard drive, depending on
                                ; what happens to be at es:[bx] and other memory locations indexed
                                ; off of it.
                                ;
                                ; Compaq was unable to tell us exactly which kind of systems have
                                ; the bug, except that they guarantee that the bug was fixed in
                                ; ROM BIOSs dated 08/04/86 and later. We'll check for the COMPAQ
                                ; string, and then look for date codes before 08/04/86 to decide
                                ; when to install the hook.

                                ;checkcmosclock:
                                ; 02/10/2022
                                checkcompaqbug:
                                                ; 20/12/2022
                                                ; es = 0F000h
                                                ;mov    ax, 0F000h      ; point to ROM BIOS
                                                ;mov    es, ax

                                                ; 19/10/2022
00002058 26813EEAFF434F                         cmp     word [es:0FFEAh], 'CO'
                                                ;cmp    word ptr es:0FFEAh, 'OC' ; look for COMPAQ
0000205F 754B                                   jnz     short not_compaq_patch
00002061 26813EECFF4D50                         cmp     word [es:0FFECh], 'MP'
                                                ;cmp    word ptr es:0FFECh, 'PM'
00002068 7542                                   jnz     short not_compaq_patch
0000206A 26813EEEFF4151                         cmp     word [es:0FFEEh], 'AQ'
                                                ;cmp    word ptr es:0FFEEh, 'QA'
00002071 7539                                   jnz     short not_compaq_patch

                                ; We're running on a COMPAQ. Now look at the date code.

00002073 26A1FBFF                               mov     ax, [es:0FFFBh]         ; get year
00002077 86E0                                   xchg    ah, al

                                ; 11/12/2023
                                ; (PCDOS 7.1 IBMBIO.COM - BIOSDATA:22B9h)
                                %if 0
                                                cmp     ax, 3836h       ; '68' (NASM syntax) (('86' in MASM syntax))
```

```
                                        ja      short not_compaq_patch
                                        jz      short chkcompaqbug1
                                        cmp     ax, 3739h       ; '97'
                                        jbe     short not_compaq_patch
                                        stc
                                chkcompaqbug1:
                                        jb      short do_compaq_patch
                                        mov     ax, [es:0FFF5h]
                                        xchg    ah, al
                                        cmp     ax, 3038h       ; '80'
                                        ja      short not_compaq_patch
                                        jb      short do_compaq_patch
                                        mov     ax, [es:0FFF8h]
                                        xchg    ah, al
                                        cmp     ax, 3034h       ; '40'
                                        jnb     short not_compaq_patch
                                do_compaq_patch:
                                %endif
                                        ; 11/12/2023
                                        ; (MSDOS 6.22 IO.SYS - BIOSDATA:1C85h)

00002079 3D3638                         cmp     ax, 3836h ; 02/10/2022 (NASM syntax)
                                        ;cmp    ax, '86'        ; 3836h
                                                                ; is it 86?
0000207C 772E                           ja      short not_compaq_patch
0000207E 7218                           jb      short do_compaq_patch
00002080 26A1F5FF                       mov     ax, [es:0FFF5h]       ; get month
00002084 86E0                           xchg    ah, al
00002086 3D3830                         cmp     ax, 3038h ; 02/10/2022 (NASM syntax)
                                        ;cmp    ax, '08'        ; 3038h
                                                                ; is it 08?
00002089 7721                           ja      short not_compaq_patch
0000208B 720B                           jb      short do_compaq_patch
0000208D 26A1F8FF                       mov     ax, [es:0FFF8h]       ; get day
00002091 86E0                           xchg    ah, al
00002093 3D3430                         cmp     ax, 3034h ; 02/10/2022 (NASM syntax)
                                        ;cmp    ax, '04'        ; 3034h
                                                                ; is it 04?
00002096 7314                           jnb     short not_compaq_patch

                                do_compaq_patch:
00002098 B9[3D18]                       mov     cx, end_compaq_i13hook
                                        ;mov    si, endatrom
                                        ; 11/12/2023
0000209B BE[2018]                       mov     si, compaq_disk_io ; endatrom

                                install_int13_patch:
0000209E 0E                             push    cs
0000209F 07                             pop     es
                                        ; 18/10/2022
000020A0 893E[B400]                     mov     [Orig13], di   ; set new rom bios int 13 vector
000020A4 8C0E[B600]                     mov     [Orig13+2], cs
000020A8 29F1                           sub     cx, si         ; size of rom fix module
000020AA F3A4                           rep movsb              ; relocate it

                                ; M065 -- end changes

                                ; ----------------------------------------------------------------------------
                                not_compaq_patch:               ; M065
                                        ; 17/10/2022
                                checkcmosclock:
                                        ; 18/10/2022
000020AC 0E                             push    cs
000020AD 07                             pop     es

                                        ; 20/12/2022
                                        ; ds = cs = es
                                        ; ss = 0
                                        ; sp = 700h

                                ; 09/08/2023 - Retro DOS v4.2 IO.SYS (optimization)
                                %if 0
                                        cmp     byte [havecmosclock], 1 ; cmos clock exists?
                                        jnz     short checkk09 ; no

                                        mov     word [daycnttoday], di
                                        ;mov    word ptr ds:daycnttoday, di ; set the address for mschar
                                        mov     cx, 209 ; enddaycnttoday - daycnt_to_day
                                        mov     si, daycnt_to_day
                                        rep movsb
                                        mov     word [bintobcd], di
                                        ;mov    word ptr ds:bintobcd, di ; set the address for msclock
                                                                ; let original segment stay
                                        ;mov    cx, 11  ; endcmosclockset - bin_to_bcd
                                        ; 08/08/2023
                                        mov     cl, 11
                                        mov     si, bin_to_bcd
                                        rep movsb
                                %endif

                                ; 09/08/2023 - Retro DOS v4.2 IO.SYS (optimization)
                                ; PCDOS 7.1 IBMBIO.COM - BIOSDATA:22F4h
                                        ;push   cs
                                        ;pop    es
                                checkk09:
000020AE 57                             push    di ; ? ; save ? ; 21/12/2022

                                ; 13/12/2023 - Retro DOS v4.2 IO.SYS
                                ; (MSDOS 6.22 IO.SYS - BIOSDATA:1CDAh)
                                %if 0

                                        mov     ax, 4101h       ; wait for bh=es:[di]
                                        mov     bl, 1           ; wait for 1 clock tick
                                        mov     bh, [es:di]
                                        stc                     ; Assume we will fail
                                        int     15h             ; SYSTEM - WAIT       ON EXTERNAL EVENT (CONVERTIBLE)
                                                                ; AL = condition type, BH = condition compare or mask value
                                                                ; BL = timeout value times 55 milliseconds, 00h     means no timeout
                                                                ; DX = I/O port       address if AL bit 4 set
                                                                ; 11/12/2023
                                                                ; ES:DI = user byte if AL bit 4 clear
                                %endif
                                        ; 13/12/2023 - Retro DOS v5.0 IBMBIO.COM/IO.SYS
                                        ; (PCDOS 7.1 IBMBIO.COM - BIOSDATA:1CDAh)

                                        ; ........

000020AF B80041                         mov     ax, 4100h       ; wait for any external event (al=0)
000020B2 B304                           mov     bl, 4           ; wait for 4 clock ticks
000020B4 F9                             stc                     ; Assume we will fail
000020B5 CD15                           int     15h             ; SYSTEM - WAIT ON EXTERNAL EVENT (CONVERTIBLE)
                                                                ; AL = condition type, BH = condition compare or mask value
                                                                ; BL = timeout value times 55 milliseconds, 00h means no timeout
                                                                ; DX = I/O port address if AL bit 4 set
```

```
                                        ; ........
000020B7 5F                             pop     di ; ?
000020B8 721B                           jc      short configdone ; 21/12/2022

000020BA C606[7900]01                   mov     byte [fhavek09], 1
                                                        ; remember we have a k09 type
000020BF 1E                             push    ds
000020C0 31C0                           xor     ax, ax
000020C2 8ED8                           mov     ds, ax

000020C4 893EB001                       mov     [6Ch*4], di
                                        ;mov    ds:1B0h, di     ; [6Ch*4]
                                                        ; new int 6Ch handler
                                        ;mov    word ptr ds:1B2h, cs ; [6Ch*4+2]
000020C8 8C0EB201                       mov     word [6Ch*4+2], cs
000020CC 1F                             pop     ds
                                        ; 20/12/2022
                                        ; ds = cs = es
                                        ;mov    si, int6c
                                        ;mov    cx, endk09-int6c ; 459
                                        ;;mov   cx, 459         ; endk09 - int6c
                                                        ; size of k09 routine
                                        ; 08/08/2023 - Retro DOS v4.2 IO.SYS (optimization)
                                        ; PCDOS 7.1 IBMBIO.COM - BIOSDATA:2315h
000020CD BE[3E18]                       mov     si, int_6Ch
000020D0 B9BC01                         mov     cx, endk09-int_6Ch ; 461 in PCDOS 7.1 IBMBIO.COM
000020D3 F3A4                           rep movsb               ;
                                                        ; set up config       stuff for sysinit
                                ; ----------------------------------------------------------------------------
                                ; Set up config stuff for SYSINIT

                                ; 17/10/2022
                                ;SETDRIVE equ SetDrive - DOSBIOSEG_2C7h ; (4D7h for MSDOS 5.0 IO.SYS)
                                ;GETBP equ GetBp - DOSBIOSEG_2C7h ; (606h for MSDOS 5.0 IO.SYS)
                                ; 09/12/2022
                                SETDRIVE equ SetDrive
                                GETBP equ GetBp

                                        ; 17/10/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 IO.SYS)
                                configdone:
                                        ; 21/12/2022
                                        ; 20/03/2019
                                        ;push   cs              ; di is final ending address of      msbio.
                                        ;pop    ds

000020D5 83C70F                         add     di, 15          ; round (up) to paragraph
                                        ; 10/12/2022
                                        ;shr    di, 1
                                        ;shr    di, 1
                                        ;shr    di, 1
                                        ;shr    di, 1
000020D8 B104                           mov     cl, 4
000020DA D3EF                           shr     di, cl
                                        ; 10/12/2022
000020DC 83C770                         add     di, 70h ; KERNEL_SEGMENT (in fact: IO.SYS loading segment)
                                        ; 19/10/2022 - Temporary !
                                        ;db     81h, 0C7h, 70h, 0 ; add di, 0070h
000020DF 893E[0300]                     mov     [DosDataSg], di         ; where the dos data segment will be

                                ; 21/12/2022 - Retro DOS v4.0 (MSDOS 5.0 combined/single kernel file)

                                ; 19/03/2018 - No need to read remain clusters of MSDOS kernel because
                                        ; Retro DOS v2.0 boot sector has loaded all of the kernel file before.

                                        ; ("MSINIT.ASM" contains kernel file reading code here, below...)

                                ; 11/12/2023 - Retro DOS v5.0 (PCDOS 7.1 combined/single kernel file)

                                        ; (PCDOS 7.1 IBMBIO.COM - BIOSDATA:2332h)

                                        ; (("IBMDOS.COM" kernel file reading code here, below...))

                                ; ----------------------------------------------------------------------------
                                ; ----------------------------------------------------------------------------
                                %if 0
                                        mov     ax, [drvfat]    ; get drive and       fat id
                                        ; 22/12/2022
                                        ; Note: SETDRIVES uses AL (drive number) only
                                        mov     bp, SETDRIVE
                                        ;mov    bp, 5AEh ; 11/12/2023 (PCDOS 7.1 IBMBIO.COM)
                                        ;;mov   bp, 4D7h         ; set_drive (in       dosbios code segment)
                                                                ; at 2C7h:4D7h = 70h:2A47h
                                        push    cs              ; simulate far call
                                        call    call_bios_code ; get bds for drive
                                        mov     bp, GETBP       ; ensure valid bpb is present
                                        ;mov    bp, 6E4h ; 11/12/2023 (PCDOS 7.1 IBMBIO.COM)
                                        ;;mov   bp, 606h         ; GetBp (2C7h:606h = 70h:2B76h)
                                        push    cs
                                        call    call_bios_code

                                ; resort to funky old segment definitions for now

                                        ; 22/12/2022
                                        ;push   es              ; copy bds to ds:di
                                        ;pop    ds

                                ; the following read of es:0000 was spurious anyway. Should look into it.
                                ;
                                ; hmmmmmm. j.k. took out a call to getfat right here a while
                                ;       back. Apparently it was what actually setup es: for the following
                                ; cas----

                                        ; 22/12/2022
                                        ;xor    di, di
                                        ;mov    al, [es:di]     ; get fat id byte
                                        ;;mov   byte ptr es:drvfat+1, al ; save fat byte
                                        ;mov    [es:drvfat+1], al
                                        ;mov    ax, [es:drvfat]

                                        ; 22/12/2022
                                        ; ds = cs
                                ;;;     mov     al, [drvfat]

                                ; cas -- why do a SECOND setdrive here???

                                        ; 22/12/2022
                                        ;push   es              ; save whatever's in es
                                        ;push   ds              ; copy bds to es:di
                                        ;pop    es
                                        ;push   cs              ; copy Bios_Data to ds
                                        ;pop    ds

                                ; 22/12/2022
```

```
;;;     mov     bp, SETDRIVE
;;;     ;mov    bp, 4D7h        ; SetDrive (2C7h:47Dh = 70h:2A47h)
;;;     push    cs              ; simulate far call
;;;     call    call_bios_code ; get correct bds for this drive

        ; 22/12/2022
        ;push   es              ; copy bds back to ds:di
        ;pop    ds
        ;pop    es              ; pop whatever was in es

; Now we load in the MSDOS.SYS file

; 22/12/2022
; -----
;       mov     bx, [di+6]      ; [di+BDS.BDS_BPB.BPB_BYTESPERSECTOR]
;       mov     [cs:md_sectorsize], bx; used by get_fat_sector proc.
;       mov     bl, [di+1Fh]    ; [di+BDS.fatsiz]
;                               ; get size of fat on media
;       ;mov    es:16DEh, bl
;       mov     [es:fbigfat], bl
;       mov     cl, [di+8]
;       mov     ax, [di+17h] ; [di+BDS.BDS_BPB.BPB_HIDDENSECTORS]
;       ;sub    es:16D8h, ax
;       sub     [es:bios_l], ax       ; subtract hidden sectors since we
;                               ; need a logical sector number that will
;                               ; be used by getclus(diskrd procedure)
;       mov     ax, [di+19h] ; [di+BDS.BDS_BPB.BPB_HIDDENSECTORS+2]
;       ;sbb    es:16DAh, ax
;       sbb     [es:bios_h], ax       ; subtract upper 16 bits of sector num
; -----

        ; 11/12/2023 - Retro DOS v5.0 IO.SYS/IBMBIO.COM
; -----; 22/12/2022
        mov     bx, [es:di+6]  ; [di+BDS.BDS_BPB.BPB_BYTESPERSECTOR]
        mov     [md_sectorsize], bx ; used by get_fat_sector proc.
        ; 11/12/2023 ; *
        mov     bl, [es:di+3Bh]       ; [di+BDS.fatsiz] ; *
        ;mov    bl, [es:di+1Fh]       ; [di+BDS.fatsiz]
                                ; get size of fat on media
        mov     [fbigfat], bl
        mov     cl, [es:di+8]
        mov     ax, [es:di+17h] ; [di+BDS.BDS_BPB.BPB_HIDDENSECTORS]
        sub     [First_Data_Sector], ax ; *
        ;sub    [bios_l], ax   ; subtract hidden sectors since we
                                ; need a logical sector number that will
                                ; be used by getclus(diskrd procedure)
        mov     ax, [es:di+19h] ; [di+BDS.BDS_BPB.BPB_HIDDENSECTORS+2]
        sbb     [First_Data_Sector+2], ax ; *
        ;sbb    [bios_h], ax   ; subtract upper 16 bits of sector num
; ------

        xor     ch, ch  ; cx = sectors/cluster

; the boot program has left the directory at 0:500h

        ; 11/12/2023 - - Retro DOS v5.0 IBMBIO.COM/IO.SYS
        ;push   di
        push    ds
        ;xor    di, di
        ;mov    ds, di
        xor     bx, bx ; 0
        mov     ds, bx
        mov     bx, [53Ah]
        ;mov    bx, ds:53Ah           ; (First cluster of the 2nd dir entry
                                ; of root directory in the buffer at 500h)
        pop     ds
        mov     si, [firstcluster_hw] ; 11/12/2023
                                ; (32 bit cluster number for FAT32 fs)
        ;pop    ds
        ;pop    di

        ; 12/12/2023
        ; (PCDOS 7.1 IBMBIO.COM - BIOSDATA:2397h)
        ; ...........
        ; ds = cs
        mov     al, [fbigfat]
        push    ax              ; (*) save fbigfat flags
        mov     al, [drvfat]
        or      al, [Boot_Drv]
        jnz     short boot_drv_fixed ; hard disk
boot_drv_removable:                     ; calculate cluster count and set fbig or fbigbig flag
        push    bx              ; for removable drives
        push    cx
        ; 28/12/2023
        ;push   dx ; (not necessary)

        ; 12/12/2023
        push    es
        pop     ds

        mov     ax, [di+0Eh]    ; [di+BDS.totalsecs16]
        xor     dx, dx
        or      ax, ax
        jnz     short prep_totalsecs_ok
        mov     ax, [di+1Bh]    ; [di+BDS.totalsecs32]
        mov     dx, [di+1Dh]
prep_totalsecs_ok:
        sub     ax, [di+9]      ; [di+BDS.resectors]
        sbb     dx, 0
        push    ax
        push    dx
        mov     bx, [di+11h]    ; [di+BDS.fatsecs16]
        xor     ax, ax
        or      bx, bx
        jnz     short prep_fatsecs_ok
        mov     bx, [di+1Fh]    ; [di+BDS.fatsecs32]
        mov     ax, [di+21h]
prep_fatsecs_ok:
        mov     cl, [di+0Bh]    ; ax:bx = 32 bit count of FAT sectors
                                ; [di+BDS.fats]
        xor     ch, ch
        mul     cx
        xchg    ax, cx
        mul     bx
        add     cx, dx
        mov     bx, ax          ; cx:bx = total (2*) fat sectors
        pop     dx
        pop     ax              ; dx:ax = totals sectors - reserved sectors
        sub     ax, bx
        sbb     dx, cx          ; dx:ax = data sectors (includes root dir sectors)
        mov     bx, [di+0Ch]    ; [di+BDS.direntries]
        add     bx, 15          ; 16 directory entries per sector
                                ; (round up sector count by adding 15)
        mov     cl, 4           ; (rounded) dir entries / 16
```

```
7078                                    shr     bx, cl
7079                                    xor     cx, cx
7080                                    sub     ax, bx
7081                                    sbb     dx, cx          ; dx:ax = data sectors (except root directory sectors)
7082                                                            ; (will be used for cluster count calculation)
7083                                    mov     cl, [di+8]      ; [di+BDS.secperclus]
7084
7085                                    ; 12/12/2023
7086                                    push    cs
7087                                    pop     ds
7088
7089                                    push    ax              ; 32 bit division (data sectors / sector per cluster)
7090                                    mov     ax, dx
7091                                    xor     dx, dx
7092                                    div     cx
7093                                    mov     bx, ax
7094                                    pop     ax
7095                                    div     cx
7096                                    or      bx, bx          ; 32 bit cluster count if bx > 0
7097                                    jnz     short set_fbigbig_flag ; too big cluster number
7098                                    cmp     ax, 0FFF6h
7099                                    jb      short set_fbig_flag
7100                            set_fbigbig_flag:
7101                                    or      byte [fbigfat], 20h ; FAT32 ; fbigbig
7102                                    jmp     short set_fbig_flag_ok
7103                            ; ---------------------------------------------------------------------------
7104
7105                            set_fbig_flag:
7106                                    cmp     ax, 0FF6h       ; 4096-10
7107                                                            ; is this 16-bit fat?
7108                                    jb      short set_fbig_flag_ok ; no, small fat
7109                                    or      byte [fbigfat], 40h ; FAT16 ; fbig
7110                            set_fbig_flag_ok:
7111                                    ; 28/12/2023
7112                                    ;pop    dx
7113                                    pop     cx
7114                                    pop     bx
7115                            boot_drv_fixed:
7116                                    xor     di, di
7117
7118                                    ; cx = sectors/cluster
7119                                    ; si:bx = first cluster
7120                                    ; di = 0
7121
7122                                    ; ...........
7123                            loadit:
7124                                    mov     ax, SYSINITSEG ; 46Dh
7125                                    ;mov    ax, 544h        ; 11/12/2023 - PCDOS 7.1 IBMBIO.COM
7126                                    ;;mov   ax, 46Dh        ; sysinit segment
7127                                    mov     es, ax
7128                                    mov     es, [es:CURRENTDOSLOCATION] ; 09/12/2022
7129                                    ;mov    es, [es:271h]
7130
7131                                    call    getclus         ; read cluster at ES:DI (DI is updated)
7132
7133                            ; ---------------------------------------------------------------------------
7134
7135                                    ; 13/12/2023 - Retro DOS v5.0 (PCDOS 7.1) IBMBIO.COM
7136                                    ; (PCDOS 7.1 IBMBIO.COM - BIOSDATA:2431h)
7137
7138                                    ;test   byte [cs:fbigfat], 20h ; fbigbig ; FAT32 fs flag
7139                                    test    byte [fbigfat], 20h ; fbigbig ; FAT32 fs flag
7140                                    jz      short iseof
7141
7142                            eofbigbig:  ; si:bx = 32 bit cluster number
7143                                    cmp     si, 0FFFh
7144                                    jnz     short iseofx
7145                                    cmp     bx, 0FFF7h
7146                                    jmp     short iseofx
7147
7148                            ; ---------------------------------------------------------------------------
7149                                    ; 13/12/2023
7150                            iseof:
7151                                    ;;test  byte [cs:fbigfat], fbig
7152                                    ;test   byte [cs:fbigfat], 40h ; fbig
7153                                    ; 12/12/2023
7154                                    ; ds = cs
7155                                    test    byte [fbigfat], 40h ; fbig
7156                                    jnz     short eofbig
7157                                    cmp     bx, 0FF7h
7158                                    jmp     short iseofx
7159                            ; ---------------------------------------------------------------------------
7160
7161                            eofbig:
7162                                    cmp     bx, 0FFF7h
7163                            iseofx:
7164                                    jb      short loadit   ; keep loading until cluster = eof
7165
7166                            ; ---------------------------------------------------------------------------
7167
7168                                    ; 19/04/2024
7169                                    ; 28/12/2023
7170                                    pop     ax              ; (*) restore fbigfat flags
7171                                                            ; (after loading DOS kernel)
7172                                    ; 06/04/2024
7173                                    ;mov    [cs:fbigfat], al
7174                                    mov     [fbigfat], al
7175                            %endif
7176                            ; ---------------------------------------------------------------------------
7177                            ; ---------------------------------------------------------------------------
7178
7179                                    ; 19/04/2024
7180
7181 000020E3 E8FE04                    call    setdrvparms
7182
7183                                    ;;;jmp  far ptr 46Dh:267h ; jmp       SYSINIT_SEG:SYSINIT_START
7184                                    ;;jmp   far 46Dh:267h
7185                                    ; 12/12/2023
7186                                    ;jmp    far 544h:269h  ; (PCDOS 7.1 IBMBIO.COM)
7187
7188 000020E6 EA[6902]D904              jmp     SYSINITSEG:SYSINITSTART
7189
7190                            ; =============== S U B   R O U T I N E =======================================
7191
7192                            ; Following are subroutines to support resident device driver initialization
7193                            ;
7194                            ;M011 -- note:  deleted setup_bdsms and reset_bdsms here
7195
7196                            ;   M035 -- begin changed section
7197
7198                            ;****************************************************************************
7199                            ; module name: remap
7200                            ;
7201                            ; descriptive name: all the code for himem that could be separated from msbio
```

```
                                ;
                                ; function:  remap the bds chain to adjusted logical drive numbers (drive
                                ;        letters) if more than two diskette drives on the system.
                                ;
                                ;     scheme:  if more than 2 diskette drives, first map the bds structure
                                ;          as usual and then rescan the bds chain to adjust the drive
                                ;          letters. to do this, scan for disk drives and assign logical
                                ;          drive number starting from 2 and then rescan diskette drives
                                ;          and assign next to the last logical drive number of last disk
                                ;          drive to the 3rd and 4th diskette drives.
                                ;
                                ; input:       none
                                ; exit:    drive letters have been remapped in bds chain
                                ; exit error:  none
                                ; called from: msinit
                                ;
                                ; notes:  this function  will be called only if more than 2 diskettes are
                                ;     found in the system
                                ;     this function assumes that there are no more than 26 drives assigned
                                ;       this is guaranteed by the code that creates bdss for partitions
                                ;     this function assumes that the first entries in the chain are
                                ;      floppy drives, and all the rest are hard drives
                                ;     will alter the boot drive if necessary to reflect remapping
                                ;
                                ;*****************************************************************************

                                ; 17/10/2022
                                ; 02/10/2022
                                          ; 15/12/2023 - Retro DOS v5.0 (PCDOS 7.1) IBMBIO.COM
                                          ; (PCDOS 7.1 IBMBIO.COM - BIOSDATA:2464h)
                                          ; (MSDOS 6.22 IO.SYS - BIOSDATA:1D9Eh)

                                remap:            ; proc near

                                          ; 15/12/2023
                                          ; ds = cs
000020EB 8B3E[1901]                       ;mov    di, [cs:start_bds] ; get first bds
                                          mov     di, [start_bds]

                                ; search for 1st fixed disk physical drive num

                                drive_loop:
000020EF 807D0480                         cmp     byte [di+4], 80h ; [di+BDS.drivenum]
                                                                   ; first hard disk??
000020F3 7409                             jz      short fdrv_found ; yes,       continue
000020F5 8B3D                             mov     di, [di]         ; [di+BDS.link]
                                                                   ; get next bds,       assume segment
000020F7 83FFFF                           cmp     di, -1 ; 0FFFFh        ; last bds?
000020FA 75F3                             jnz     short drive_loop ; loop       if not
000020FC EB49                             jmp     short rmap_exit        ; yes, no hard drive on       system

                                ;---------------------------------------------------------------------------
                                ;first disk drive bds, now change the logical drive num to 2 and the subsequent
                                ;logical drive nums to 3, 4, 5 etc.
                                ;---------------------------------------------------------------------------

                                fdrv_found:
000020FE B002                             mov     al, 2          ; start with logical drv num=2
                                fdrv_loop:
00002100 884505                           mov     [di+5], al     ; [di+BDS.drivelet]
00002103 8B3D                             mov     di, [di]       ; [di+BDS.link]
                                                                 ; ds:di--> next       bds
                                          ;inc    al             ; set num for next drive
                                          ; 18/12/2022
00002105 40                               inc     ax
00002106 83FFFF                           cmp     di, 0FFFFh     ; last hard drive ?
00002109 75F5                             jnz     short fdrv_loop        ; no - assign more disk drives

                                ;---------------------------------------------------------------------------
                                ; now, rescan and find bds of 3rd floppy drive and assign next drive letter
                                ; in al to 3rd. if the current drive letter is past z, then do not allocate
                                ; any more.
                                ;---------------------------------------------------------------------------

                                          ;mov    di, [cs:start_bds] ; [start_bds]
                                          ; 15/12/2023
0000210B 8B3E[1901]                       mov     di, [start_bds]        ; get first bds
0000210F 8B3D                             mov     di, [di]       ; [di+BDS.link]
                                                                 ; ds:di-->bds2
                                          ;mov    ah, [cs:dsktnum] ; get number of floppies to remap
00002111 8A26[2501]                       mov     ah, [dsktnum]
00002115 80EC02                           sub     ah, 2          ; adjust for a:       & b:
                                remap_loop1:
00002118 8B3D                             mov     di, [di]       ; [di+BDS.link]
                                                                 ; set new num to next floppy
0000211A 884505                           mov     [di+5], al     ; [di+BDS.drivelet]
0000211D FEC0                             inc     al             ; new number for next floppy
0000211F FECC                             dec     ah             ; count down extra floppies
00002121 75F5                             jnz     short remap_loop1

                                ; now we've got to adjust the boot drive if we reassigned it

                                          ; 15/12/2023
                                          ;mov    al, [cs:drvfat]
00002123 A0[FA19]                         mov     al, [drvfat]
00002126 3C02                             cmp     al, 2          ; is it a: or b: ?
00002128 721D                             jb      short rmap_exit
                                          ;sub    al, [cs:dsktnum]
0000212A 2A06[2501]                       sub     al, [dsktnum]  ; is it one of the other floppies?
0000212E 7204                             jb      short remap_boot_flop ;       brif so

                                ; we've got to remap the boot hard drive
                                ; subtract the number of EXTRA floppies from it

00002130 0402                             add     al, 2          ; bootdrv -= (dsktnum-2)
00002132 EB04                             jmp     short remap_change_boot_drv
                                ; ---------------------------------------------------------------------------

                                ; we've got to remap the boot floppy.
                                ; add the number of hard drive partitions to it

                                remap_boot_flop:
                                          ;add    al, [cs:drvmax]        ; bootdrv += (drvmax-dsktnum)
                                          ; 15/12/2023
00002134 0206[7500]                       add     al, [drvmax]
                                remap_change_boot_drv:
                                          ;mov    [cs:drvfat], al ; alter msdos.sys load drive
00002138 A2[FA19]                         mov     [drvfat], al
0000213B FEC0                             inc     al
0000213D 1E                               push    ds
0000213E BFD904                           mov     di, SYSINITSEG ; 46Dh
                                          ;mov    di, 544h       ; PCDOS 7.1 IBMBIO.COM
                                          ;;mov   di, 46Dh       ; SYSINIT segment
00002141 8EDF                             mov     ds, di
```

```
7326 00002143 A2[9802]                          mov     [DEFAULTDRIVE], al
7327                                            ;mov    ds:296h, al     ; [SYSINIT+DEFAULT_DRIVE]
7328                                                                    ; pass it to sysinit as      well
7329 00002146 1F                                pop     ds ; ds = cs
7330                                    rmap_exit:
7331 00002147 C3                                retn
7332
7333                                    ; ============== S U B   R O U T I N E =======================================
7334
7335                                    ; 17/10/2022
7336                                    ; 02/10/2022 - Retro DOS v4.0 (MSDOS 5.0 -actual-)
7337                                    ; 28/12/2018 - Retro DOS v4.0 (MSDOS 6.21 -draft-)
7338                                    ; 02/06/2018 - Retro DOS v3.0 (MSDOS 3.3)
7339                                    ; 19/03/2018 - Retro DOS v2.0 (MSDOS 2.11)
7340                                    ;************************************************
7341                                    ; getboot - get the boot sector for a hard disk
7342                                    ;
7343                                    ; Reads the boot sector from a specified drive into
7344                                    ; a buffer at the top of memory.
7345                                    ;
7346                                    ; dl = int13 drive number to read boot sector for
7347                                    ;************************************************
7348
7349                                    ; 17/10/2022
7350                                    bootbias equ 200h
7351
7352                                    getboot:    ; proc near
7353
7354                                                ; 15/12/2023 - Retro DOS v5.0
7355                                                ;       (Modified PCDOS 7.1) IBMBIO.COM/IO.SYS
7356                                                ; ds = cs
7357
7358                                                ; 08/04/2018
7359                                                ; Retro DOS v2.0 (IBMBIO.COM, IBMDOS 2.1)
7360                                                ; 28/03/2018 - MSDOS 6.0 - MSINIT.ASM, 1991
7361                                                ; 02/10/2022 - Retro DOS v4.0
7362                                                ;            (disassembled IO.SYS code of MSDOS 5.0)
7363
7364                                                ;mov    ax, [cs:init_bootseg] ; 17/10/2022
7365                                                ; 15/12/2023
7366 00002148 A1[041A]                          mov     ax, [init_bootseg]
7367 0000214B 8EC0                              mov     es, ax
7368
7369                                                ; 17/10/2022
7370 0000214D BB0002                            mov     bx, bootbias ; 200h
7371                                                ;mov    bx, 200h       ; bootbias
7372                                                                    ; load BX, ES:BX is where sector goes
7373 00002150 B80102                            mov     ax, 201h
7374 00002153 30F6                              xor     dh, dh
7375 00002155 B90100                            mov     cx, 1
7376 00002158 CD13                              int     13h             ; DISK - READ SECTORS INTO MEMORY
7377                                                                    ; AL = number of sectors to read, CH = track, CL = sector
7378                                                                    ; DH = head, DL       = drive, ES:BX -> buffer to fill
7379                                                                    ; Return: CF set on error, AH =      status, AL = number of sectors read
7380 0000215A 7209                              jc      short erret
7381                                                ; 17/10/2022
7382 0000215C 26813EFE0355AA                    cmp     word [es:bootbias+1FEh], 0AA55h
7383                                                ;cmp    word ptr es:3FEh, 0AA55h ; [es:bootbias+1FEh]
7384                                                                    ; Dave Litton magic word?
7385 00002163 7401                              jz      short norm_ret ; yes
7386                                    erret:
7387 00002165 F9                                stc
7388                                    norm_ret:
7389 00002166 C3                                retn
7390
7391                                    ; ============== S U B   R O U T I N E =======================================
7392
7393                                    ; 28/12/2018 - Retro DOS v4.0
7394
7395                                    ;--------------------------------------------------------------------------
7396                                    ;   sethard - generate bpb for a variable sized hard file. ibm has a
7397                                    ;   partitioned hard file; we must read physical sector 0 to determine where
7398                                    ;   our own logical sectors start. we also read in our boot sector to
7399                                    ;   determine version number
7400                                    ;
7401                                    ;   inputs: dl is rom drive number (80...)
7402                                    ;           bh is partition number (0....)
7403                                    ;           ds:di points to bds
7404                                    ;   outputs:        carry clear -> bpb is filled in
7405                                    ;           carry set   -> bpb is left uninitialized due to error
7406                                    ;   trashes (at least) si, cx
7407                                    ;   MUST PRESERVE ES:!!!!
7408                                    ;--------------------------------------------------------------------------
7409
7410                                                ; 16/12/2023 - Retro DOS v5.0 (PCDOS 7.1) IBMBIO.COM
7411                                                ; (PCDOS 7.1 IBMBIO.COM - BIOSDATA:24E9h)
7412
7413                                    sethard:    ; proc near
7414                                                ; 12/08/2023
7415                                                ; ds = cs = BIOSDATA
7416 00002167 57                                push    di
7417 00002168 53                                push    bx
7418                                                ;push   ds  ; ds = cs = BIOSDATA ; 12/08/2023
7419 00002169 06                                push    es
7420 0000216A 885D05                            mov     [di+5], bl      ; [di+BDS.drivelet]
7421 0000216D 885504                            mov     [di+4], dl      ; [di+BDS.drivenum]
7422                                                ; 16/12/2023
7423 00002170 804D3F01                          or      byte [di+3Fh], 1 ; PCDOS 7.1
7424                                                ;or     byte [di+23h], 1 ; [di+BDS.flags]
7425                                                                    ; fnon_removable
7426 00002174 C6453E05                          mov     byte [di+3Eh], 5 ; PCDOS 7.1
7427                                                ;mov    byte [di+22h], 5 ; [di+BDS.formfactor]
7428                                                                    ; ffHardFile
7429 00002178 C606[061A]00                      mov     byte [fbigfat], 0 ; assume 12 bit FAT
7430 0000217D 88FE                              mov     dh, bh          ; partition number
7431 0000217F 52                                push    dx
7432 00002180 B408                              mov     ah, 8
7433 00002182 CD13                              int     13h             ; DISK - DISK -       GET CURRENT DRIVE PARAMETERS (XT,AT,XT286,CONV,PS)
7434                                                                    ; DL = drive number
7435                                                                    ; Return: CF set on error, AH =      status code, BL       = drive type
7436                                                                    ; DL = number of consecutive drives
7437                                                                    ; DH = maximum value for head number, ES:DI -> drive parameter
7438                                                ;inc    dh
7439                                                ; 16/12/2023 - Retro DOS v5.0
7440 00002184 88F2                              mov     dl, dh
7441 00002186 B600                              mov     dh, 0
7442 00002188 42                                inc     dx
7443                                                ;mov    [di+15h], dh   ; [di+BDS.heads] ; get number of heads
7444 00002189 895515                            mov     [di+15h], dx
7445 0000218C 5A                                pop     dx
7446 0000218D 7253                              jc      short setret   ; error if no hard disk
7447                                                ; 16/12/2023
7448                                                ;jc     short setret_j
7449
```

```
0000218F 80E13F                         and     cl, 3Fh
00002192 884D13                         mov     [di+13h], cl   ; [di+BDS.secpertrack]
00002195 52                             push    dx             ; save partition number
00002196 E8AFFF                         call    getboot
00002199 5A                             pop     dx             ; restore partition number
0000219A 7246                           jc      short setret
                                        ; 16/12/2023
                                        ;jnc    short chk_act_part
                                ;setret_j:
                                        ;jmp    setret

                                        ; 16/12/2023 - Retro DOS v5.0 (PCDOS 7.1 IBMBIO.COM)
                                chk_act_part:
0000219C 31DB                           xor     bx, bx ; 0
                                        ;;mov   [cs:ep_start_sector], bx
                                        ;;mov   [cs:ep_start_sector+2], bx
                                        ;mov    [cs:ep_hidden_secs], bx
                                        ;mov    [cs:ep_hidden_secs+2], bx
                                        ; 16/12/2023
                                        ; ds = cs
                                        ; 20/12/2023
                                        ;mov    [ep_start_sector], bx
                                        ;mov    [ep_start_sector+2], bx
0000219E 891E[0422]                     mov     [ep_hidden_secs], bx
000021A2 891E[0622]                     mov     [ep_hidden_secs+2], bx

000021A6 BBC203                         mov     bx, 3C2h       ; 1C2h+bootbias

                                ; The first 'active' partition is 00, the second is 01....
                                ;   then the remainder of the 'primary' but non-active partitions

                                act_part:
000021A9 26F647FC80                     test    byte [es:bx-4], 80h ; is the partition active?
000021AE 740B                           jz      short no_act   ; no
                                ; 16/12/2023
                                %if 0
                                        ; 16/12/2023
                                        ; reject if partitiontype != 1, 4, 6, 0Bh, 0Ch, 0Eh
                                        cmp     byte [es:bx], 1 ; FAT12
                                        jz      short got_good_act
                                        cmp     byte [es:bx], 4       ; FAT16 CHS (<= 32MB)
                                        jz      short got_good_act

                                        ; 16/12/2023 - Retro DOS v5.0 (PCDOS 7.1 IBMBIO.COM)
                                        cmp     byte [es:bx], 0Bh ; FAT32 CHS
                                        jz      short got_good_act
                                        cmp     byte [es:bx], 0Ch ; FAT32 LBA
                                        jz      short got_good_act
                                        cmp     byte [es:bx], 0Eh ; FAT16 LBA
                                        jz      short got_good_act

                                        cmp     byte [es:bx], 6       ; FAT16 BIG CHS (> 32MB)
                                        jnz     short no_act
                                ;%else
                                        ; 16/12/2023
                                        mov     al, [es:bx]     ; partition type

                                        ; reject if partitiontype != 1, 4, 6, 0Bh, 0Ch, 0Eh
                                        cmp     al, 1          ; FAT12
                                        je      short got_good_act
                                        cmp     al, 4          ; FAT16 CHS (<= 32MB)
                                        je      short got_good_act

                                        ; 16/12/2023 - Retro DOS v5.0 (PCDOS 7.1 IBMBIO.COM)
                                        cmp     al, 0Bh        ; FAT32 CHS
                                        je      short got_good_act
                                        cmp     al, 0Ch        ; FAT32 LBA
                                        je      short got_good_act
                                        cmp     al, 0Eh        ; FAT16 LBA
                                        je      short got_good_act

                                        cmp     al, 6          ; FAT16 BIG CHS (> 32MB)
                                        jne     short no_act
                                %endif
                                        ; 16/12/2023 - Retro DOS v5.0 (Modified PCDOS 7.1)
                                        ; check if it is a primary dos partition

000021B0 E83300                         call    chk_partition_type
000021B3 7506                           jne     short no_act

                                got_good_act:                           ; 11/08/2023
000021B5 08F6                           or      dh, dh         ; is this our target partition #?
                                                               ; (0 = first primary dos or active partition)
000021B7 744F                           jz      short set2     ; WE GOT THE ONE WANTED!!
000021B9 FECE                           dec     dh             ; count down
                                no_act:
000021BB 83C310                         add     bx, 16
000021BE 81FB0204                       cmp     bx, 402h       ; 202h+bootbias
                                                               ; last entry done?
000021C2 75E5                           jnz     short act_part ; no, process next entry

000021C4 BBC203                         mov     bx, 3C2h       ; 1C2h+bootbias
                                                               ; restore original value of bx

                                ; Now scan the non-active partitions

                                get_primary:
000021C7 26F647FC80                     test    byte [es:bx-4], 80h
000021CC 750B                           jnz     short not_prim ; we've already scanned
                                                               ; the ACTIVE ones
                                ; 16/12/2023
                                %if 0
                                        ; 16/12/2023
                                        ; see if partitiontype == 1, 4, 6, 0Bh, 0Ch, 0Eh
                                        cmp     byte [es:bx], 1       ; FAT12
                                        jz      short got_prim
                                        cmp     byte [es:bx], 4       ; FAT16 CHS (<= 32MB)
                                        jz      short got_prim

                                        ; 16/12/2023 - Retro DOS v5.0 (PCDOS 7.1 IBMBIO.COM)
                                        cmp     byte [es:bx], 0Bh ; FAT32 CHS
                                        jz      short got_prim
                                        cmp     byte [es:bx], 0Ch ; FAT32 LBA
                                        jz      short got_prim
                                        cmp     byte [es:bx], 0Eh ; FAT16 LBA
                                        jz      short got_prim

                                        cmp     byte [es:bx], 6       ; FAT16 BIG CHS (> 32MB)
                                        jnz     short not_prim
                                ;%else
                                        ; 16/12/2023
                                        mov     al, [es:bx]     ; partition type

                                        ; see if partitiontype == 1, 4, 6, 0Bh, 0Ch, 0Eh
```

```
                                        cmp     al, 1           ; FAT12
                                        je      short got_prim
                                        cmp     al, 4           ; FAT16 CHS (<= 32MB)
                                        je      short got_prim

                                        ; 16/12/2023 - Retro DOS v5.0 (PCDOS 7.1 IBMBIO.COM)
                                        cmp     al, 0Bh         ; FAT32 CHS
                                        je      short got_prim
                                        cmp     al, 0Ch         ; FAT32 LBA
                                        je      short got_prim
                                        cmp     al, 0Eh         ; FAT16 LBA
                                        je      short got_prim

                                        cmp     al, 6           ; FAT16 BIG CHS (> 32MB)
                                        jne     short not_prim
                                %endif
                                        ; 16/12/2023 - Retro DOS v5.0 (Modified PCDOS 7.1)
                                        ; check if it is a primary dos partition

000021CE E81500                         call    chk_partition_type
000021D1 7506                           jne     short not_prim

                                got_prim:
000021D3 08F6                           or      dh, dh          ; is this our target partition?
000021D5 7431                           jz      short set2
000021D7 FECE                           dec     dh
                                not_prim:
000021D9 83C310                         add     bx, 16
000021DC 81FB0204                       cmp     bx, 402h        ; 202h+bootbias
000021E0 75E5                           jnz     short get_primary ; loop till we've gone through table
                                setret:
000021E2 F9                             stc                     ; error return
000021E3 E9C703                         jmp     ret_hard_err

                                ; ---------------------------------------------------------------------------

                                        ; 16/12/2023 - Retro DOS v5.0 (Modified PCDOS 7.1)
                                chk_partition_type:
                                        ; 16/12/2023
000021E6 268A07                         mov     al, [es:bx]     ; partition type

                                        ; see if partitiontype == 1, 4, 6, 0Bh, 0Ch, 0Eh
000021E9 3C01                           cmp     al, 1           ; FAT12
000021EB 7412                           je      short chk_ptype_retn
000021ED 3C04                           cmp     al, 4           ; FAT16 CHS (<= 32MB)
000021EF 740E                           je      short chk_ptype_retn

                                        ; 16/12/2023 - Retro DOS v5.0 (PCDOS 7.1 IBMBIO.COM)
000021F1 3C0B                           cmp     al, 0Bh         ; FAT32 CHS
000021F3 740A                           je      short chk_ptype_retn
000021F5 3C0C                           cmp     al, 0Ch         ; FAT32 LBA
000021F7 7406                           je      short chk_ptype_retn
000021F9 3C0E                           cmp     al, 0Eh         ; FAT16 LBA
000021FB 7402                           je      short chk_ptype_retn

000021FD 3C06                           cmp     al, 6           ; FAT16 BIG CHS (> 32MB)
                                chk_ptype_retn:
                                        ; zf = 1 -> primary DOS partition
                                        ; zf = 0 -> not a primary DOS partition
000021FF C3                             retn

                                ; ---------------------------------------------------------------------------

                                        ; 16/12/2023 - Retro DOS v5.0 (Modified PCDOS 7.1)
                                        ; (PCDOS 7.1 IBMBIO.COM - BIOSDATA:25B6h)
                                ep_start_sector:
00002200 00000000                       dd 0
00002204 00000000               ep_hidden_secs:    dd 0

                                ; ---------------------------------------------------------------------------

                                ;  until we get the real logical boot record and get the bpb,
                                ;  BDS_BPB.BPB_BIGTOTALSECTORS will be used instead of BDS_BPB.BPB_TOTALSECTORS
                                ;  for the convenience of the computation.
                                ;
                                ;  at the end of this procedure, if a bpb information is gotten from
                                ;  the valid boot record, then we are going to use those bpb information
                                ;  without change.
                                ;
                                ;  otherwise, if (hidden sectors + total sectors) <= a word, then we will move
                                ;  BDS_BPB.BPB_BIGTOTALSECTORS (low) to BDS_BPB.BPB_TOTALSECTORS and zero out
                                ;  BDS_BPB.BPB_BIGTOTALSECTORS entry to make it a conventional bpb format.

                                set2:
                                        ; 12/08/2023
                                        ; ds = cs = BIOSDATA segment (0070h)
00002208 8816[071A]                     mov     [rom_drv_num], dl
                                        ;mov    [cs:rom_drv_num], dl
                                        ; save the rom bios drive number we are handling now.
0000220C 268B4704                       mov     ax, [es:bx+4]  ; hidden sectors (start      sector)
00002210 268B5706                       mov     dx, [es:bx+6]

                                ; decrement the sector count by 1 to make it zero based. exactly 64k
                                ; sectors should be allowed

00002214 83E801                         sub     ax, 1
00002217 83DA00                         sbb     dx, 0
0000221A 26034708                       add     ax, [es:bx+8]  ; sectors in partition
0000221E 2613570A                       adc     dx, [es:bx+10]
00002222 7305                           jnc     short okdrive
00002224 800E[061A]80                   or      byte [fbigfat], 80h ; ftoobig

                                        ; 16/12/2023 - Retro DOS v5.0 (PCDOS 7.1) IBMBIO.COM
                                        ;;;
                                okdrive:
                                        ;add    ax, [cs:ep_hidden_secs]
                                        ;adc    dx, [cs:ep_hidden_secs+2]
                                        ; ds = cs
00002229 0306[0422]                     add     ax, [ep_hidden_secs]
0000222D 1316[0622]                     adc     dx, [ep_hidden_secs+2]
00002231 7305                           jnc     short okdrive_1
00002233 800E[061A]80                   or      byte [fbigfat], 80h ; ftoobig
                                okdrive_1:
00002238 26803F0C                       cmp     byte [es:bx], 0Ch ; FAT32 LBA partition ID
0000223C 7418                           je      short set_lba_flag
0000223E 26803F0E                       cmp     byte [es:bx], 0Eh ; FAT16 LBA partition ID
00002242 7412                           je      short set_lba_flag
00002244 3B5513                         cmp     dx, [di+13h]   ; if dx > [di+BDS.secpertrack] then
00002247 730D                           jnb     short set_lba_flag ; set LBA r/w flag
00002249 F77513                         div     word [di+13h]
0000224C 31D2                           xor     dx, dx
0000224E F77515                         div     word [di+15h]
00002251 3D0004                         cmp     ax, 400h        ; if ax (cylinder number) >= 1024
```

```
7698                                                    ;  set LBA r/w flag
7699 00002254 7204                      jb      short set3
7700                                set_lba_flag:
7701 00002256 804D4004                          or byte [di+40h], 4 ; fLBArw ; LBA r/w flag
7702                                    ;;;
7703                                ;okdrive:
7704                                    ; 16/12/2023
7705                                set3:
7706                                    ;mov    ax, [es:bx+4]
7707                                    ;mov    [di+17h], ax   ; [di+BDS.hiddensecs]
7708                                    ;                      ; BPB_HIDDENSECTORS = p->partitionbegin
7709                                    ;mov    ax, [es:bx+6]
7710                                    ;mov    [di+19h], ax   ; [di+BDS.hiddensecs+2]
7711
7712                                    ; 16/12/2023 - Retro DOS v5.0 (PCDOS 7.1) IBMBIO.COM
7713                                    ;;;
7714 0000225A 268B4704                  mov     ax, [es:bx+4]  ; start sector (LBA) of the partition
7715 0000225E 268B5706                  mov     dx, [es:bx+6]
7716                                    ;add    ax, [cs:ep_hidden_secs]
7717                                    ;adc    dx, [cs:ep_hidden_secs+2]
7718                                    ; ds = cs
7719 00002262 0306[0422]                add     ax, [ep_hidden_secs]
7720                                                    ; + hidden secs of the extd dos partion
7721 00002266 1316[0622]                adc     dx, [ep_hidden_secs+2]
7722 0000226A 894517                    mov     [di+17h], ax   ; [di+BDS.hiddensecs]
7723 0000226D 895519                    mov     [di+19h], dx   ; [di+BDS.hiddensecs+2]
7724 00002270 31C0                      xor     ax, ax ; 0
7725 00002272 89457B                    mov     [di+7Bh], ax   ; [di+BDS.bdsm_hidden_trks]
7726 00002275 89450E                    mov     [di+0Eh], ax   ; [di+BDS.totalsec16]
7727                                    ;;;
7728
7729 00002278 268B570A                  mov     dx, [es:bx+10] ; # of sectors (high)
7730 0000227C 268B4708                  mov     ax, [es:bx+8]  ; # of sectors (low)
7731 00002280 89551D                    mov     [di+1Dh], dx   ; [di+BDS.totalsecs32+2]
7732 00002283 89451B                    mov     [di+1Bh], ax   ; [di+BDS.totalsecs32]
7733                                                    ; bpb->maxsec =       p->partitionlength
7734                                    ;cmp    dx, 0
7735                                    ;ja     short okdrive_1
7736                                    ; 16/12/2023
7737 00002286 09D2                      or      dx, dx
7738 00002288 7505                      jnz     short set3_read
7739 0000228A 83F840                    cmp     ax, 64         ; if (p->partitionlength < 64)
7740                                    ;jb     short setret   ; return -1;
7741 0000228D 7264                      jb      short set3_err
7742                                ;okdrive_1:
7743                                    ; 16/12/2023
7744                                set3_read:
7745 0000228F 8B5519                    mov     dx, [di+19h]   ; [di+BDS.hiddensecs+2]
7746 00002292 8B4517                    mov     ax, [di+17h]   ; [di+BDS.hiddensecs]
7747 00002295 31DB                      xor     bx, bx         ; boot sector number - for mini      disk
7748                                                    ; usually equal       to the # of sec/trk.
7749 00002297 8A5D13                    mov     bl, [di+13h]   ; [di+BDS.secpertrack]
7750 0000229A 50                        push    ax
7751 0000229B 89D0                      mov     ax, dx
7752 0000229D 31D2                      xor     dx, dx
7753 0000229F F7F3                      div     bx             ; (sectors)dx:ax / (BDS.secpertrack)bx =
7754                                                    ; (track)temp_h:ax + (sector)dx
7755                                ; 16/12/2023
7756                                %if 0
7757                                    ; 17/10/2022
7758                                    ;mov    [cs:temp_h], ax
7759                                    ; 12/08/2023 (ds=cs)
7760                                    mov     [temp_h], ax
7761                                    pop     ax
7762                                    div     bx
7763                                    mov     cl, dl
7764                                    inc     cl
7765                                    xor     bx, bx
7766                                    mov     bl, [di+15h]   ; [di+BDS.heads]
7767                                    push    ax
7768                                    xor     dx, dx
7769                                    ;mov    ax, [cs:temp_h]
7770                                    mov     ax, [temp_h] ; 12/08/2023
7771                                    div     bx
7772                                    ;mov    [cs:temp_h], ax
7773                                    mov     [temp_h], ax ; 12/08/2023
7774                                    pop     ax
7775                                    div     bx             ; dl is head, ax is cylinder
7776                                    ; 12/08/2023 (ds=cs)
7777                                    cmp     word [temp_h], 0
7778                                    ;cmp    word [cs:temp_h], 0
7779                                    ja      short setret_brdg ; exceeds the      limit of int 13h
7780                                    cmp     ax, 1024
7781                                    ja      short setret_brdg ; exceeds the      limit of int 13h
7782                                            ; Retro DOS v3.2 note by Erdogan Tan - 28/07/2019
7783                                            ; **MSDOS code accepts if ax = 1024 but it is nonsense here
7784                                            ; ('ja' must be 'jnb')
7785                                okdrive_2:
7786                                            ; 28/07/2019
7787                                ; dl is head.
7788                                ; ax is cylinder
7789                                ; cl is sector number (assume less than 2**6 = 64 for int 13h)
7790
7791                                ;*** for mini disks ***
7792
7793                                    cmp     word [di+47h], 1 ; [di+BDS.bdsm_ismini]
7794                                                             ; check for mini disk
7795                                    jnz     short oknotmini       ; not mini disk.
7796                                    add     ax, [di+49h]   ; [di+BDS.bdsm_hidden_trks]
7797                                                             ; set the physical track number
7798                                oknotmini:
7799                                %endif
7800                                    ; 16/12/2023 - Retro DOS v5.0 (PCDOS 7.1) IBMBIO.COM
7801                                    ;;;
7802                                    ;mov    [cs:saved_word], ax
7803 000022A1 A3[9E04]                  mov     [saved_word], ax
7804 000022A4 58                        pop     ax
7805 000022A5 F7F3                      div     bx
7806 000022A7 88D1                      mov     cl, dl
7807 000022A9 FEC1                      inc     cl
7808 000022AB 8B5D15                    mov     bx, [di+15h]   ; [di+BDS.heads]
7809 000022AE 50                        push    ax
7810 000022AF 31D2                      xor     dx, dx
7811                                    ;mov    ax, [cs:saved_word]
7812 000022B1 A1[9E04]                  mov     ax, [saved_word]
7813 000022B4 F7F3                      div     bx
7814                                    ;mov    [cs:saved_word], ax
7815 000022B6 A3[9E04]                  mov     [saved_word], ax ; not necessary !? (ax must be 0)
7816 000022B9 58                        pop     ax
7817 000022BA F7F3                      div     bx             ; dl is head, ax is cylinder
7818                                    ; 16/12/2023
7819 000022BC 0E                        push    cs
7820 000022BD 07                        pop     es ; (*)
7821 000022BE BB[5201]                  mov     bx, disksector ; (**)
```

```
                                        ;
000022C1 F6454004                       test    byte [di+40h], 4 ; fLBArw ; LBA read/write flag
000022C5 742F                           jz      short set3_chs_read
                                set3_lba_read:

                                ; 16/12/2023
                                %if 0
                                        ;push   cs
                                        ;pop    es ; (*)
                                        ;mov    bx, disksector ; (**)

                                        ;push   ds
                                        ;push   si
                                        xor     ax, ax  ; 0
                                        push    ax
                                        push    ax
                                        mov     ax, [di+19h]   ; [di+BDS.hiddensectors+2]
                                        push    ax
                                        mov     ax, [di+17h]   ; [di+BDS.hiddensectors]
                                        push    ax
                                        push    es              ; buffer address
                                        push    bx
                                        mov     ax, 1           ; sector (read) count
                                        push    ax
                                        ;mov    ax, 16          ; DAP size
                                        mov     al, 16
                                        push    ax
                                        mov     dl, [rom_drv_num] ; ds = cs
                                        mov     ax, ss
                                        mov     ds, ax ; ds = ss
                                        mov     si, sp
                                        ;mov    dl, [cs:rom_drv_num]
                                        mov     ah, 42h
                                        int     13h             ; DISK - IBM/MS Extension
                                                                ; EXTENDED READ (DL - drive, DS:SI - disk address packet)
                                        ;pop    si
                                        ;pop    ds
                                        jnc     short set3_lba_read_ok
                                        add     sp, 16
                                        ;pop    si
                                        ;pop    ds
                                set3_err:
                                        ;jmp    setret
                                        jmp     ret_hard_err

                                set3_lba_read_ok
                                        add     sp, 16
                                        ;pop    si
                                        ;pop    ds
                                        jmp     short set3_read_ok
                                %else
                                        ; 16/12/2023
                                        ;push   si ; * ; (not necessary)
                                        ;mov    si, empty_dap_buff ; dap_buffer
000022C7 BE[571B]                       mov     si, dap_buffer ; empty_dap_buff
000022CA 56                             push    si
000022CB 87F7                           xchg    si, di
                                        ; si = BDS
                                        ; di = DAP buffer
000022CD B81000                         mov     ax, 16
000022D0 AB                             stosw           ; DAP size
000022D1 B001                           mov     al, 1
000022D3 AB                             stosw           ; sector (read) count
                                        ; buffer address
000022D4 89D8                           mov     ax, bx  ; offset disksector
000022D6 AB                             stosw
000022D7 8CC0                           mov     ax, es  ; es=ds=cs = BIOSDATA segment
000022D9 AB                             stosw
                                        ; sector address (bits 0 to 31)
000022DA 8B4417                         mov     ax, [si+17h] ; [di+BDS.hiddensectors]
000022DD AB                             stosw
000022DE 8B4419                         mov     ax, [si+19h] ; [di+BDS.hiddensectors+2]
000022E1 AB                             stosw
                                        ; sector address bits 32 to 63 (0)
000022E2 31C0                           xor     ax, ax ; 0
000022E4 AB                             stosw
000022E5 AB                             stosw
                                        ;xchg   di, si
000022E6 89F7                           mov     di, si
                                        ; di = BDS
000022E8 5E                             pop     si ; DAP buffer address

000022E9 8A16[071A]                     mov     dl, [rom_drv_num] ; ds = cs
000022ED B442                           mov     ah, 42h
000022EF CD13                           int     13h             ; DISK - IBM/MS Extension
                                                                ; EXTENDED READ (DL - drive, DS:SI - disk address packet)
                                        ;pop    si ; *
000022F1 7324                           jnc     short set3_read_ok
                                set3_err:
                                        ;jmp    setret
000022F3 E9B702                         jmp     ret_hard_err
                                %endif

                                set3_chs_read:
000022F6 837D7901                       cmp     word [di+79h], 1 ; [di+BDS.bdsm_ismini] ; check for mini disk
000022FA 7503                           jnz     short oknotmini
000022FC 03457B                         add     ax, [di+7Bh]   ; [di+BDS.bdsm_hidden_trks]
                                        ;;;

                                oknotmini:
                                ;*** end of added logic for mini disk

000022FF D0CC                           ror     ah, 1           ; move high two bits of cyl to high
00002301 D0CC                           ror     ah, 1           ; two bits of upper byte
00002303 80E4C0                         and     ah, 0C0h        ; turn off remainder of bits
00002306 08E1                           or      cl, ah          ; move two bits to correct spot
00002308 88C5                           mov     ch, al          ; ch iscylinder (low 8 bits)
                                                                ; cl is sector + 2 high bits of cylinder
0000230A 88D6                           mov     dh, dl          ; dh is head

                                        ; 12/08/2023 (ds=cs)
0000230C 8A16[071A]                     mov     dl, [rom_drv_num]
                                        ;mov    dl, [cs:rom_drv_num] ; dl is drive number

                                ; cl is sector + 2 high bits of cylinder
                                ; ch is low 8 bits of cylinder
                                ; dh is head
                                ; dl is drive

                                ; for convenience, we are going to read the logical boot sector
                                ; into cs:disksector area.

                                ; read in boot sector using bios disk interrupt. the buffer where it
                                ; is to be read in is cs:disksector.
```

```
                                        ; 16/12/2023
                                        ; es=ds=cs = BIOSDATA segment
                                        ; bx = disksector ; (**)

                                        ;push   cs
                                        ;pop    es ; (*)

                                        ;mov    bx, disksector ; for convenience,
                                                               ; we are going to read the logical boot sector
                                                               ; into cs:disksector area.
00002310 B80102                         mov     ax, 201h
00002313 CD13                           int     13h             ; DISK - READ SECTORS INTO MEMORY
                                                               ; AL = number of sectors to read, CH = track, CL = sector
                                                               ; DH = head, DL       = drive, ES:BX -> buffer to fill
                                                               ; Return: CF set on error, AH =      status, AL = number of sectors read
                                        ; 16/12/2023
00002315 72DC                           jc      short set3_err

                                ; cs:disksec contains the boot sector. in theory, (ha ha) the bpb in this thing
                                ; is correct. we can, therefore, suck out all the relevant statistics on the
                                ; media if we recognize the version number.

                                set3_read_ok:
                                        ; 11/08/2023
                                        ;mov    bx, disksector ; BIOSDATA:014Eh ; MSDOS 6.21 ; 11/08/2023
                                                               ; BIOSDATA:0152h ; PCDOS 7.1 IBMBIO.COM
                                        ; 18/12/2023
                                        ;push   bx ; +
                                        ;push   ax ; (not necessary)

                                        ; 16/12/2023 - Retro DOS v5.0 (PCDOS 7.1 IBMBIO.COM)
                                        ;;;
00002317 81BFFE0155AA                   cmp     word [bx+1FEh], 0AA55h
0000231D 7541                           jne     short invalid_boot_record
                                        ; 16/12/2023
                                        ; 12/08/2023
                                        ; ds = cs = BIOSDATA segment ('disksector:' is in BIOSDATA)
0000231F 803FE9                         cmp     byte [bx], 0E9h         ; is it a near jump?
00002322 740B                           je      short check_1_ok ; yes
00002324 803FEB                         cmp     byte [bx], 0EBh         ; is it a short jump?
00002327 7537                           jne     short invalid_boot_record ; no
00002329 807F0290                       cmp     byte [bx+2], 90h ; yes, is the next one a nop?
0000232D 7531                           jne     short invalid_boot_record ; no, invalid bs ; 11/08/2023
                                check_1_ok:
0000232F 837F1600                       cmp      word [bx+16h], 0 ; [bx+BPB_FATSz16]
                                        ;jz     short check_1  ; 16 bit FAT size is 0 if it is FAT32 bs
                                                               ; 16/12/2023
00002333 740E                           jz      short check_2  ; FAT32 bs

                                        ; FAT16 or FAT12 bs

                                        ;push   ds
                                        ;push   si  ; (not necessary)
00002335 57                             push    di
                                        ; es=ds=cs = BIOSDATA segment
                                        ;push   es
                                        ;pop    ds

                                        ;mov    cx, 28
00002336 B90E00                         mov     cx, 14 ; *
00002339 8D7724                         lea     si, [bx+24h]   ; move offset 36 to 63
                                                               ;      to offset 64 (28 bytes)
0000233C 8D7F40                         lea     di, [bx+40h]   ; boot sector offset 64
0000233F FC                             cld     ; (not necessary, 'std' is not used before here)
                                        ;rep movsb
00002340 F3A5                           rep movsw ; *
00002342 5F                             pop     di
                                        ;pop    si
                                        ;pop    ds
                                        ;;;
                                ; 16/12/2023
                                %if 0
                                ;check_1:
                                        ; 12/08/2023
                                        ; ds = cs = BIOSDATA segment ('disksector:' is in BIOSDATA)
                                        cmp     byte [bx], 0E9h
                                        ;cmp    byte [cs:bx], 0E9h ; is it a near jump?
                                        je      short check_1_ok ; yes
                                        cmp     byte [bx], 0EBh
                                        ;cmp    byte [cs:bx], 0EBh ; is it a short jump?
                                        jne     short invalid_boot_record ; no
                                        cmp     byte [bx+2], 90h
                                        ;cmp    byte [cs:bx+2], 90h ; yes, is the next one a nop?
                                        jne     short invalid_boot_record ; no, invalid bs ; 11/08/2023
                                check_1_ok:
                                %endif

                                ; 18/12/2023
                                %if 0
                                        ; 14/08/2023
                                check_2:
                                        mov     bx, disksector+11 ; disksector+EXT_BOOT.BPB
                                        ;mov    bx, 159h        ; disksector+EXT_BOOT.BPB
                                                                ; point to the bpb in the boot record
                                        ;mov    al, [cs:bx+10] ; [bx+EBPB.MEDIADESCRIPTOR]
                                        mov     al, [bx+10] ; 12/08/2023
                                                                ; get the mediadescriptor byte
                                        and     al, 0F0h        ; mask off low nibble
                                        cmp     al, 0F0h        ; is high nibble = 0Fh?
                                        jne     short invalid_boot_record ; no, invalid boot record
                                        ;cmp    word [cs:bx], 512 ; [bx+EBPB.BYTESPERSECTOR]
                                        cmp     word [bx], 512 ; 12/08/2023
                                        jne     short invalid_boot_record ; invalidate non 512 byte sectors

                                check2_ok:                      ; yes, mediadescriptor ok.
                                        mov     al, [bx+2] ; 12/08/2023
                                        ;mov    al, [cs:bx+2]  ; now make sure that
                                                                ; the sectorspercluster is
                                                                ; a power of 2
                                                                ;
                                                                ; [bx+EBPB.SECTORSPERCLUSTER]
                                                                ; get the sectorspercluster
                                %endif
                                        ;;;
                                check_2:
                                        ; 18/12/2023
                                        ; bx = disksector
00002343 8A4715                         mov     al, [bx+21]     ; [bx+EXT_BOOT.BPB+EBPB.MEDIADESCRIPTOR]
                                                                ; get the mediadescriptor byte
00002346 24F0                           and     al, 0F0h        ; mask off low nibble
00002348 3CF0                           cmp     al, 0F0h        ; is high nibble = 0Fh?
0000234A 7514                           jne     short invalid_boot_record ; no, invalid boot record
0000234C 817F0B0002                     cmp     word [bx+11], 512 ; [bx+EXT_BOOT.BPB+EBPB.BYTESPERSECTOR]
```

```
8070 00002351 750D                          jne     short invalid_boot_record ; invalidate non 512 byte sectors
8071
8072                                check2_ok:  ; yes, mediadescriptor ok.
8073 00002353 8A470D                        mov     al, [bx+13]     ; now make sure that
8074                                                                ; the sectorspercluster is
8075                                                                ; a power of 2
8076                                                                ;
8077                                                                ; [bx++EXT_BOOT.BPB+EBPB.SECTORSPERCLUSTER]
8078                                                                ; get the sectorspercluster
8079                                        ;;;
8080
8081 00002356 08C0                          or      al, al          ; is it zero?
8082 00002358 7406                          jz      short invalid_boot_record ; yes, invalid boot record
8083
8084                                ck_power_of_two:
8085 0000235A D0E8                          shr     al, 1           ; shift until first bit emerges
8086 0000235C 73FC                          jnc     short ck_power_of_two
8087 0000235E 7406                          jz      short valid_boot_record
8088
8089                                invalid_boot_record:
8090                                        ; 18/12/2023
8091                                        ;pop    ax
8092                                        ;pop    bx ; +
8093 00002360 E96001                        jmp     unknown         ; jump to invalid boot record
8094                                                                ; unformatted or illegal media.
8095                                ; 16/12/2023
8096                                ; ---------------------------------------------------------------------------
8097                                ;   ; 12/08/2023
8098                                ;setret_brdg:
8099                                ;           jmp     setret
8100                                ; ---------------------------------------------------------------------------
8101
8102                                unknown3_0_j:
8103 00002363 E96101                        jmp     unknown3_0      ; legally formatted media,
8104                                                                ; although, content might be bad.
8105                                ; ---------------------------------------------------------------------------
8106
8107                                valid_boot_record:
8108                                        ; 18/12/2023
8109                                        ;pop    ax
8110                                        ;pop    bx ; +
8111
8112                                        ; 18/12/2023
8113                                        ; bx = offset disksector ; +
8114
8115                                ; Signature found. Now check version.
8116
8117                                        ; 14/08/2023
8118 00002366 817F08322E                    cmp     word [bx+8], '2.'
8119                                        ;cmp    word [cs:bx+8], '2.' ; 03/10/2022 (NASM syntax)
8120                                        ;;cmp   word ptr cs:[bx+8], 2E32h ; '2.'
8121 0000236B 7506                          jne     short try5
8122 0000236D 807F0A30                      cmp     byte [bx+10], '0'
8123                                        ;cmp    byte [cs:bx+0Ah], '0' ; 03/10/2022 (NASM syntax)
8124                                        ;;cmp   byte ptr cs:[bx+0Ah], 30h ; '0'
8125                                        ; 12/08/2023
8126                                        ;jnz    short try5
8127                                        ;jmp    short copybpb
8128 00002371 7425                          je      short copybpb
8129
8130                                ;; ---------------------------------------------------------------------------
8131                                ;;  ; 12/08/2023
8132                                ;;setret_brdg:
8133                                ;;          jmp     setret
8134                                ;; ---------------------------------------------------------------------------
8135                                ;
8136                                ;unknown3_0_j:
8137                                ;           jmp     unknown3_0      ; legally formatted media,
8138                                ;                                   ; although, content might be bad.
8139                                ; ---------------------------------------------------------------------------
8140
8141                                try5:
8142 00002373 E83B02                        call    cover_fdisk_bug
8143
8144                                ; see if it is an os2 signature
8145
8146                                        ; 12/08/2023
8147                                        ; ds = cs = BIOSDATA segment
8148 00002376 817F08302E                    cmp     word [bx+8], '0.'
8149                                        ;cmp    word [cs:bx+8], '0.' ; 03/10/2022 (NASM syntax)
8150                                        ;;cmp   word ptr cs:[bx+8], 2E30h ; '0.'
8151 0000237B 750C                          jne     short no_os2
8152 0000237D 8A4707                        mov     al, [bx+7] ; 12/08/2023
8153                                        ;mov    al, [cs:bx+7]  ; 17/10/2022 (NASM syntax)
8154 00002380 2C31                          sub     al, '1'
8155                                        ;sub    al, 31h         ; '1'
8156 00002382 24FE                          and     al, 0FEh
8157 00002384 7412                          jz      short copybpb  ; accept either       '1' or '2'
8158 00002386 E93A01                        jmp     unknown
8159                                ; ---------------------------------------------------------------------------
8160
8161                                ; no os2 signature, this is to check for real dos versions
8162
8163                                no_os2:
8164                                        ; 12/08/2023
8165                                        ; ds = cs = BIOSDATA
8166 00002389 817F08332E                    cmp     word [bx+8], '3.'
8167                                        ;cmp    word [cs:bx+8], '3.' ; 03/10/2022 (NASM syntax)
8168                                        ;;cmp   word ptr cs:[bx+8], 2E33h ; '3.'
8169 0000238E 72D3                          jb      short unknown3_0_j ; must be 2.1 boot record.
8170                                                                   ; do not trust it, but still legal.
8171 00002390 7506                          jnz     short copybpb  ; honor os2 boot record
8172                                                                ; or dos 4.0 version
8173 00002392 807F0A31                      cmp     byte [bx+10], '1' ; 12/08/2023
8174                                        ;cmp    byte [cs:bx+10], '1'
8175                                        ;;cmp   byte ptr cs:[bx+0Ah], 31h ; '1'
8176 00002396 72CB                          jb      short unknown3_0_j ; if version >= 3.1, then o.k.
8177                                copybpb:
8178
8179                                ; 03/10/2022
8180
8181                                ; we have a valid boot sector. use the bpb in it to build the
8182                                ; bpb in bios. it is assumed that only
8183                                ;   BDS_BPB.BPB_SECTORSPERCLUSTER
8184                                ;   BDS_BPB.BPB_ROOTENTRIES, and
8185                                ;   BDS_BPB.BPB_SECTORSPERFAT
8186                                ; need to be set (all other values in already). fbigfat is also set.
8187
8188                                ; if it is non fat based system, then just copy the bpb from the boot sector
8189                                ; into the bpb in bds table, and also set the boot serial number, volume id,
8190                                ; and system id according to the boot record.
8191                                ; for the non_fat system, don't need to set the other value. so just do goodret.
8192
8193                                        ; 17/12/2023 - Retro DOS v5.0 (PCDOS 7.1) IBMBIO.COM
```

```
                                        ; (PCDOS 7.1 IBMBIO.COM - BIOSDATA:2787h)
                                        ;;;
                                        ; 17/12/2023
00002398 BE[5D01]                       mov     si, disksector+11
                                        ;sub    ch, ch ; ; (ch may be > 0)
0000239B 29C9                           sub     cx, cx ; 0
                                        ;mov    cl, [disksector+16] ; BPB_NumFATs
0000239D 8A4C05                         mov     cl, [si+5] ; number of FATs

                                        ; NOTE: This check is not proper for FAT32 boot sector (standard spec)
                                        ; (after PCDOS 7.1). So, it is not existing in Windows ME IO.SYS
                                        ; Erdogan Tan - 01/09/2023 ((IBMBIO.COM 7.1 disassembly note))

                                        ;;cmp   word ptr cs:disksector+4Dh, 0 ; ???
                                        ;cmp    word [disksector+4Dh], 0
                                        ;jnz    short check_3

                                        ; 17/12/2023
                                        ; check extended boot signature (0x29)
                                        ;
                                        ; (***) NOTE: 28 bytes of FAT16/FAT12 boot sector from offset 36
                                        ; have been moved to offset 64 (see label 'check_1_ok:' above)
                                        ; ((now, BS_BootSig is at offset 66 even if it was at offset 38))

                                        ;cmp    cs:disksector+42h, 29h; BS_BootSig (FAT32)
000023A0 803E[9401]29                   cmp     byte [disksector+42h], 29h ; BS_BootSig (***)
                                        ;jmp    short check_4
                                check_3:
                                        ;;cmp   cs:disksector+26h, 29h; BS_BootSig (FAT16/FAT12)
                                        ;cmp    byte [disksector+26h], 29h ; (***)
                                check_4:
000023A5 7538                           jnz     short copybpb_fat     ; conventional fat system

                                ; 17/12/2023
                                %if 0
                                        ; 10/12/2022
                                        ; (number of FATs optimization)
                                        mov     si, disksector+11 ; disksector+0Bh
                                        ;;mov   cl, [cs:disksector+10h] ; Number of FATs (may be 2 or 1)
                                        ;mov    cl, [cs:si+05h]
                                        ; 12/08/2023
                                        ; ds = cs = BIOSDATA segment (0070h)
                                        mov     cl, [si+05h] ; number of FATs

                                        cmp     byte [si+1Bh], 29h ; 12/08/2023
                                        ;cmp    byte [cs:si+1Bh], 29h ; 10/12/2022
                                        ;;cmp   byte [cs:disksector+26h], 29h ; 17/10/2022
                                                                ; [disksector+EXT_BOOT.SIG]
                                                                ; EXT_BOOT_SIGNATURE
                                        jnz     short copybpb_fat ; conventional fat system

                                        ; 03/10/2022
                                        ; 29/12/2018 - Retro DOS v4.0 modification note:
                                        ; Regarding 'fat_big_small' part of this (MSDOS 6.0) code
                                        ;           number of FATs must be 2 ; =*?=
                                        ; (Otherwise, '# of data sectors' would be calculated as wrong!!!)
                                        ;
                                        ;cmp    byte [disksector+EXT_BOOT.BPB+EBPB.NUMBEROFFATS], 2 ; =*?=

                                        ; 10/12/2022
                                        ;cmp    byte [cs:disksector+10h], 0
                                                                ; [disksector+EXT_BOOT.BPB+EBPB.NUMBEROFFATS]
                                        ;jnz    short copybpb_fat ; a fat system.
                                        or      cl, cl  ; [cs:disksector+10h]
                                        jnz     short copybpb_fat ; a fat system.
                                %endif

                                        ; 17/12/2023 - Retro DOS v5.0
                                        ;;cmp   byte [cs:disksector+10h], 0 ; BPB.fats
                                        ;cmp    byte [disksector+10h], 0 ; BPB_NUmFATs
                                        ;jnz    short copybpb_fat ; a fat system
                                        ; 17/12/2023
                                        ; cl = [disksector+10h]
000023A7 20C9                           and     cl, cl ; 0 ?
000023A9 7534                           jnz     short copybpb_fat ; a fat system

                                ; non fat based    media.

000023AB 57                             push    di  ; BDS
                                        ; 12/08/2023
                                        ;push   ds  ; ds = cs = BIOSDATA segment

                                        ; 17/12/2023
                                        ; es = ds = cs
                                        ;push   ds
                                        ;pop    es

                                        ; 12/08/2023
                                        ; ds = cs
                                        ;push   cs
                                        ;pop    ds

                                        ; 10/12/2022
                                        ; (number of FATs optimization)
                                        ; SI = disksector+11
                                        ; 17/10/2022
                                        ;;mov   si, 159h       ; disksector+EXT_BOOT.BPB
                                        ;mov    si, disksector+11
000023AC 83C706                         add     di, 6           ; add di,BDS.BPB

                                ; just for completeness, we'll make sure that total_sectors and
                                ; big_total_sectors aren't both zero. I've seen examples of
                                ; this on DOS 3.30 boot records. I don't know exactly how it
                                ; got that way. If it occurs, then use the values from the
                                ; partition table.

                                        ; 17/12/2023
                                        ; cx = 0
                                        ; 18/12/2022
                                        ;sub    cx, cx

                                        ;cmp    word [cs:si+8], 0     ; [cs:si+EBPB.TOTALSECTORS]
                                        ;jnz    short already_nonz
                                        ;                       ; how about big_total?
                                        ;cmp    word [cs:si+15h], 0   ; [cs:si+EBPB.BIGTOTALSECTORS]
                                        ;jnz    short already_nonz ; we're okay if any are != 0
                                        ;cmp    word [cs:si+17h], 0   ; [cs:si+EBPB.BIGTOTALSECTORS+2]
                                        ;jnz    short already_nonz

                                        ; 12/08/2023
                                        ; ds = cs = BIOSDATA segment (0070h)

                                        ; 17/12/2023
                                        ; 12/08/2023
```

```
8318 000023AF 394C08                    cmp     [si+8], cx ; 0        ; [si+EBPB.TOTALSECTORS]
8319 000023B2 751C                      jnz     short already_nonz
8320                                                    ; how about big_total?
8321 000023B4 394C15                    cmp     [si+15h], cx ; 0      ; [si+EBPB.BIGTOTALSECTORS]
8322 000023B7 7517                      jnz     short already_nonz ; we're okay if any are != 0
8323 000023B9 394C17                    cmp     [si+17h], cx ; 0      ; [si+EBPB.BIGTOTALSECTORS+2]
8324 000023BC 7512                      jnz     short already_nonz
8325
8326                                ; now let's copy the values from the partition table (now in the BDS)
8327                                ; into the BPB in the boot sector buffer, before they get copied back.
8328
8329 000023BE 8B4508                        mov     ax, [di+8]     ; [di+BDS.totalsecs16]
8330                                        ; 12/08/2023
8331                                        ;mov    [cs:si+8], ax  ; [cs:si+EBPB.TOTALSECTORS]
8332 000023C1 894408                        mov     [si+8], ax
8333 000023C4 8B4515                        mov     ax, [di+15h]   ; [di+BDS.totalsecs32]
8334                                        ;mov    [cs:si+15h], ax       ; [cs:si+EBPB.BIGTOTALSECTORS]
8335 000023C7 894415                        mov     [si+15h], ax
8336 000023CA 8B4517                        mov     ax, [di+17h]   ; [di+BDS.totalsecs32+2]
8337                                        ;mov    [cs:si+17h], ax       ; [cs:si+EBPB.BIGTOTALSECTORS+2]
8338 000023CD 894417                        mov     [si+17h], ax
8339
8340                                 already_nonz:
8341                                        ; 18/12/2022
8342                                        ; cx = 0
8343                                        ;mov    cl, 25
8344                                        ;;mov   cx, 25         ; A_BPB.size - 6 ; Use SMALL version!
8345                                        ; 17/12/2023 - Retro DOS v5.0
8346 000023D0 B135                        mov     cl, 53  ; PCDOS 7.1 IBMBIO.COM
8347                                                        ; BDS.BPB size (25 + 28 for FAT32 parms)
8348 000023D2 F3A4                        rep movsb
8349                                        ;pop    ds
8350                                        ; 12/08/2023
8351                                        ; ds = cs
8352                                        ;pop    bp  ; ds (on top of stack) = BIOSDATA
8353 000023D4 5F                          pop     di  ; BDS
8354                                        ;push   es
8355                                        ;push   ds
8356                                        ;pop    es
8357                                        ;push   cs
8358                                        ;pop    ds
8359                                        ; 12/08/2023
8360                                        ;mov    es, bp
8361                                        ; ds = cs = es
8362
8363                                        ; 14/08/2023
8364 000023D5 BD[4F08]                    mov     bp, MOVMEDIAIDS ; mov_media_ids
8365                                        ; 18/12/2022
8366                                        ;mov    bp, mov_media_ids
8367                                        ;;mov   bp, 751h       ; mov_media_ids
8368                                                               ; at 2C7h:751h = 70h:2CC1h
8369                                                               ; set volume id, systemid, serial.
8370 000023D8 0E                          push    cs             ; simulate far call
8371 000023D9 E895F6                      call    call_bios_code
8372                                        ; 12/08/2023
8373                                        ; ds = cs = es
8374                                        ;push   es
8375                                        ;pop    ds
8376                                        ;pop    es
8377 000023DC E9C701                      jmp     goodret
8378
8379                                ; ---------------------------------------------------------------------------
8380
8381                                ; ****** cas ---
8382                                ; IBM DOS 3.30 doesn't seem to mind that the TOTAL_SECTORS and
8383                                ; BIG_TOTAL_SECTORS field in the boot sector are 0000. This
8384                                ; happens with some frequency -- perhaps through some OS/2 setup
8385                                ; program. We haven't actually been COPYING the TOTAL_SECTORS
8386                                ; from the boot sector into the DPB anyway, we've just been using
8387                                ; it for calculating the fat size. Pretty scary, huh? For now,
8388                                ; we'll go ahead and copy it into the DPB, except in the case
8389                                ; that it equals zero, in which case we just use the values in
8390                                ; the DPB from the partition table.
8391
8392                                ; 17/10/2022
8393                                ;MOVMEDIAIDS equ mov_media_ids - DOSBIOSEG_2C7h ; (751h for MSDOS 5.0 IO.SYS)
8394                                ;CLEARIDS equ clear_ids - DOSBIOSEG_2C7h ; (5D9h for MSDOS 5.0 IO.SYS)
8395                                ; 09/12/2022
8396                                MOVMEDIAIDS equ mov_media_ids
8397                                CLEARIDS equ clear_ids
8398                                ; 11/09/2023
8399                                CLEARIDS_X equ clear_ids_x
8400
8401                                copybpb_fat:
8402                                        ; 17/12/2023
8403                                        ; ch = 0, cl = number of FATs
8404                                        ; 10/12/2022
8405                                        ; (number of FATs optimization)
8406                                        ; SI = disksector+11
8407                                        ; 17/10/2022
8408                                        ;mov    si, disksector+11
8409                                        ;;mov   si, 159h       ; disksector+EXT_BOOT.BPB
8410                                                               ; cs:si -> bpb in boot
8411                                        ; 17/12/2023
8412                                        ; dx = 0
8413                                        ; 08/05/2024
8414                                        ; 18/04/2024 (BugFix)
8415 000023DF 31D2                        xor     dx, dx
8416
8417                                        ; 12/08/2023
8418                                        ; ds = cs = BIOSDATA segment (0070h)
8419 000023E1 8B4408                      mov     ax, [si+8]
8420                                        ;mov    ax, [cs:si+8]  ; [cs:si+EBPB.TOTALSECTORS]
8421                                                               ; get totsec from boot sec
8422 000023E4 09C0                        or      ax, ax
8423 000023E6 7514                        jnz     short copy_totsec ; if non zero, use that
8424 000023E8 8B4415                      mov     ax, [si+15h] ; 12/08/2023
8425                                        ;mov    ax, [cs:si+15h]       ; [cs:si+EBPB.BIGTOTALSECTORS]
8426                                                               ; get the big version
8427                                                               ; (32 bit total       sectors)
8428 000023EB 8B5417                      mov     dx, [si+17h] ; 12/08/2023
8429                                        ;mov    dx, [cs:si+17h]       ; [cs:si+EBPB.BIGTOTALSECTORS+2]
8430                                        ; 10/12/2022
8431                                        ; (number of FATs optimization)
8432                                        ; CL = number of FATs (2 or 1)
8433 000023EE 89D3                        mov     bx, dx         ; see if it is a big zero
8434 000023F0 09C3                        or      bx, ax
8435 000023F2 7508                        jnz     short copy_totsec
8436                                        ; screw it. it was bogus.
8437 000023F4 8B451B                      mov     ax, [di+1Bh]   ; [di+BDS.totalsecs32]
8438 000023F7 8B551D                      mov     dx, [di+1Dh]   ; [di+BDS.totalsecs32+2]
8439 000023FA EB06                        jmp     short fat_big_small
8440
8441                                        ;mov    cx, dx
```

```
                                        ;or     cx, ax          ; see if it is a big zero
                                        ;jz     short totsec_already_set ; screw it. it     was bogus.
                                copy_totsec:
000023FC 89451B                         mov     [di+1Bh], ax    ; [di+BDS.totalsecs32]
                                                                ; make DPB match boot sec
000023FF 89551D                         mov     [di+1Dh], dx    ; [di+BDS.totalsecs32+2]

                                        ; 10/12/2022
                                ;totsec_already_set:
                                        ;mov    ax, [di+1Bh]    ; [di+BDS.totalsecs32]
                                        ;mov    dx, [di+1Dh]    ; [di+BDS.totalsecs32+2]

                                ; determine fat entry size.

                                fat_big_small:

                                ;at this moment dx;ax = total sector number

                                ;Do not assume 1 reserved sector. Update the reserved sector field in BDS
                                ;from the BPB on the disk

                                        ; 12/08/2023
                                        ; ds = cs = BIOSDATA segment (0070h)

00002402 8B5C03                         mov     bx, [si+3]
                                        ;mov    bx, [cs:si+3]  ; [cs:si+EBPB.RESERVEDSECTORS]
                                                                ; get #reserved_sectors       from BPB
00002405 895D09                         mov     [di+9], bx      ; [di+BDS.resectors]
                                                                ; update BDS field
00002408 29D8                           sub     ax, bx
0000240A 83DA00                         sbb     dx, 0           ; update the count
                                        ; 12/08/2023
0000240D 8B5C0B                         mov     bx, [si+0Bh]
                                        ;mov    bx, [cs:si+0Bh]       ; [cs:si+EBPB.SECTORSPERFAT]
                                                                ; bx = sectors/fat
00002410 895D11                         mov     [di+11h], bx    ; [di+BDS.fatsecs]
                                                                ; set in bds bpb
                                        ; 17/12/2023 - Retro DOS v5.0
                                        ;             (PCDOS 7.1 IBMBIO.COM)
00002413 53                             push    bx ; FAT sectors
00002414 09DB                           or      bx, bx
00002416 753A                           jnz     short fat_16bit

                                ; 17/12/2023
                                %if 0
                                        sub     ax, [si+19h]    ; FAT32 file system (PCDOS 7.1 BUG!)
                                                                ; BPB.FATSz32
                                        sbb     dx, [si+1Bh]    ; BPB.FATSz32+2 (PCDOS 7.1 BUG!)
                                        ; dx:ax = partition size - (one FAT sectors + reserved sects)
                                        mov     bx, [si+19h]    ; BPB.FATSz32
                                        mov     [di+1Fh], bx    ; [di+BDS.fatsecs32]
                                        mov     bx, [si+1Bh]    ; BPB.FATSz32+2
                                        mov     [di+21h], bx    ; [di+BDS.fatsecs32+2]
                                        mov     bx, [si+1Dh]    ; BPB.BPB_ExtFlags
                                        mov     [di+23h], bx    ; [di+BDS.extflags]
                                        mov     bx, [si+1Fh]    ; BPB.FSVer
                                        mov     [di+25h], bx    ; [di+BDS.fsver]
                                        mov     bx, [si+21h]    ; BPB.RootClus
                                        mov     [di+27h], bx    ; [di+BDS.rootdirclust]
                                        mov     bx, [si+23h]    ; BPB.RootClus+2
                                        mov     [di+29h], bx    ; [di+BDS.rootdirclust+2]
                                        mov     bx, [si+25h]    ; BPB.FSInfo
                                        mov     [di+2Bh], bx    ; [di+BDS.fsinfo]
                                        mov     bx, [si+27h]    ; BPB.FSInfo+2
                                        mov     [di+2Dh], bx    ; [di+BDS.fsinfo+2]
                                        jmp     short fat_32bit       ; PCDOS 7.1 BUG! Erdogan Tan - 8/8/2023
                                                                ; correct code (would be):
                                                                ;   mov cl, [cs:si+05h] ; BPB_NumFATs
                                                                ; sub_fat32_size:
                                                                ;   sub ax, [cs:si+19h] ; BPB_FATSz32
                                                                ;   sbb dx, [cs:si+1Bh] ; BPB_FATSz32+2
                                                                ;   dec cl
                                                                ;   jg short sub_fat32_size
                                                                ;   jmp short fat_32bit
                                %endif
                                        ; 17/12/2023
                                        ; cl = BPB_NumFATs (2 or 1)
                                        ; ch = 0
00002418 8B5C19                         mov     bx, [si+19h]    ; BPB.FATSz32
                                sub_fat32_size:
0000241B 29D8                           sub     ax, bx
0000241D 1B541B                         sbb     dx, [si+1Bh]    ; BPB.FATSz32+2
                                        ;dec    cl
00002420 49                             dec     cx
00002421 7FF8                           jg      short sub_fat32_size

00002423 895D1F                         mov     [di+1Fh], bx    ; [di+BDS.fatsecs32]
00002426 8B5C1B                         mov     bx, [si+1Bh]    ; BPB.FATSz32+2
00002429 895D21                         mov     [di+21h], bx    ; [di+BDS.fatsecs32+2]

0000242C 8B5C1D                         mov     bx, [si+1Dh]    ; BPB.BPB_ExtFlags
0000242F 895D23                         mov     [di+23h], bx    ; [di+BDS.extflags]
00002432 8B5C1F                         mov     bx, [si+1Fh]    ; BPB.FSVer
00002435 895D25                         mov     [di+25h], bx    ; [di+BDS.fsver]
00002438 8B5C21                         mov     bx, [si+21h]    ; BPB.RootClus
0000243B 895D27                         mov     [di+27h], bx    ; [di+BDS.rootdirclust]
0000243E 8B5C23                         mov     bx, [si+23h]    ; BPB.RootClus+2
00002441 895D29                         mov     [di+29h], bx    ; [di+BDS.rootdirclust+2]
00002444 8B5C25                         mov     bx, [si+25h]    ; BPB.FSInfo
00002447 895D2B                         mov     [di+2Bh], bx    ; [di+BDS.fsinfo]
0000244A 8B5C27                         mov     bx, [si+27h]    ; BPB.FSInfo+2
0000244D 895D2D                         mov     [di+2Dh], bx    ; [di+BDS.fsinfo+2]
00002450 EB08                           jmp     short fat_32bit

                                fat_16bit:
                                        ; 17/12/2023 - Retro DOS v5.0
                                        ;             (PCDOS 7.1 IBMBIO.COM)
                                        ; 10/12/2022
                                        ; (number of FATs optimization)
                                        ; CL = number of FATs (2 or 1)
                                        ; CH = 0 ; 17/12/2023
                                        ;dec    cl ; *
                                        ; 18/12/2022
00002452 49                             dec     cx ; *
00002453 D3E3                           shl     bx, cl
                                        ;shl    bx, 1   ; =*?=  ; always 2 fats

00002455 29D8                           sub     ax, bx          ; sub # fat sectors
00002457 83DA00                         sbb     dx, 0
                                fat_32bit:
                                        ; 17/12/2023
0000245A 8B5C06                         mov     bx, [si+6] ; 12/08/2023
                                        ;mov    bx, [cs:si+6]  ; [cs:si+EBPB.ROOTENTRIES]
                                                                ; # root entries
```

```
8566 0000245D 895D0C                    mov     [di+0Ch], bx   ; [di+BDS.direntries]
8567                                    ; set in bds bpb
8568 00002460 B104                      mov     cl, 4
8569 00002462 D3EB                      shr     bx, cl         ; div by 16 ents/sector
8570 00002464 29D8                      sub     ax, bx         ; sub # dir sectors
8571 00002466 83DA00                    sbb     dx, 0          ;
8572                                    ; dx:ax now contains the
8573                                    ; # of data sectors
8574                                    ; 17/12/2023
8575                                    ; ch = 0
8576                                    ;xor    cx, cx ; *
8577 00002469 8A4C02                    mov     cl, [si+2] ; 12/08/2023
8578                                    ;mov    cl, [cs:si+2]  ; [cs:si+EBPB.SECTORSPERCLUSTER]
8579                                    ; sectors per cluster
8580 0000246C 884D08                    mov     [di+8], cl     ; [di+BDS.secperclus]
8581                                    ; set in bios bpb
8582 0000246F 50                        push    ax
8583 00002470 89D0                      mov     ax, dx
8584 00002472 31D2                      xor     dx, dx
8585 00002474 F7F1                      div     cx             ; cx = sectors per cluster
8586                                    ; 12/08/2023 (ds=cs)
8587                                    ;mov    [temp_h], ax
8588                                    ;;mov   [cs:temp_h], ax       ; [temp_h]:ax now contains the
8589                                    ; # clusters.
8590                                    ; 17/12/2023
8591 00002476 A3[9E04]                  mov     [saved_word], ax ; hw of cluster number
8592 00002479 58                        pop     ax
8593 0000247A F7F1                      div     cx
8594                                    ; 17/12/2023
8595                                    ;;cmp   word [cs:temp_h], 0
8596                                    ;cmp    word [temp_h], 0  ; 12/08/2023
8597                                    ;cmp    word [saved_word], 0 ; (*)
8598                                    ;ja     short toobig_ret ; too big cluster number
8599
8600                                    ; 17/12/2023
8601                                    ;;;
8602 0000247C 5B                        pop     bx ; FAT sectors (16 bit)
8603                                    ;and    bx, bx ; 0 ?
8604 0000247D 09DB                      or      bx, bx ; 0 ?
8605 0000247F 751F                      jnz     short chk_clnum_hw
8606                                            ; 16 bit fat sectors > 0 ; FAT12 or FAT16 fs
8607
8608 00002481 813E[9E04]FF0F                    cmp     word [saved_word], 0FFFh
8609 00002487 7503                              jne     short fat32_clust_limit
8610 00002489 83F8F6                            cmp     ax, 0FFF6h     ; FAT32 cluster number limit: 0FFFFFF6h
8611                                    fat32_clust_limit:
8612 0000248C 772D                              ja      short short toobig_ret ; too big cluster number
8613 0000248E 391E[9E04]                        cmp     [saved_word], bx ; 0 ?
8614                                            ;jnz    short fat16_clust_limit
8615 00002492 7505                              jnz     short set_fbigbig_flag ; 17/12/2023
8616                                    fat16_clust_limit: ; 17/12/2023
8617 00002494 83F8F6                            cmp     ax, 0FFF6h     ; FAT16 cluster number limit: 0FFF6h
8618                                    ;fat16_clust_limit:
8619 00002497 760E                              jna     short fat12_clust_limit ; jbe
8620                                    set_fbigbig_flag:  ; 17/12/2023
8621 00002499 800E[061A]20                      or      byte [fbigfat], 20h ; fbigbig ; FAT32 fs
8622 0000249E EB11                              jmp     short copymediaid
8623                                    chk_clnum_hw:
8624 000024A0 833E[9E04]00                      cmp     word [saved_word], 0 ; (*)
8625 000024A5 7714                              ja      short toobig_ret ; too big cluster number
8626                                            ;;;
8627                                    fat12_clust_limit:
8628 000024A7 3DF60F                            cmp     ax, 0FF6h      ; 4096-10
8629                                            ; is this 16-bit fat?
8630 000024AA 7205                              jb      short copymediaid ; no,       small fat
8631                                            ; 17/10/2022
8632 000024AC 800E[061A]40                      or      byte [fbigfat], 40h ; fbig ; FAT16 fs
8633                                            ;or     ds:fbigfat, 40h       ; fbig
8634                                            ; 16 bit fat
8635                                    copymediaid:
8636                                            ; 17/12/2023
8637                                            ; es = ds = cs
8638
8639                                            ;push   es
8640                                            ;push   ds
8641                                            ;pop    es
8642
8643                                            ; 12/08/2023
8644                                            ; ds = cs = BIOSDATA
8645                                            ;push   cs
8646                                            ;pop    ds
8647                                            ; 17/10/2022
8648 000024B1 BD[4F08]                          mov     bp, MOVMEDIAIDS
8649                                            ;mov    bp, 865h       ; (PCDOS 7.1 IBMBIO.COM)
8650                                            ;;mov   bp, 751h       ; mov_media_ids
8651                                            ; at 2C7h:751h = 70h:2CC1h
8652                                            ; copy filesys_id, volume label
8653 000024B4 0E                                push    cs             ; simulate far call
8654 000024B5 E8B9F5                            call    call_bios_code
8655
8656                                            ; 12/08/2023
8657                                            ;push   es
8658                                            ;pop    ds
8659                                            ; 17/12/2023
8660                                            ;pop    es
8661
8662 000024B8 E9CD00                            jmp     massage_bpb    ; now final check for bpb info
8663                                            ; and return.
8664                                    ; ---------------------------------------------------------------------------
8665
8666                                    toobig_ret:
8667                                            ; 12/08/2023 (ds=cs=BIOSDATA)
8668 000024BB 800E[061A]80                      or      byte [fbigfat], 80h ; ftoobig
8669                                            ;or     byte [cs:fbigfat], 80h ; ftoobig
8670                                            ; too big (32 bit clust #) for FAT16
8671 000024C0 E9E300                            jmp     goodret        ; still drive letter is       assigned
8672                                            ; but useless. to big for
8673                                            ; current pc dos fat file system
8674                                    ; ---------------------------------------------------------------------------
8675
8676                                    unknown:
8677                                            ; 17/12/2023 - Retro DOS v5.0 (PCDOS 7.1 IBMBIO.COM)
8678 000024C3 804D4002                          or      byte [di+40h], 2 ; [di+BDS.flags+1]
8679                                             ; unformatted_media
8680                                            ; 12/12/2022
8681                                            ;or     byte [di+24h], 02h
8682                                            ;;or    word [di+23h], 200h ; [di+BDS.flags]
8683                                            ; unformatted_media
8684                                            ; Set unformatted media       flag.
8685
8686                                    ; the boot signature may not be   recognizable,
8687                                    ; but we should    try and read it anyway.
8688
8689                                    unknown3_0:
```

```
8690 000024C7 8B551D                    mov     dx, [di+1Dh]   ; skip setting unformatted_media bit
8691                                                           ; [di+BDS.totalsecs32+2]
8692 000024CA 8B451B                    mov     ax, [di+1Bh]   ; [di+BDS.totalsecs32]
8693 000024CD BE[161A]                  mov     si, disktable2
8694                                scan:                                  ; 08/08/2023
8695                                    ;cmp    dx, [cs:si]    ; total sectors hw
8696                                    ; 12/08/2023 (ds=cs)
8697 000024D0 3B14                      cmp     dx, [si]
8698 000024D2 720C                      jb      short gotparm
8699 000024D4 7705                      ja      short scan_next
8700                                    ;cmp    ax, [cs:si+2]  ; total sectors lw
8701 000024D6 3B4402                    cmp     ax, [si+2]
8702 000024D9 7605                      jbe     short gotparm
8703                                scan_next:
8704 000024DB 83C60A                    add     si, 10         ; 5*2
8705 000024DE EBF0                      jmp     short scan     ; covers upto 512 mb media
8706                                ; ---------------------------------------------------------------------------
8707
8708                                gotparm:
8709 000024E0 8A4C08                    mov     cl, [si+8]     ; fat size for fbigfat flag
8710                                    ;or     ds:fbigfat, cl
8711                                    ; 17/10/2022
8712 000024E3 080E[061A]                or      [fbigfat], cl  ; (fbig flag, 40h or 0) ; 08/08/2023
8713                                    ; 12/08/2023
8714                                    ; ds = cs = BIOSDATA
8715 000024E7 8B4C04                    mov     cx, [si+4]
8716                                    ;mov    cx, [cs:si+4]  ; ch = number of sectors per cluster
8717                                                           ; cl = log base 2 of ch
8718 000024EA 8B5406                    mov     dx, [si+6]
8719                                    ;mov    dx, [cs:si+6]  ; dx = number of root dir entries
8720
8721                                ; now calculate size of fat table
8722
8723 000024ED 89550C                    mov     [di+0Ch], dx   ; [di+BDS.direntries]
8724                                                           ; save number of (root)      dir entries
8725 000024F0 8B551D                    mov     dx, [di+1Dh]   ; [di+BDS.totalsecs32+2]
8726 000024F3 8B451B                    mov     ax, [di+1Bh]   ; [di+BDS.totalsecs32]
8727 000024F6 886D08                    mov     [di+8], ch     ; [di+BDS.secperclus]
8728                                                           ; save sectors per cluster
8729
8730                                    ; 17/12/2023 - Retro DOS v5.0 (PCDOS 7.1 IBMBIO.COM)
8731 000024F9 F606[061A]60              test    byte [fbigfat], 60h ; fbig+fbigbig ; FAT16 or FAT32
8732                                    ; 11/09/2023
8733                                    ; 17/10/2022
8734                                    ;test   byte [fbigfat], 40h
8735                                    ;;test  ds:fbigfat, 40h        ; fbig
8736                                                           ; if (fbigfat)
8737 000024FE 751E                      jnz     short dobig    ; goto dobig; (16 bit fat)
8738
8739                                ; we don't need to change "small fat" logic since it is guaranteed
8740                                ; that double word total sector will not use 12 bit fat (unless
8741                                ; it's sectors/cluster >= 16 which will never be in this case.)
8742                                ; so in this case we assume dx = 0 !!
8743
8744 00002500 31DB                          xor     bx, bx         ; 12 bit fat (FAT12 fs)
8745 00002502 88EB                          mov     bl, ch
8746 00002504 4B                            dec     bx
8747 00002505 01C3                          add     bx, ax         ; dx=0
8748 00002507 D3EB                          shr     bx, cl         ; bx = 1+(bpb->maxsec+BDS.secperclus-1)/
8749 00002509 43                            inc     bx             ; BDS.secperclus
8750 0000250A 80E3FE                        and     bl, 0FEh       ; bx &= ~1; (=number of clusters)
8751 0000250D 89DE                          mov     si, bx
8752 0000250F D1EB                          shr     bx, 1
8753 00002511 01F3                          add     bx, si         ; number of FAT bytes ; 08/08/2023
8754 00002513 81C3FF01                      add     bx, 511        ; bx += 511 + bx/2
8755 00002517 D0EF                          shr     bh, 1          ; bh >>= 1; (=bx/512)
8756 00002519 887D11                        mov     [di+11h], bh   ; [di+BDS.fatsecs]
8757                                                               ; save number of fat sectors
8758 0000251C EB6A                          jmp     short massage_bpb
8759                                ; ---------------------------------------------------------------------------
8760
8761                                ; for bigfat we do need to extend this logic to 32 bit sector calculation.
8762
8763                                dobig:
8764 0000251E B104                          mov     cl, 4          ; 16 (2^4) directory entries per sector
8765 00002520 52                            push    dx             ; save total sectors (high)
8766 00002521 8B550C                        mov     dx, [di+0Ch]   ; [di+BDS.direntries]
8767 00002524 D3EA                          shr     dx, cl         ; root dir sectors = BDS.direntries / 16;
8768 00002526 29D0                          sub     ax, dx
8769 00002528 5A                            pop     dx
8770 00002529 83DA00                        sbb     dx, 0          ; dx:ax = total sectors - root dir sectors
8771 0000252C 83E801                        sub     ax, 1
8772 0000252F 83DA00                        sbb     dx, 0          ; dx:ax = t - r - d
8773                                                               ; total secs - reserved       secs - root dir       secs
8774 00002532 B302                          mov     bl, 2
8775 00002534 8A7D08                        mov     bh, [di+8]     ; [di+BDS.secperclus]
8776                                                               ; bx = 256 * BDS.secperclus + 2
8777
8778                                ; I don't understand why to add bx here!!!
8779
8780                                        ; 29/12/2018 - Erdogan Tan (Retro DOS v4.0)
8781                                        ; 27/09/2022
8782                                        ; (Microsoft FAT32 File      System Specification,
8783                                        ; December 2000, Page 21)
8784                                        ; TmpVal1 = DskSize - (BPB_ResvdSecCnt+RootrDirSectors)
8785                                        ; TmpVal2 = (256*BPB_SecPerClus)+BPB_NumFATs
8786                                        ; 8/8/2023 (Retro DOS v5.0)
8787                                        ; If(FATType == FAT32)
8788                                        ;   TmpVal2 = TmpVal2 / 2;
8789                                        ; FATsz = (TmpVal1+(TmpVal2-1))/TmpVal2
8790                                        ; 8/8/2023 (Retro DOS v5.0)
8791                                        ; If(FATType == FAT32) {
8792                                        ;   BPB_FATSz16 = 0;
8793                                        ;   BPB_FATSz32 = FATSz;
8794                                        ;} else {
8795                                        ;   BPB_FATSz16 = LOWORD(FATSz);
8796                                        ;/* there is no BPB_FATSz32 in a FAT16 BPB */
8797                                        ;}
8798                                                               ; dx:ax = TmpVal1, bx = TmpVal2
8799 00002537 01D8                        add     ax, bx         ;
8800 00002539 83D200                    adc     dx, 0          ; dx:ax = TmpVal1+TmpVal2
8801 0000253C 83E801                    sub     ax, 1
8802 0000253F 83DA00                    sbb     dx, 0          ; dx:ax = TmpVal1+TmpVal2-1
8803
8804                                    ;;;
8805                                    ; 17/12/2023 - Retro DOS v5.0 (PCDOS 7.1 IBMBIO.COM)
8806 00002542 F606[061A]20              test    byte [fbigfat], 20h ; fbigbig (FAT32) flag
8807 00002547 740D                      jz      short dobig1
8808
8809 00002549 D1EB                      shr     bx, 1           ; TmpVal2 = TmpVal2 / 2
8810                                                           ; dx:ax = TmpVal1+(2*TmpVal2)-1
8811 0000254B 83E81F                    sub     ax, 31          ; reserved sectors = 32 (for FAT32 fs) /// 1+31 = 32
8812 0000254E 83DA00                    sbb     dx, 0
8813 00002551 29D8                      sub     ax, bx
```

```
00002553 83DA00                         sbb     dx, 0           ; dx:ax = TmpVal1+(2*TmpVal2)-TmpVal2-1
                                                                ;       = TmpVal1+(TmpVal2-1)
                                dobig1:
00002556 50                             push    ax              ; save lw of dividend
00002557 89D0                           mov     ax, dx          ; divide hw of dx:ax at first (as 1st stage)
00002559 31D2                           xor     dx, dx
0000255B F7F3                           div     bx              ; 32 bit division, dx:ax/bx
                                                                ; remainder in dx is hw of 2nd stage dividend
0000255D 89C5                           mov     bp, ax          ; hw of quotient
0000255F 58                             pop     ax              ; restore lw of dividend (of 1st stage)
                                        ;;;

                                ; assuming dx in the table will never be bigger than bx.

00002560 F7F3                           div     bx              ; BDS.fatsecs =
                                                                ; ceil((total-dir-res)/(256*BDS.secperclus+2))
00002562 894511                         mov     [di+11h], ax    ; [di+BDS.fatsecs]
                                                                ; number of fat       sectors
                                        ;;;

                                        ; 18/12/2023 - Retro DOS v5.0 (PCDOS 7.1 IBMBIO.COM)
00002565 8A1E[061A]                     mov     bl, [fbigfat]
00002569 885D3B                         mov     [di+3Bh], bl    ; [di+BDS.fatsiz] ; fat size flag

0000256C F6C320                         test    bl, 20h         ; fbigbig (FAT32) flag
0000256F 7410                           jz      short dobig2    ; not FAT32

00002571 89451F                         mov     [di+1Fh], ax    ; [di+BDS.fatsecs32]
00002574 896D21                         mov     [di+21h], bp    ; [di+BDS.fatsecs32+2]
00002577 C745110000                     mov     word [di+11h], 0 ; [di+BDS.fatsecs] = 0
                                                                ; clear 16 bit FAT size field
0000257C C745092000                     mov     word [di+9], 32         ; [di+BDS.resectors]
                                                                ; set reserved sectors to 32 (FAT32 de facto)
                                dobig2:
                                        ;;;

                                ; now, set the default filesys_id, volume label, serial number

                                        ; 05/08/2023
                                        ; [di+1Fh] = [fbigfat]
                                        ;
                                        ;;mov   bl, ds:fbigfat
                                        ;; 17/10/2022
                                        ;mov    bl, [fbigfat]
                                        ;mov    [di+1Fh], bl    ; [di+BDS.fatsiz] ; fat       size flag

                                        ; 12/08/2023
                                        ;push   ds ; ds = cs = BIOSDATA

                                        ; 17/12/2023
                                        ; es = ds = cs
                                        ;push   ds
                                        ;pop    es

                                        ; 12/08/2023
                                        ; ds = cs = BIOSDATA
                                        ;push   cs
                                        ;pop    ds

                                        ; 18/12/2023 - Retro DOS v5.0
                                        ; bl = [fbigfat] (clear_ids_x uses bl value here)
                                        ; 11/09/2023
                                        ;mov    al, [fbigfat]
00002581 BD[A106]                       mov     bp, CLEARIDS_X ; clear_ids_x (uses AL value here)
                                        ; 17/10/2022
                                        ;mov    bp, CLEARIDS
                                        ;;mov   bp, 5D9h        ; clear_ids
                                                                ; at 2C7h:5D9h = 70h:2B49h
                                                                ; at BIOSCODE:06ABh
                                                                ;       in PCDOS 7.1 IBMBIO.COM
00002584 0E                             push    cs
00002585 E8E9F4                         call    call_bios_code

                                        ; 12/08/2023
                                        ;pop    ds ; ds = cs = BIOSDATA

                                ; at this point, in bpb of bds table, BDS_BPB.BPB_BIGTOTALSECTORS which is
                                ; set according to the partition information. we are going to
                                ; see if (hidden sectors + total sectors) > a word. if it is true,
                                ; then no change. otherwise, BDS_BPB.BPB_BIGTOTALSECTORS will be moved
                                ; to BDS_BPB.BPB_TOTALSECTORS and BDS_BPB.BPB_BIGTOTALSECTORS will be set to 0.
                                ; we don't do this for the bpb information from the boot record. we
                                ; are not going to change the bpb information from the boot record.

                                massage_bpb:
                                        ; 05/08/2023
                                        ; [di+1Fh] = [fbigfat]
                                        ;
                                        ;; 12/12/2022
                                        ;mov    bl, [fbigfat]
                                        ;mov    [di+1Fh], bl    ; [di+BDS.fatsiz]
                                        ;                       ; set size of fat on media
                                        ;
00002588 8B551D                         mov     dx, [di+1Dh]    ; [di+BDS.totalsecs32+2]
0000258B 8B451B                         mov     ax, [di+1Bh]    ; [di+BDS.totalsecs32]
                                        ; 11/09/2023
0000258E 09D2                           or      dx, dx
00002590 7514                           jnz     short goodret
                                        ;cmp    dx, 0           ; double word total sectors?
                                        ;;ja    short goodret   ; don't have to change it.
                                        ;; 12/12/2022
                                        ;ja     short short goodret2
                                        ;cmp    word [di+19h], 0 ; [di+BDS.hiddensecs+2]
                                        ;ja     short goodret   ; don't have to change it.
                                        ; 12/12/2022
00002592 395519                         cmp     [di+19h], dx ; 0
                                        ;ja     short goodret2
00002595 770F                           ja      short goodret   ; 11/09/2023
00002597 034517                         add     ax, [di+17h]    ; [di+BDS.hiddensecs]
                                        ;jb     short goodret
                                        ; 12/12/2022
                                        ;jc     short goodret
0000259A 7209                           jc      short goodret_clc ; 11/09/2023
0000259C 8B451B                         mov     ax, [di+1Bh]    ; [di+BDS.totalsecs32]
0000259F 89450E                         mov     [di+0Eh], ax    ; [di+BDS.totalsecs16]
                                        ;mov    word [di+1Bh], 0 ; [di+BDS.totalsecs32]
                                        ; 12/12/2022
000025A2 89551B                         mov     [di+1Bh], dx ; 0
                                goodret_clc:
                                        ; 11/09/2023
000025A5 F8                             clc
                                goodret:
                                        ;mov    bl, ds:fbigfat
                                        ; 11/09/2023
```

```
                                        ; 12/12/2022
                                        ; 17/10/2022
000025A6 8A1E[061A]                     mov     bl, [fbigfat]
                                        ; 17/12/2023 - Retro DOS v5.0 (PCDOS 7.1 IBMBIO.COM)
000025AA 885D3B                         mov     [di+3Bh], bl   ; [di+BDS.fatsiz]
                                        ;mov    [di+1Fh], bl   ; [di+BDS.fatsiz]
                                                               ; set size of fat on media
                                        ; 11/09/2023
                                        ;clc
                                ret_hard_err:
                                        ; 12/12/2022
                                goodret2:
000025AD 07                             pop     es
                                        ;pop    ds      ; ds = cs = BIOSDATA ; 14/08/2023
000025AE 5B                             pop     bx
000025AF 5F                             pop     di
000025B0 C3                             retn

                                ; =============== S U B   R O U T I N E =======================================

                                ; 15/10/2022

                                ;fdisk of pc dos 3.3 and below, os2 1.0 has a bug. the maximum number of
                                ;sector that can be handled by pc dos 3.3 ibmbio should be 0ffffh.
                                ;instead, sometimes fdisk use 10000h to calculate the maximum number.
                                ;so, we are going to check that if BPB_TOTALSECTORS + hidden sector = 10000h
                                ;then subtract 1 from BPB_TOTALSECTORS.

                                        ; 17/10/2022
                                cover_fdisk_bug:
                                        ; 12/08/2023 - Retro DOS v4.2 IO.SYS (optimization)
                                        ; ds = cs

                                        ; 18/12/2023 - Retro DOS v5.0 (PCDOS 7.1 IBMBIO.COM)
                                        ; (optimization)
                                        ;push   ax
                                        ;push   dx
                                        ;push   si

                                        ; 18/12/2023
                                        ; bx = offset disksector

                                        ; 18/12/2023
000025B1 807F2629                       cmp     byte [bx+26h], 29h
                                        ; 12/08/2023
                                        ;cmp    byte [disksector+26h], 29h
                                        ;;cmp   byte [cs:disksector+26h], 29h
                                                               ; [disksector+EXT_BOOT.SIG],
                                                               ; EXT_BOOT_SIGNATURE
000025B5 7426                           je      short cfb_retit       ; if extended bpb, then       >= pc dos 4.00

000025B7 817F073130                     cmp     word [bx+7], 3031h
                                        ;cmp    word [cs:bx+7], 3031h ; '10' ; os2 1.0 = ibm 10.0
000025BC 7506                           jne     short cfb_chk_totalsecs ; 11/08/2023
000025BE 807F0A30                       cmp     byte [bx+10], '0'
                                        ;cmp    byte [cs:bx+10], '0'
000025C2 7519                           jne     short cfb_retit

                                cfb_chk_totalsecs:
                                        ; 11/08/2023
                                ; 18/12/2023
                                %if 0
                                        ; 17/10/2022
                                        mov     si, disksector+11 ; 14Eh+0Bh
                                        ;mov    si, 159h       ; disksector+EXT_BOOT.BPB
                                        ; 12/08/2023
                                        cmp     word [si+8], 0
                                        ;cmp    word [cs:si+8], 0 ; [cs:si+EBPB.TOTALSECTORS]
                                                               ; just to make sure.
                                        jz      short cfb_retit
                                        ;mov    ax, [cs:si+8]  ; [cs:si+EBPB.TOTALSECTORS]
                                        ;add    ax, [cs:si+11h]       ; [cs:si+EBPB.HIDDENSECTORS]
                                        ; 12/08/2023
                                        mov     ax, [si+8]
                                        add     ax, [si+11h]

                                        jnb     short cfb_retit
                                        jnz     short cfb_retit
                                                               ; if carry set and ax=0
                                        dec     word [si+8]
                                        ;dec    word [cs:si+8] ; 0 -> 0FFFFh
                                                               ; then decrease       BPB_TOTALSECTORS by 1
                                %endif
                                        ; 18/12/2023
                                        ;cmp    word [bx+19], 0
000025C4 8B4713                         mov     ax, [bx+19]    ; [bx+EBPB.TOTALSECTORS]
000025C7 21C0                           and     ax, ax ; 0 ?
000025C9 7412                           jz      short cfb_retit

                                        ;mov    ax, [bx+19]
000025CB 03471C                         add     ax, [bx+28]    ; [bx+EBPB.HIDDENSECTORS]
000025CE 730D                           jnc     short cfb_retit
000025D0 750B                           jnz     short cfb_retit
                                        ; ax = 0               ; 0 -> 0FFFFh
000025D2 FF4F13                         dec     word [bx+19]   ; then decrease       BPB_TOTALSECTORS by 1

000025D5 836D1B01                       sub     word [di+1Bh], 1 ; [di+BDS.totalsecs32]
000025D9 835D1D00                       sbb     word [di+1Dh], 0 ; [di+BDS.totalsecs32+2]
                                cfb_retit:
                                        ; 18/12/2023
                                        ;pop    si
                                        ;pop    dx
                                        ;pop    ax

000025DD C3                             retn

                                ; ---------------------------------------------------------------------------

000025DE 0200                   word2:          dw 2
000025E0 0300                   word3:          dw 3
000025E2 0002                   word512:    dw 512

                                ; =============== S U B   R O U T I N E =======================================

                                ; 15/10/2022

                                ; setdrvparms sets up the recommended bpb in each bds in the system based on
                                ; the form factor. it is assumed that the bpbs for the various form factors
                                ; are present in the bpbtable. for hard files, the recommended bpb is the same
                                ; as the bpb on the drive.
                                ;
                                ; no attempt is made to preserve registers since we are going to jump to
                                ; sysinit straight after this routine.
```

```
9062                                    ; 18/12/2023 - Retro DOS v5.0
9063                                    ; (PCDOS 7.1 IBMBIO.COM - BIOSDATA:2A43h)
9064                                setdrvparms:
9065                                    ; 12/12/2023
9066                                    ; ds = cs
9067 000025E4 31DB                     xor     bx, bx
9068                                    ; 18/10/2022
9069 000025E6 C43E[1901]                les     di, [start_bds] ; get first bds in list
9070                                _next_bds:
9071 000025EA 06                        push    es
9072 000025EB 57                        push    di
9073
9074                                    ; 18/12/2023 - Retro DOS v5.0 (PCDOS 7.1 IBMBIO.COM)
9075 000025EC 268A5D3E                  mov     bl, [es:di+3Eh]         ; [es:di+BDS.formfactor]
9076                                    ;mov    bl, [es:di+22h]         ; [es:di+BDS.formfactor]
9077
9078 000025F0 80FB05                    cmp     bl, 5           ; ffHardFile
9079 000025F3 753A                      jnz     short nothardff
9080 000025F5 31D2                      xor     dx, dx
9081 000025F7 268B450E                  mov     ax, [es:di+0Eh]         ; [es:di+BDS.totalsecs16]
9082 000025FB 09C0                      or      ax, ax
9083 000025FD 7508                      jnz     short get_ccyl
9084 000025FF 268B551D                  mov     dx, [es:di+1Dh]         ; [es:di+BDS.totalsecs32+2]
9085 00002603 268B451B                  mov     ax, [es:di+1Bh]         ; [es:di+BDS.totalsecs32]
9086                                get_ccyl:
9087 00002607 52                        push    dx
9088 00002608 50                        push    ax
9089 00002609 268B4515                  mov     ax, [es:di+15h]         ; [es:di+BDS.heads]
9090 0000260D 26F76513                  mul     word [es:di+13h] ; [es:di+BDS.secpertrack]
9091                                                                    ; assume sectors per cyl. < 64k.
9092 00002611 89C1                      mov     cx, ax          ; cx has # sectors per cylinder
9093 00002613 58                        pop     ax
9094 00002614 5A                        pop     dx              ; dx:ax = total sectors
9095 00002615 50                        push    ax
9096 00002616 89D0                      mov     ax, dx
9097 00002618 31D2                      xor     dx, dx
9098 0000261A F7F1                      div     cx
9099                                    ; 12/12/2023  ; !*!
9100                                    ; (data segment may not be same with code segment here)
9101                                    ;mov    [cs:temp_h], ax         ; ax be 0 here.
9102                                    ; 18/12/2023 - Retro DOS v5.0
9103                                    ;mov    [cs:saved_word], ax
9104 0000261C 58                        pop     ax
9105 0000261D F7F1                      div     cx              ; div #sec by sec/cyl to get # cyl.
9106 0000261F 09D2                      or      dx, dx
9107 00002621 7401                      jz      short no_cyl_rnd ; came out even
9108 00002623 40                        inc     ax              ; round up
9109                                no_cyl_rnd:
9110                                    ; 18/12/2023 - Retro DOS v5.0
9111 00002624 26894541                  mov     [es:di+41h], ax         ; [es:di+BDS.cylinders]
9112                                    ;mov    [es:di+25h], ax         ; [es:di+BDS.cylinders]
9113
9114 00002628 06                        push    es
9115 00002629 1F                        pop     ds  ; !*! ; 12/12/2023
9116
9117 0000262A 8D7506                    lea     si, [di+6]      ; [di+BDS.bytespersec]
9118                                                            ; ds:si -> bpb for hard         file
9119 0000262D EB55                      jmp     short set_recbpb
9120                                ; ---------------------------------------------------------------------------
9121
9122                                nothardff:
9123 0000262F 0E                        push    cs
9124 00002630 1F                        pop     ds
9125
9126                                ; if fake floppy drive variable is set then we don't have to handle this bds.
9127                                ; we can just go and deal with the next bds at label go_to_next_bds.
9128
9129                                    ; 10/12/2022
9130                                    ; ds = cs
9131                                    ; 17/10/2022 (ds=cs)
9132 00002631 803E[111A]01              cmp     byte [fakefloppydrv], 1
9133                                    ;cmp    byte [cs:fakefloppydrv], 1
9134 00002636 7454                      jz      short go_to_next_bds
9135 00002638 80FB07                    cmp     bl, 7           ; ffother
9136                                                            ; special case "other" type of medium
9137 0000263B 753D                      jnz     short not_process_other
9138                                process_other:
9139 0000263D 31D2                      xor     dx, dx
9140
9141                                    ;mov    ax, [di+25h]    ; [di+BDS.cylinders]
9142                                    ;mul    word [di+36h]   ; [di+BDS.rheads]
9143                                    ;mul    word [di+34h]   ; [di+BDS.rsecpertrack]
9144                                    ;mov    [di+2Fh], ax    ; [di+BDS.rtotalsecs16]
9145                                    ;                       ; have the total number of sectors
9146                                    ; 18/12/2023 - Retro DOS v5.0
9147 0000263F 8B4541                    mov     ax, [di+41h]    ; [di+BDS.cylinders]
9148 00002642 F76552                    mul     word [di+52h]   ; [di+BDS.rheads]
9149 00002645 F76550                    mul     word [di+50h]   ; [di+BDS.rsecpertrack]
9150 00002648 89454B                    mov     [di+4Bh], ax    ; [di+BDS.rtotalsecs16]
9151                                                            ; have the total number of sectors
9152 0000264B 48                        dec     ax
9153 0000264C B201                      mov     dl, 1
9154                                _again:
9155 0000264E 3DF60F                    cmp     ax, 0FF6h       ; 4096-10
9156 00002651 7206                      jb      short _@@
9157 00002653 D1E8                      shr     ax, 1
9158 00002655 D0E2                      shl     dl, 1
9159 00002657 EBF5                      jmp     short _again
9160                                ; ---------------------------------------------------------------------------
9161
9162                                _@@:
9163 00002659 80FA01                    cmp     dl, 1           ; is it a small disk ?
9164 0000265C 7405                      jz      short __@@      ; yes, 224 root         entries is enuf
9165
9166                                    ; 18/12/2023 - Retro DOS v5.0
9167 0000265E C74549F000                mov     word [di+49h], 240 ; [di+BDS.rdirentries]
9168                                    ;mov    word [di+2Dh], 240 ; [di+BDS.rdirentries]
9169                                __@@:
9170                                    ; 18/12/2023 - Retro DOS v5.0
9171 00002663 885545                    mov     [di+45h], dl    ; [di+BDS.rsecperclus]
9172                                    ;mov    [di+29h], dl    ; [di+BDS.rsecperclus]
9173
9174                                ; logic to get the sectors/fat area.
9175                                ; fat entry is assumed to be 1.5 bytes!!!
9176
9177                                    ; 10/12/2022
9178                                    ; ds = cs
9179                                    ; 17/10/2022 (ds=cs)
9180 00002666 F726[E025]                 mul     word [word3]    ; * 3
9181 0000266A F736[DE25]                div     word [word2]    ; / 2
9182 0000266E 31D2                      xor     dx, dx
9183 00002670 F736[E225]                div     word [word512] ; / 512
9184                                    ;
9185                                    ; 10/12/2022
```

```
                                        ;mul    word [cs:word3]       ; * 3
                                        ;div    word [cs:word2]       ; / 2
                                        ;xor    dx, dx
                                        ;div    word [cs:word512] ; / 512
                                        ;
00002674 40                             inc     ax              ; + 1
                                no_round_up:
                                        ; 18/12/2023 - Retro DOS v5.0
00002675 89454E                         mov     [di+4Eh], ax    ; [di+BDS.rfatsecs]
                                        ;mov    [di+32h], ax    ; [di+BDS.rfatsecs]

00002678 EB12                           jmp     short go_to_next_bds
                                ; ---------------------------------------------------------------------------

                                not_process_other:
0000267A D1E3                           shl     bx, 1           ; bx is word index into       table of bpbs

                                        ;mov    si, bpbtable
                                        ;mov    si, [bpbtable+bx] ; 15/10/2022
                                        ; 09/12/2022
                                        ;mov    si, BPBTABLE
                                        ;mov    si, [bx+si]     ; get address of bpb
                                        ; 10/12/2022
                                        ;mov    si, [BPBTABLE+bx]
                                        ; 13/12/2022
                                        ;mov    si, [SYSINITOFFSET+bpbtable+bx] ; wrong ! 14/08/2023

                                        ; 14/08/2023
                                        SYSINIT_OFFSET equ (SYSINITSEG-DOSBIODATASEG<<4)
                                                                        ; correct offset
0000267C 8BB7[BE96]                     mov     si, [bx+SYSINIT_OFFSET+bpbtable]

                                        ; 18/12/2023
                                        ; si = address of the requested disk(ette) parameter block
                                        ;      ! as offset from SYSINIT segment !

                                        ; 28/08/2023
00002680 81C69046                       add     si, SYSINIT_OFFSET
                                                ; + displacement from BIOSDATA segment ; 18/12/2023
                                set_recbpb:
                                        ; 18/12/2023
                                        ;lea    di, [di+27h]    ; [di+BDS.R_BPB]
                                        ;                       ; es:di -> recbpb
                                        ;mov    cx, 25          ; bpbx.size
                                        ;rep movsb              ; move (size bpbx) bytes

                                        ; 18/12/2023 - Retro DOS v5.0
00002684 8D7D43                         lea     di, [di+43h]    ; [di+BDS.R_BPB]
                                                                ; es:di -> recbpb
00002687 B93500                         mov     cx, 53          ; bpbx.size
0000268A F3A4                           rep movsb               ; move (size bpbx) byte
                                go_to_next_bds:
0000268C 5F                             pop     di
0000268D 07                             pop     es              ; restore pointer to bds
0000268E 26C43D                         les     di, [es:di]     ; [es:di+BDS.link]
00002691 83FFFF                         cmp     di, 0FFFFh      ; -1
00002694 740A                           jz      short got_end_of_bds_chain
00002696 E951FF                         jmp     _next_bds

                                ; ---------------------------------------------------------------------------

                                        ; 18/12/2022
                                ;got_end_of_bds_chain:
                                        ;retn

                                ; =============== S U B   R O U T I N E =======================================

                                ; 15/10/2022
                                ; 30/12/2018 - Retro DOS v4.0

                                ; al = device number

                                print_init:
00002699 98                             cbw
0000269A 89C2                           mov     dx, ax
0000269C B401                           mov     ah, 1
0000269E CD17                           int     17h             ; PRINTER - INITIALIZE
                                                                ; DX = printer port (0-3)
                                                                ; Return: AH = status
                                got_end_of_bds_chain:     ; 18/12/2022
000026A0 C3                             retn

                                ; =============== S U B   R O U T I N E =======================================

                                ; al = device number

                                aux_init:
000026A1 98                             cbw
000026A2 89C2                           mov     dx, ax
                                        ;mov    al, 0A3h        ; RSINIT ; 0A3h
                                                                ; 2400,n,1,8 (msequ.inc)
                                        ;mov    ah, 0
                                        ; 10/12/2022
000026A4 B8A300                         mov     ax, 00A3h
000026A7 CD14                           int     14h             ; SERIAL I/O - INITIALIZE USART
                                                                ;       AL = initializing parameters,
                                                                ;       DX = port number (0-3)
                                                                ; Return: AH = RS-232 status code bits,
                                                                ;         AL = modem status bits
000026A9 C3                             retn

                                ; =============== S U B   R O U T I N E =======================================

                                ; 18/12/2023 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMBIO.COM)
                                ; 08/08/2023 - Retro DOS v4.2 (Modified MSDOS 6.22 IO.SYS)
                                ; 15/10/2022 (Modified MSDOS 5.0 IO.SYS) -Retro DOS v4 2022- (MSDOS 5.0-6.21)
                                ; 30/12/2018 - Retro DOS v4.0
                                ; 03/06/2018 - Retro DOS v3.0
                                ; (19/03/2018 - Retro DOS v2.0)

                                ; domini ********************************************************************************
                                ;
                                ;mini disk initialization routine. called right after dohard
                                ;modified for >2 hardfile support
                                ;
                                ; **cs=ds=es=datagrp
                                ;
                                ; **domini will search for every extended partition in the system, and
                                ;   initialize it.
                                ;
                                ; **bdsm stands for bds table for mini disk and located right after the label
                                ;   end96tpi. end_of_bdsm will have the offset value of the ending
                                ;   address of bdsm table.
                                ;
```

```
                                ; **bdsm is the same as usual bds structure except that tim_lo, tim_hi entries
                                ;   are overlapped and used to identify mini disk and the number of hidden_trks.
                                ;   right now, they are called as ismini, hidden_trks respectively.
                                ;
                                ; **domini will use the same routine in sethard routine after label set2 to
                                ;   save coding.
                                ;
                                ; **drvmax determined in dohard routine will be used for the next
                                ;   available logical mini disk drive number.
                                ;
                                ; input: drvmax, dskdrvs
                                ;
                                ; output: minidisk installed. bdsm table established and installed to bds.
                                ;     end_of_bdsm - ending offset address of bdsm.
                                ;
                                ; called modules:
                                ;             getboot
                                ;             find_mini_partition (new), xinstall_bds (new), M038
                                ;
                                ;             setmini (new, it will use set2 routine)
                                ;
                                ; variables used: end_of_bdsm
                                ;             rom_minidisk_num
                                ;             mini_hdlim, mini_seclim
                                ;             BDS_STRUC, start_bds
                                ;
                                ;******************************************************************************
                                            ; 18/12/2023 - Retro DOS v5.0 IO.SYS/IBMBIO.COM
                                            ; (PCDOS 7.1 IBMBIO.COM - BIOSDATA:2B10h)
                                
                                            ; 19/10/2022
                                domini:
000026AA 8A36[5D1A]                         mov     dh, [hnum]      ; get number of hardfiles
                                            ; 10/12/2022
000026AE 20F6                               and     dh, dh
                                            ;cmp    dh, 0
000026B0 743C                               jz      short dominiret         ; no hard file?       then exit.
000026B2 B280                               mov     dl, 80h         ; start with hardfile 80h
                                domini_loop:
                                            ; 18/12/2023 - Retro DOS v5.0
000026B4 31C0                               xor     ax, ax ; 0
                                            ; ds = cs
                                            ;mov    [cs:ep_start_sector], ax
                                            ;mov    [cs:ep_start_sector+2], ax
                                            ;mov    [cs:ep_hidden_secs], ax
                                            ;mov    [cs:ep_hidden_secs+2], ax
000026B6 A3[0022]                           mov     [ep_start_sector], ax
000026B9 A3[0222]                           mov     [ep_start_sector+2], ax
000026BC A3[0422]                           mov     [ep_hidden_secs], ax
000026BF A3[0622]                           mov     [ep_hidden_secs+2], ax
                                            ;
000026C2 52                                 push    dx
000026C3 8816[5C1A]                         mov     [rom_minidisk_num], dl
000026C7 B408                               mov     ah, 8
000026C9 CD13                               int     13h             ; DISK - DISK -       GET CURRENT DRIVE PARAMETERS (XT,AT,XT286,CONV,PS)
                                                                    ; DL = drive number
                                                                    ; Return: CF set on error, AH =      status code, BL       = drive type
                                                                    ; DL = number of consecutive drives
                                                                    ; DH = maximum value for head number, ES:DI -> drive parameter
                                
                                            ; 08/08/2023 - Retro DOS v4.2 IO.SYS (optimization)
                                            ; PCDOS 7.1 IBMBIO.COM - BIOSDATA:2B36h
                                            ;inc    dh
                                            ;xor    ax, ax
                                            ;mov    al, dh
000026CB 31C0                               xor     ax, ax
000026CD 88F0                               mov     al, dh  ; <= 255
000026CF 40                                 inc     ax      ; (0FFh -> 100h)
000026D0 A3[621A]                           mov     [mini_hdlim], ax ; # of heads
                                            ;and    cl, 3Fh
                                            ;mov    al, cl
                                            ; 08/08/2023
000026D3 88C8                               mov     al, cl
000026D5 83E03F                             and     ax, 3Fh
000026D8 A3[641A]                           mov     [mini_seclim], ax ; # of sectors/track
                                
                                            ; 18/12/2023
                                            ;push   es ; * ; not necessary
000026DB 8A16[5C1A]                         mov     dl, [rom_minidisk_num]
000026DF E866FA                             call    getboot         ; read master boot record into
                                                                    ; initbootsegment:bootbias
000026E2 7203                               jc      short domininext
000026E4 E80800                             call    find_mini_partition
                                domininext:
                                            ;pop    es ; *
000026E7 5A                                 pop     dx
000026E8 FEC2                               inc     dl              ; next hard file
000026EA FECE                               dec     dh
000026EC 75C6                               jnz     short domini_loop
                                dominiret:
000026EE C3                                 retn
                                
                                ; =============== S U B   R O U T I N E =======================================
                                
                                ; 15/10/2022 (Modified MSDOS 5.0 IO.SYS)
                                ; 30/12/2018 - Retro DOS v4.0
                                
                                ;find_mini_partition tries to find every extended partition on a disk.
                                ;at entry:  di -> bdsm entry
                                ;           es:bx -> 07c0:bootbias - master boot record
                                ;           rom_minidisk_num - rom drive number
                                ;           drvmax - logical drive number
                                ;           mini_hdlim, mini_seclim
                                ;
                                ;called routine: setmini which uses set2 (in sethard routine)
                                ;variables & equates used from original bios - flags, fnon_removable, fbigfat
                                
                                
                                            ; 19/12/2023 - Retro DOS v5.0
                                            ;       (Modified PCDOS 7.1 IBMBIO.COM)
                                            ; (PCDOS 7.1 IBMBIO.COM - BIOSADATA:2BFCh)
                                
                                find_mini_partition:
000026EF 81C3C201                           add     bx, 1C2h        ; bx -> file system id
                                
                                            ; 19/12/2023
                                            ; PCDOS 7.1 IBMBIO.COM
                                            ;mov    word [ld_p_number], 26
                                fmpnext:
                                            ;add    word [ld_p_number], 16
                                            ;cmp    word [ld_p_number], 4122
                                            ;               ; 64 logical disk partitions (64 EBRs)
                                            ;               ; (64*4 = 256 pte's, 256*16 = 4096, + 26 = 4122)
                                            ;jg     short fmpnextfound
```

```
9434
9435 000026F3 26803F05                          cmp     byte [es:bx], 5 ; 05h = extended partition id.
9436 000026F7 7410                              je      short fmpgot ; Extended DOS CHS
9437
9438                                            ; 19/12/2023 - Retro DOS v5.0
9439 000026F9 26803F0F                          cmp     byte [es:bx], 0Fh ; Extended DOS LBA
9440 000026FD 740A                              je      short fmpgot
9441
9442 000026FF 83C310                            add     bx, 16
9443 00002702 81FB0204                          cmp     bx, 402h        ; 202h+bootbias
9444 00002706 75EB                              jnz     short fmpnext
9445                                            ;jmp    short fmpnextfound ; extended partition     not found
9446                                            ; 18/12/2022
9447                                    fmpnextfound:
9448 00002708 C3                                retn
9449
9450                                    ;       ; 30/07/2019 - Retro DOS v3.2
9451                                    ;       jb      short fmpnext
9452                                    ;fmpret:
9453                                    ;       retn    ; 29/05/2019
9454
9455                                    ; ---------------------------------------------------------------------------
9456
9457                                            ; 19/10/2022
9458                                    fmpgot:                                 ; found my partition.
9459 00002709 E82B01                            call    dmax_check      ; check for drvmax already 26
9460 0000270C 73FA                              jnb     short fmpnextfound ; done if too many
9461
9462 0000270E 8B3E[601A]                        mov     di, [end_of_bdss] ; get next free bds
9463
9464                                            ; 19/12/2023
9465                                            ;mov    word [di+47h], 1 ; [di+BDS.bdsm_ismini]
9466                                            ;; 10/12/2022
9467                                            ;or     byte [di+23h], 1
9468                                            ;;or    word [di+23h], 1 ; [di+BDS.flags]
9469                                            ;                       ; fNon_Removable
9470                                            ;mov    byte [di+22h], 5 ; [di+BDS.formfactor]
9471                                            ;                       ; ffHardFile
9472                                            ; 19/12/2023 - Retro DOS v5.0
9473 00002712 C745790100                        mov     word [di+79h], 1 ; [di+BDS.bdsm_ismini]
9474 00002717 804D3F01                          or      byte [di+3Fh], 1 ; [di+BDS.flags], fNon_Removable
9475 0000271B C6453E05                          mov     byte [di+3Eh], 5 ; [di+BDS.formfactor], ffHardFile
9476
9477 0000271F C606[061A]00                      mov     byte [fbigfat], 0 ; assume 12 bit fat.
9478 00002724 A1[621A]                          mov     ax, [mini_hdlim]
9479 00002727 894515                            mov     [di+15h], ax    ; [di+BDS.heads]
9480 0000272A A1[641A]                          mov     ax, [mini_seclim]
9481 0000272D 894513                            mov     [di+13h], ax    ; [di+BDS.secpertrack]
9482 00002730 A0[5C1A]                          mov     al, [rom_minidisk_num]
9483 00002733 884504                            mov     [di+4],al       ; [di+BDS.drivenum]
9484                                                                    ; set physical number
9485 00002736 A0[7500]                          mov     al, [drvmax]
9486 00002739 884505                            mov     [di+5],al       ; [di+BDS.drivelet]
9487                                                                    ; set logical number
9488 0000273C 26837F0A00                        cmp     word [es:bx+10], 0
9489                                            ;ja     short fmpgot_cont
9490 00002741 7707                              ja      short fmpgot1  ; 19/12/2023
9491 00002743 26837F0840                        cmp     word [es:bx+8], 64 ; with current bpb,
9492                                                                    ; only lower word is meaningful.
9493 00002748 72BE                              jb      short fmpnextfound
9494                                                                    ; should be bigger than 64 sectors at least
9495                                    fmpgot1:    ; 19/12/2023
9496                                    ;fmpgot_cont:
9497 0000274A 83EB04                            sub     bx, 4           ; let bx point to the start of the entry
9498 0000274D 268A7702                          mov     dh, [es:bx+2]  ; cylinder
9499 00002751 80E6C0                            and     dh, 0C0h        ; get higher bits of cyl
9500 00002754 D0C6                              rol     dh, 1
9501 00002756 D0C6                              rol     dh, 1
9502 00002758 268A5703                          mov     dl, [es:bx+3]  ; cyl byte
9503                                            ; 19/12/2023 - Retro DOS v5.0
9504 0000275C 89557B                            mov     [di+7Bh], dx    ; [di+BDS.bdsm_hidden_trks]
9505                                            ;mov    [di+49h], dx    ; [di+BDS.bdsm_hidden_trks]
9506                                                                    ; set hidden trks
9507                                            ; 19/12/2023
9508                                            ;push   bx ; * ; PCDOS 7.1
9509                                            ;;;
9510 0000275F 268B4F08                          mov     cx, [es:bx+8]  ; partition size, lw
9511 00002763 268B470A                          mov     ax, [es:bx+10] ; partition size, hw
9512 00002767 030E[0022]                        add     cx, [ep_start_sector]
9513 0000276B 1306[0222]                        adc     ax, [ep_start_sector+2]
9514 0000276F 31D2                              xor     dx, dx ; 19/12/2023
9515 00002771 3916[0022]                        cmp     [ep_start_sector], dx ; 0
9516                                            ;cmp    word [ep_start_sector], 0
9517 00002775 750D                              jnz     short fmpgot2
9518 00002777 3916[0222]                        cmp     [ep_start_sector+2], dx ; 0
9519                                            ;cmp    word [ep_start_sector+2], 0
9520 0000277B 7507                              jnz     short fmpgot2
9521 0000277D 890E[0022]                        mov     [ep_start_sector], cx
9522 00002781 A3[0222]                          mov     [ep_start_sector+2], ax
9523                                    fmpgot2:
9524 00002784 890E[0422]                        mov     [ep_hidden_secs], cx
9525 00002788 A3[0622]                          mov     [ep_hidden_secs+2], ax
9526
9527                                            ; convert start sector address to CHS
9528
9529                                            ; 19/12/2023
9530                                            ; dx = 0
9531                                            ;push   bx ; * ; not necessary
9532
9533                                            ;mov    bx, [di+13h]   ; [di+BDS.secpertrack]
9534 0000278B 8B7513                            mov     si, [di+13h]   ; [di+BDS.secpertrack]
9535                                            ;xor    dx, dx  ; dx = 0
9536                                            ;div    bx
9537 0000278E F7F6                              div     si
9538 00002790 91                                xchg    ax, cx
9539                                            ;div    bx
9540 00002791 F7F6                              div     si
9541                                            ;mov    bx, [di+15h]   ; [di+BDS.heads]
9542                                            ; 07/05/2024
9543                                            ; 17/04/2024 (BugFix)
9544 00002793 8B7515                            mov     si, [di+15h]   ; [di+BDS.heads]
9545 00002796 91                                xchg    ax, cx
9546 00002797 31D2                              xor     dx, dx
9547                                            ;div    bx
9548 00002799 F7F6                              div     si
9549 0000279B 91                                xchg    ax, cx
9550                                            ;div    bx
9551 0000279C F7F6                              div     si
9552
9553                                            ;pop    bx ; *
9554
9555 0000279E 09C9                              or      cx, cx
9556 000027A0 7505                              jnz     short fmpgot_lba_rd
9557 000027A2 3D0004                            cmp     ax, 1024        ; cylinder number < 1024, CHS read is proper
```

```
9558 000027A5 7235                      jb      short fmpgot_chs_rd
9559                                fmpgot_lba_rd:
9560 000027A7 804D4004                  or      byte [di+40h], 4 ; set fLBArw flag ; LBA read/write ok/ready
9561 000027AB 8A16[5C1A]                mov     dl, [rom_minidisk_num]
9562 000027AF 1E                        push    ds
9563                                    ; 19/12/2023
9564                                    ;push   si ; ** ; not necessary
9565 000027B0 31C0                      xor     ax, ax          ; push bp
9566                                                             ; mov bp, sp ; (*)
9567 000027B2 50                        push    ax ; 0
9568 000027B3 50                        push    ax ; 0
9569 000027B4 FF36[0622]                push    word [ep_hidden_secs+2]
9570 000027B8 FF36[0422]                push    word [ep_hidden_secs]
9571 000027BC B80002                    mov     ax, bootbias ; 200h
9572                                    ;mov    ax, 200h        ; bootbias (buffer offset)
9573 000027BF 06                        push    es              ; buffer segment
9574 000027C0 50                        push    ax
9575 000027C1 B80100                    mov     ax, 1
9576 000027C4 50                        push    ax              ; read count
9577 000027C5 B81000                    mov     ax, 10h         ; DAP size = 16
9578 000027C8 50                        push    ax
9579 000027C9 8CD0                      mov     ax, ss
9580 000027CB 8ED8                      mov     ds, ax
9581 000027CD 89E6                      mov     si, sp          ; ds:si = Disk Address Packet
9582
9583 000027CF B442                      mov     ah, 42h         ; LBA read
9584 000027D1 CD13                      int     13h             ; DISK - IBM/MS Extension
9585                                                            ; EXTENDED READ (DL - drive, DS:SI - disk address packet)
9586                                    ; 19/12/2023
9587                                    ;pushf          ; PCDOS 7.1 IBMBIO.COM BUG! Erdogan Tan - 08/08/2023
9588                                    ;add    sp, 16
9589                                    ;popf           ; BUG!
9590                                                            ; mov sp, bp ; (*)
9591                                                            ; pop bp
9592                                    ; 19/12/2023
9593 000027D3 9F                        lahf            ; load status flags into AH
9594 000027D4 83C410                    add     sp, 16
9595 000027D7 9E                        sahf            ; store AH into flags
9596
9597                                    ;pop    si ; ** ; 19/12/2023
9598 000027D8 1F                        pop     ds
9599 000027D9 7317                      jnc     short fmpgot3
9600                                fmpnotfound:        ; 19/12/2023
9601 000027DB C3                        retn
9602                                    ;jmp    short fmpgot3
9603                                    ;;;
9604
9605                                    ; 19/12/2023
9606                                fmpgot_chs_rd:
9607 000027DC 268B4F02                  mov     cx, [es:bx+2]  ; cylinder,cylinder/sector
9608 000027E0 268A7701                  mov     dh, [es:bx+1]  ; head
9609 000027E4 8A16[5C1A]                mov     dl, [rom_minidisk_num]
9610 000027E8 BB0002                    mov     bx, 200h        ; bootbias
9611 000027EB B80102                    mov     ax, 201h
9612 000027EE CD13                      int     13h             ; DISK - READ SECTORS INTO MEMORY
9613                                                            ; AL = number of sectors to read, CH = track, CL = sector
9614                                                            ; DH = head, DL       = drive, ES:BX -> buffer to fill
9615                                                            ; Return: CF set on error, AH =      status, AL = number of sectors read
9616                                ;fmpgot3:   ; 19/12/2023
9617                                    ;jc     short fmpnextfound
9618 000027F0 72E9                      jc      short fmpnotfound
9619                                fmpgot3:
9620 000027F2 BBC203                    mov     bx, 3C2h        ; 1C2h+bootbias
9621
9622                                    ; 08/08/2023 - Retro DOS v4.2 IO.SYS (optimization)
9623                                    ; PCDOS 7.1 IBMBIO.COM - BIOSDATA:2C7Ch
9624 000027F5 26817F3C55AA              cmp     word [es:bx+3Ch], 0AA55h ; 03C2h+03Ch = 3FEh
9625                                    ;jne    short fmpnextfound ; not a valid boot sector !
9626                                    ; 19/12/2023
9627 000027FB 75DE                      jne     short fmpnotfound ; not a valid boot sector !
9628
9629                                    ; 13/08/2023
9630                                    ;push   es
9631 000027FD E80800                    call    setmini         ; install a mini disk.
9632                                                            ; bx value saved.
9633                                    ;pop    es  ; 13/08/2023
9634 00002800 7203                      jc      short fmpnextchain
9635 00002802 E84700                    call    xinstall_bds    ; -- install the bdsm into table
9636                                fmpnextchain:
9637 00002805 E9EBFE                    jmp     fmpnext         ; let's find out
9638                                                            ; if we have any chained partition
9639                                ; ---------------------------------------------------------------------------
9640
9641                                    ; 18/12/2022
9642                                ;fmpnextfound:
9643                                    ;retn
9644
9645                                ; ============== S U B   R O U T I N E =======================================
9646
9647                                ; 15/10/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 IO.SYS)
9648                                ; 28/12/2018 - Retro DOS v4.0 (MSDOS 6.21)
9649
9650                                ; 19/12/2022 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMBIO.COM)
9651                                ; (PCDOS 7.1 IBMBIO.COM - BIOSDATA:2C92h)
9652
9653                                setmini:    ; 'setmini' is called from 'find_mini_partition' procedure
9654
9655 00002808 57                            push    di
9656 00002809 53                            push    bx
9657                                        ; 12/08/2023 - Retro DOS v4.2 IO.SYS (optimization)
9658                                        ; ds = cs = BIOSDATA segment
9659                                        ;push   ds
9660 0000280A 06                            push    es
9661                                setmini_1:
9662                                        ;cmp    byte [es:bx], 1 ; FAT12 partition
9663                                        ;je     short setmini_2
9664                                        ;cmp    byte [es:bx], 4 ; FAT16 (CHS) partition
9665                                        ;je     short setmini_2
9666                                        ;cmp    byte [es:bx], 6 ; FAT16 BIG (CHS) partition
9667                                        ;je     short setmini_2
9668                                        ;
9669                                        ; 19/12/2023 - Retro DOS v5.0
9670                                        ;cmp    byte [es:bx], 0Bh ; FAT32 (CHS) partition
9671                                        ;je     short setmini_2
9672                                        ;cmp    byte [es:bx], 0Ch ; FAT32 (LBA) partition
9673                                        ;je     short setmini_2
9674                                        ;cmp    byte [es:bx], 0Eh ; FAT16 (LBA) partition
9675                                        ;je     short setmini_2
9676
9677                                        ; 19/12/2023
9678 0000280B 268A07                        mov     al, [es:bx]
9679 0000280E 3C01                          cmp     al, 1           ; FAT12 partition
9680 00002810 7422                          je      short setmini_2
9681 00002812 3C04                          cmp     al, 4           ; FAT16 (CHS) partition
```

```
9682 00002814 741E                              je      short setmini_2
9683 00002816 3C06                              cmp     al, 6           ; FAT16 BIG (CHS) partition
9684 00002818 741A                              je      short setmini_2
9685 0000281A 3C0B                              cmp     al, 0Bh         ; FAT32 (CHS) partition
9686 0000281C 7416                              je      short setmini_2
9687 0000281E 3C0C                              cmp     al, 0Ch         ; FAT32 (LBA) partition
9688 00002820 7412                              je      short setmini_2
9689 00002822 3C0E                              cmp     al, 0Eh         ; FAT16 (LBA) partition
9690 00002824 740E                              je      short setmini_2
9691
9692 00002826 83C310                            add     bx, 16
9693 00002829 81FB0204                          cmp     bx, 402h        ; 202h+bootbias
9694                                            ;jne    short setmini_1
9695 0000282D 72DC                              jb      short setmini_1 ; 19/12/2023
9696 0000282F F9                                stc
9697 00002830 07                                pop     es
9698                                            ; 12/08/2023
9699                                            ;pop    ds
9700 00002831 5B                                pop     bx
9701 00002832 5F                                pop     di
9702 00002833 C3                                retn
9703
9704                                    ; ---------------------------------------------------------------------------
9705                                    setmini_2:
9706 00002834 E9D1F9                            jmp     set2            ; branch into middle of sethard
9707
9708                                    ; =============== S U B   R O U TI N E =======================================
9709
9710                                    ; 30/12/2022 - Retro DOS v4.2
9711                                    ; (SYSINITSEG is 473h for MSDOS 6.21 IO.SYS)
9712
9713                                    ; 15/10/2022
9714                                    ; 28/12/2018 - Retro DOS v4.0
9715                                    ;
9716                                    ; dmax_check -- call this when we want to install a new drive.
9717                                    ;               it checks for drvmax < 26 to see if there is
9718                                    ;               a drive letter left.
9719                                    ;
9720                                    ;   drvmax < 26 : carry SET!
9721                                    ;   drvmax >=26 : carry RESET!, error flag set for message later
9722                                    ;                 trash ax
9723
9724                                            ; 19/12/2023 - Retro DOS v5.0
9725                                    dmax_check:
9726 00002837 803E[7500]1A                      cmp     byte [drvmax], 26 ; checks for drvmax < 26
9727 0000283C 720D                              jb      short dmax_ok  ; return with carry if okay
9728 0000283E 06                                push    es
9729                                            ;;mov   ax, 46Dh        ; SYSINIT_SEG (SYSINIT segment)
9730                                            ;mov    ax, 544h        ; 19/12/2023 (PCDOS 7.1)
9731 0000283F B8D904                            mov     ax, SYSINITSEG ; 17/10/2022
9732 00002842 8EC0                              mov     es, ax
9733                                            ; 18/10/2022
9734 00002844 26C606[8803]01                    mov     byte [es:TOOMANYDRIVESFLAG], 1 ; 09/12/2022
9735                                            ;mov    byte ptr es:3FFh, 1 ; [es:toomanydrivesflag]
9736                                                                    ; set message flag
9737                                                                    ; [SYSINIT+toomanydrivesflag]
9738 0000284A 07                                pop     es
9739
9740                                            ;;push  es
9741                                            ;;mov   ax,SYSINIT_SEG
9742                                            ;;mov   es,ax
9743                                            ;;mov   byte [es:toomanydrivesflag],1
9744                                                                    ; set message flag
9745                                            ;;pop   es
9746                                            ;
9747                                            ;mov    byte [SYSINIT+toomanydrivesflag],1
9748                                    dmax_ok:
9749 0000284B C3                                retn
9750
9751                                    ; =============== S U B   R O U TI N E =======================================
9752
9753                                    ; 18/10/2022
9754                                    ; 15/10/2022
9755                                    ; 28/12/2018 - Retro DOS v4.0
9756                                    ;
9757                                    ;   link next bds (at ds:di) into the chain. assume that the
9758                                    ;     chain is entirely within ds == datagrp. also update drvmax,
9759                                    ;     dskdrv_table, and end_of_bdss.
9760
9761                                            ; 19/12/2023 - Retro DOS v5.0
9762                                            ; (PCDOS 7.1 IBMBIO.COM - BIOSDATA:2CE1h)
9763                                    xinstall_bds:
9764 0000284C 56                                push    si
9765 0000284D 53                                push    bx
9766 0000284E 8B36[1901]                        mov     si, [start_bds]       ; get first bds
9767                                    xinstall_bds_1:
9768 00002852 833CFF                            cmp     word [si], 0FFFFh ; is this the last one?
9769 00002855 7404                              jz      short xinstall_bds_2 ;skip ahead if so
9770                                            ;mov    si, [si+BDS.link]
9771 00002857 8B34                              mov     si, [si]        ; chain through list
9772 00002859 EBF7                              jmp     short xinstall_bds_1
9773
9774                                    xinstall_bds_2:
9775                                            ;mov    [si+BDS.link], di
9776 0000285B 893C                              mov     [si], di
9777                                            ;mov    [si+BDS.link+2], ds
9778 0000285D 8C5C02                            mov     [si+2], ds
9779                                            ;mov    word [di+BDS.link], -1
9780 00002860 C705FFFF                          mov     word [di], 0FFFFh ; make sure it is a null ptr.
9781                                            ;mov    [di+BDS.link+2], ds
9782 00002864 8C5D02                            mov     [di+2], ds ; might as well plug segment
9783                                            ; 20/03/2019 - Retro DOS v4.0
9784                                            ;lea    bx, [di+BDS.BPB]
9785 00002867 8D5D06                            lea     bx, [di+6]
9786 0000286A 8B36[5E1A]                        mov     si, [last_dskdrv_table]
9787 0000286E 891C                              mov     [si], bx
9788 00002870 8306[5E1A]02                      add     word [last_dskdrv_table], 2
9789 00002875 FE06[7500]                        inc     byte [drvmax]
9790                                            ;add    word [end_of_bdss], 100 ; BDS.size = 100
9791                                            ; 19/12/2023 - Retro DOS v5.0
9792 00002879 8106[601A]9600                    add     word [end_of_bdss], 150 ; BDS.size = 150
9793 0000287F 5B                                pop     bx
9794 00002880 5E                                pop     si
9795 00002881 C3                                retn
9796
9797                                    ; =============== S U B   R O U TI N E =======================================
9798
9799                                    ; 17/10/2022
9800                                    ; 15/10/2022
9801                                    ; 28/12/2018 - Retro DOS v4.0
9802                                    ; 03/06/2018 - Retro DOS v3.0
9803
9804                                            ; 19/12/2023 - Retro DOS v5.0
9805                                    cmos_clock_read:
```

```
9806 00002882 50                        push    ax
9807 00002883 51                        push    cx
9808 00002884 52                        push    dx
9809 00002885 55                        push    bp
9810 00002886 31ED                      xor     bp, bp
9811                            loop_clock:
9812 00002888 31C9                      xor     cx, cx
9813 0000288A 31D2                      xor     dx, dx
9814 0000288C B402                      mov     ah, 2
9815 0000288E CD1A                      int     1Ah             ; CLOCK - READ REAL TIME CLOCK (AT,XT286,CONV,PS)
9816                                                            ; Return: CH = hours in      BCD
9817                                                            ; CL = minutes in BCD
9818                                                            ; DH = seconds in BCD
9819                                    ; 19/12/2023
9820                                    ;cmp    cx, 0
9821 00002890 21C9                      and     cx, cx
9822 00002892 750F                      jnz     short clock_present
9823                                    ;cmp    dx, 0
9824 00002894 09D2                      or      dx, dx
9825 00002896 750B                      jnz     short clock_present
9826                                    ;cmp    bp, 1           ; read again after a slight delay, in case clock
9827                                    ;je     short no_readdate ; was      at zero setting.
9828 00002898 21ED                      and     bp, bp
9829 0000289A 751A                      jnz     short no_readdate
9830 0000289C 45                        inc     bp              ; only perform delay once.
9831                                    ;mov    cx, 4000h       ; 16384
9832                                    ; 19/12/2023
9833 0000289D B540                      mov     ch, 40h ; cx = 4000h ; 16384
9834                            delay:
9835 0000289F E2FE                      loop    delay
9836 000028A1 EBE5                      jmp     short loop_clock
9837                            ; ---------------------------------------------------------------------------
9838
9839                            clock_present:
9840                                    ;mov    byte [cs:havecmosclock], 1 ; set the flag for cmos clock
9841                                    ; 19/12/2023
9842                                    ; ds = cs
9843 000028A3 C606[8C04]01             mov     byte [havecmosclock], 1 ; set the flag for cmos clock
9844
9845 000028A8 E81000                    call    cmosck          ; reset cmos clock rate       that may be
9846                                                            ; possibly destroyed by       cp dos and
9847                                                            ; post routine did not restore that.
9848 000028AB 56                        push    si
9849 000028AC E8BCEF                    call    read_real_date ; read real-time clock for date
9850 000028AF FA                        cli
9851                                    ;mov    ds:daycnt, si  ; set system date
9852 000028B0 8936[8904]                mov     [daycnt], si
9853 000028B4 FB                        sti
9854 000028B5 5E                        pop     si
9855                            no_readdate:
9856 000028B6 5D                        pop     bp
9857 000028B7 5A                        pop     dx
9858 000028B8 59                        pop     cx
9859 000028B9 58                        pop     ax
9860
9861                            cmosck9:    ; 19/12/2023
9862 000028BA C3                        retn
9863
9864                            ; ---------------------------------------------------------------------------
9865
9866                            ; the following code is written by jack gulley in engineering group.
9867                            ; cp dos (CP/DOS, OS/2) is changing cmos clock rate for its own purposes
9868                            ; and if the use cold boot the system to use pc dos while running cp dos,
9869                            ; the cmos clock rate are still slow which slow down disk operations
9870                            ; of pc dos which uses cmos clock. pc dos is put this code in msinit
9871                            ; to fix this problem at the request of cp dos.
9872                            ;
9873                            ; the program is modified to be run on msinit. equates are defined
9874                            ; in cmosequ.inc. this program will be called by cmos_clock_read procedure.
9875                            ;
9876                            ;  the following code cmosck is used to insure that the cmos has not
9877                            ;   had its rate controls left in an invalid state on older at's.
9878                            ;
9879                            ;   it checks for an at model byte "fc" with a submodel type of
9880                            ;   00, 01, 02, 03 or 06 and resets the periodic interrupt rate
9881                            ;   bits in case post has not done it. this initilization routine
9882                            ;   is only needed once when dos loads. it should be run as soon
9883                            ;   as possible to prevent slow diskette access.
9884                            ;
9885                            ;   this code exposes one to dos clearing cmos setup done by a
9886                            ;   resident program that hides and re-boots the system.
9887
9888                            cmosck:                                 ; check and reset rtc rate bits
9889
9890                            ;model byte and submodel byte were already determined in msinit.
9891
9892                                ; 16/06/2018 - Retro DOS v3.0
9893                                ; 19/03/2018 (Model: 0FCh, Sub Model: 01h, REF: AMIBIOS Prog. Guide)
9894
9895                                ; 19/12/2023 - Retro DOS v5.0
9896
9897                                        ; 19/12/2023
9898                                        ; ds = cs
9899                                        ;push   ax ; not necessary ; 19/12/2023
9900                                        ;
9901 000028BB 803E[AF05]FC                 cmp     byte [model_byte], 0FCh
9902                                        ;cmp    byte [cs:model_byte], 0FCh
9903 000028C0 75F8                         jnz     short cmosck9  ; Exit if not an AT model
9904 000028C2 803E[B005]06                 cmp     byte [secondary_model_byte], 6 ; 21/04/2024
9905                                        ;cmp    byte [cs:secondary_model_byte], 6
9906                                                                ; Is it 06 for the industral AT ?
9907 000028C7 7407                         jz      short cmosck4  ; Go reset CMOS periodic rate if 06
9908 000028C9 803E[B005]04                 cmp     byte [secondary_model_byte], 4
9909                                        ;cmp    byte [cs:secondary_model_byte], 4
9910                                                                ; Is it 00, 01, 02, or 03 ?
9911 000028CE 73EA                         jnb     short cmosck9  ; EXIT if problem fixed by POST
9912                                                                ; Also,Secondary_model_byte = 0
9913                                                                ;   when AH=0C0h, int 15h failed.
9914                                                                ;       RESET THE CMOS PERIODIC RATE
9915                                                                ;  Model=FC submodel=00,01,02,03 or 06
9916                                    cmosck4:
9917 000028D0 B08A                         mov     al, 8Ah        ; cmos_reg_a|nmi
9918                                                               ; NMI disabled on return
9919 000028D2 B426                         mov     ah, 26h        ; 00100110b
9920                                                               ; Set divider & rate selection
9921 000028D4 E80B00                       call    cmos_write
9922 000028D7 B08B                         mov     al, 8Bh        ; cmos_reg_b|nmi
9923                                                               ; NMI disabled on return
9924 000028D9 E82000                       call    cmos_read
9925 000028DC 2407                         and     al, 7          ; 00000111b
9926                                                               ; clear SET,PIE,AIE,UIE,SQWE
9927 000028DE 88C4                         mov     ah, al
9928 000028E0 B00B                         mov     al, 0Bh        ; cmos_reg_b
9929                                                               ; NMI enabled on return
```

```
                                ; 19/12/2023
                                ;call   cmos_write
                        ;cmosck9:
                                ;pop    ax ; 19/12/2023
                                ;retn

                                ; 19/12/2023
                                ;jmp    short cmos_write

                        ; =============== S U B   R O U T I N E =======================================

                        ;--- cmos_write -----------------------------------------------------------------
                        ;           write byte to cmos system clock configuration table       :
                        ;                                                                     :
                        ; input: (al)=     cmos table address to be written to                 :
                        ;           bit    7 = 0 for nmi enabled and 1 for nmi disabled on exit    :
                        ;           bits 6-0 = address of table location to write                 :
                        ;    (ah)= new value to be placed in the addressed table location       :
                        ;                                                                     :
                        ; output:   value in (ah) placed in location (al) with nmi left disabled   :
                        ;           if bit 7 of (al) is on. during the cmos update both nmi and    :
                        ;           normal interrupts are disabled to protect cmos data integrity. :
                        ;           the cmos address register is pointed to a default value and    :
                        ;           the interrupt flag restored to the entry state on return.      :
                        ;           only the cmos location and the nmi state is changed.             :
                        ;-------------------------------------------------------------------------------

                        cmos_write:                     ; write (ah) to location (al)
 000028E2 9C                    pushf                   ;
 000028E3 50                    push    ax              ; save work register values
 000028E4 FA                    cli
 000028E5 50                    push    ax              ; save user nmi       state
 000028E6 0C80                  or      al, 80h         ; disable nmi for us
 000028E8 E670                  out     70h, al         ; CMOS Memory/RTC Index      Register:
                                                        ; RTC Seconds
 000028EA 90                    nop
 000028EB 88E0                  mov     al, ah
 000028ED E671                  out     71h, al         ; CMOS Memory/RTC Data Register
 000028EF 58                    pop     ax              ; get user nmi
 000028F0 2480                  and     al, 80h
 000028F2 0C0F                  or      al, 0Fh
 000028F4 E670                  out     70h, al         ; CMOS Memory/RTC Index      Register:
                                                        ; RTC Seconds
 000028F6 90                    nop
 000028F7 E471                  in      al, 71h         ; CMOS Memory/RTC Data Register
 000028F9 58                    pop     ax              ; restore work registers

                                ; 19/12/2023
                                ;push   cs              ; *place code segment in stack and
                                ;call   cmos_popf       ; *handle popf for b- level 80286
                                ;retn
 000028FA EB18                  jmp     short cmos_rw_popf

                        ; =============== S U B   R O U T I N E =======================================

                        ;--- CMOS_READ ------------------------------------------------------------------
                        ;           read byte from cmos system clock configuration table             :
                        ;                                                                         :
                        ; input: (al)=     cmos table address to be read                           :
                        ;           bit    7 = 0 for nmi enabled and 1 for nmi disabled on exit    :
                        ;           bits 6-0 = address of table location to read                  :
                        ;                                                                         :
                        ; output: (al)     value at location (al) moved into (al). if bit 7 of (al) was   :
                        ;           on then nmi left disabled. during the cmos read both nmi and  :
                        ;           normal interrupts are disabled to protect cmos data integrity. :
                        ;           the cmos address register is pointed to a default value and    :
                        ;           the interrupt flag restored to the entry state on return.      :
                        ;           only the (al) register and the nmi state is changed.             :
                        ;-------------------------------------------------------------------------------

                        cmos_read:                      ; read location (al) into (al)
 000028FC 9C                    pushf
 000028FD FA                    cli
 000028FE 53                    push    bx
                                ;push   ax ; *          ; AL = cmos table address to be read
                                ; 19/12/2023
 000028FF 89C3                  mov     bx, ax ; * ; input
 00002901 0C80                  or      al, 80h
 00002903 E670                  out     70h, al         ; CMOS Memory/RTC Index      Register:
                                                        ; RTC Seconds
 00002905 90                    nop                     ; (undocumented delay needed)
 00002906 E471                  in      al, 71h         ; CMOS Memory/RTC Data Register

                                ;mov    bx, ax ; output
                                ;pop    ax ; * ; input

                                ; 19/12/2023
                                ; al = output, bl = input
 00002908 93                    xchg    ax, bx ; *
                                ; bl = output, al = input

 00002909 2480                  and     al, 80h
 0000290B 0C0F                  or      al, 0Fh
 0000290D E670                  out     70h, al         ; CMOS Memory/RTC Index      Register:
                                                        ; RTC Seconds
 0000290F 90                    nop
 00002910 E471                  in      al, 71h         ; CMOS Memory/RTC Data Register
                                ;mov    ax, bx ; * ; output
                                ; 19/12/2023
 00002912 93                    xchg    ax, bx
 00002913 5B                    pop     bx

                                ; 19/12/2023
                        cmos_rw_popf:
 00002914 0E                    push    cs              ; *place code segment in stack and
 00002915 E80100                call    cmos_popf       ; *handle popf for b- level 80286
 00002918 C3                    retn                    ; return with flags restored

                        ; -------------------------------------------------------------------------------

                        cmos_popf:
 00002919 CF                    iret                    ; popf for level b- parts
                                                        ; return far and restore flags

                        ; 21/12/2022
                        ; -------------------------------------------------------------------------------
                        ; -------------------------------------------------------------------------------
                        %if 0

                        ; -------------------------------------------------------------------------------
                        ; MSINIT.ASM (MSDOS 6.0, 1991)
                        ; -------------------------------------------------------------------------------
                        ; The following routines provide support for reading in the file MSDOS.SYS.
                        ; -------------------------------------------------------------------------------
```

```
10054
10055                           ; 15/10/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 IO.SYS)
10056                           ;
10057                           ; (For Retro DOS, 'IO.SYS' and 'MSDOS.SYS' are already loaded together
10058                           ;  at once -as single kernel file- by the Retro DOS boot sector code.
10059                           ;  So, following disk reads -MSDOS.SYS loading- is not needed!
10060                           ;  Only needing is to move MSDOS Kernel to it's final memory location.)
10061
10062                           ; ============== S U B   R O U T I N E =======================================
10063
10064                           ; GetClus, read in a cluster at a specified address
10065                           ;
10066                           ;  bx = cluster to read
10067                           ;  cx = sectors per cluster
10068                           ;  es:di = load location
10069
10070                           ; 17/10/2022
10071                           ;DISKRD equ diskrd - DOSBIOSEG_2C7h      ; (8E5h for MSDOS 5.0 IO.SYS)
10072                           ; 09/12/2022
10073                           DISKRD equ diskrd
10074
10075                                   ; 29/12/2023
10076                                   ; 20/12/2023 - Retro DOS v5.0
10077                                   ; (PCDOS 7.1 IBMBIO.COM - BIOSDATA:2DC4h)
10078
10079                                   ; si:bx = (32 bit) cluster to read
10080                                   ; cx = sectors per cluster
10081                                   ; es:di = load location
10082
10083                                   ; 17/10/2022
10084                           getclus:
10085                                   ; 12/12/2023
10086                                   ; ds = cs
10087
10088                                   push    cx ; 1*
10089                                   push    di ; 2*
10090                                   ;mov    [cs:doscnt], cx
10091                                   mov     [doscnt], cx ; 12/12/2023
10092
10093                                   ; 20/12/2023
10094                                   ;;mov   [cs:ClusterH], si ; high word of cluster number
10095                                   ;mov    [ClusterH], si ; high word of cluster number
10096                                   mov     bp, si
10097
10098                                   mov     ax, bx
10099
10100                                   ;dec    ax
10101                                   ;dec    ax
10102                                   ; 20/12/2023
10103                                   sub     ax, 2
10104
10105                                   ;;sbb   [cs:ClusterH], 0
10106                                   ;sbb    [ClusterH], 0
10107                                   sbb     bp, 0
10108
10109                                   ; 20/12/2023
10110                                   ;;xchg  ax, [cs:ClusterH]
10111                                   ;xchg   ax, [ClusterH]
10112                                   xchg    ax, bp
10113
10114                                   mul     cx
10115
10116                                   ;;xchg  ax, [cs:ClusterH]
10117                                   ;xchg   ax, [ClusterH]
10118                                   xchg    ax, bp ; (+)
10119                                   ;
10120                                   mul     cx              ;; convert to logical sector
10121                                                           ;; dx:ax = matching logical sector number
10122                                                           ;;       starting from the data sector
10123                                   ;;add   ax, [cs:bios_l]
10124                                   ;;adc   dx, [cs:bios_h]        ; dx:ax = first logical sector to read
10125                                   ; 12/12/2023
10126                                   ;add    ax, [bios_l]
10127                                   ;adc    dx, [bios_h]   ; dx:ax = first logical sector to read
10128
10129                                   ; 20/12/2023
10130                                   ;;add   dx, [cs:ClusterH]
10131                                   ;add    ax, [cs:First_Data_Sector]
10132                                   ;adc    dx, [cs:First_Data_Sector+2]
10133                                   add     dx, bp ; (+)
10134                                   ;add    dx, [ClusterH] ; convert to logical sector
10135                                                           ; dx:ax = matching logical sector number
10136                                                           ;       starting from the data sector
10137                                   add     ax, [First_Data_Sector]
10138                                   adc     dx, [First_Data_Sector+2]
10139                                                           ; dx:ax = first logical sector to read
10140                           unpack:
10141                                   ; 20/12/2023
10142                                   push    ds ; 3* ; ds = cs ; 12/12/2023
10143                                   push    dx ; 4* ; * ; 12/12/2023
10144                                   push    ax ; 5*
10145                                   ; 29/12/2023
10146                                   push    si ; 6*
10147                                   push    bx ; 7*
10148
10149                                   ;;mov   si, [cs:fatloc]
10150                                   ;mov    si, [fatloc] ; 12/12/2023
10151                                   ;mov    ds, si
10152                                   ; 20/12/2023
10153                                   ;mov    ax, [fatloc]
10154                                   ;mov    ds, ax
10155                                   push    bx ; 8*
10156                                   push    word [fatloc] ; 9*
10157
10158                                   ;test   byte [cs:fbigfat], 20h
10159                                   test    byte [fbigfat], 20h   ; fbigbig FAT32 ?
10160                                   pop     ds ; 9* ; ds = [fatloc]
10161                                   jz      short not_32bit_cluster ; no
10162                           unpack32:
10163                                   ;push   dx
10164                                   mov     dx, si
10165                                   ;mov    si, bx
10166                                   pop     si ; 8* ; si = bx
10167                                   add     si, si
10168                                   adc     dx, dx
10169                                   add     si, si
10170                                   adc     dx, dx
10171                                                   ; dx:si = 4*(si:bx) ; clust num offset from FAT entry 0
10172                                   call    get_fat_sector
10173                                   mov     si, [bx+2]      ; high word of the FAT32 cluster number
10174                                   mov     bx, [bx]        ; low word of the FAT32 cluster number
10175                                   ;pop    dx
10176                                   jmp     short getcl1
10177
```

```
                        not_32bit_cluster:
                                    ;mov    si, bx          ; next cluster
                                    pop     si ; 8* ; si = bx
                                    test    byte [cs:fbigfat], 40h; fbig
                                                            ; 16 bit fat?
                                    jnz     short unpack16 ; yes
                        unpack12:
                                    shr     si, 1           ; 12 bit fat. si = si/2
                                                            ; si = clus + clus/2
                                    add     si, bx          ;
                                                            ; (si = byte offset of the cluster in the FAT)
                                    ;push   dx ; 12/12/2023
                                    xor     dx, dx
                                    ; 12/12/2023
                                    ; ds = FAT buffer segment
                                    call    get_fat_sector
                                    ;pop    dx ; 12/12/2023

                                    mov     ax, [bx]        ; save it into ax
                                    jnz     short even_odd ; if not a splitted fat, check even-odd.
                                    ; 25/06/2023
                                    ;mov    al, [bx]        ; splitted fat

                                    ; 12/12/2023
                                    ;mov    [cs:temp_cluster], al
                                    push    ax ; ** ; al = low 8 bits of 12 bits cluster number

                                    inc     si              ; (next byte)

                                    ;push   dx ; 12/12/2023
                                    xor     dx, dx
                                    call    get_fat_sector
                                    ;pop    dx ; 12/12/2023

                                    ;mov    al, ds:0
                                    ; 12/12/2023
                                    ; ds = FAT buffer segment
                                    ;mov    al, [0] ; 19/10/2022
                                    ;mov    [cs:temp_cluster+1], al
                                    ;mov    ax, [cs:temp_cluster]
                                    ; 12/12/2023
                                    ;mov    al, [cs:temp_cluster]
                                    pop     ax  ; ** ; al = low 8 bits of 12 bits cluster number
                                    mov     ah, [0] ; high 4 bits (bits 7 to 11) of 12 bits cluster num
                        even_odd:
                                    ; 29/12/2023
                                    pop     bx ; 7*         ; restore old fat entry       value
                                    push    bx              ; save it right       away.
                                    shr     bx, 1           ; was it even or odd?
                                    jnc     short havclus  ; it was even.
                                    shr     ax, 1           ; odd. massage fat value and keep
                                                            ; the highest 12 bits.
                                    shr     ax, 1
                                    shr     ax, 1
                                    shr     ax, 1
                        havclus:
                                    mov     bx, ax          ; now bx = new fat entry.
                                    and     bx, 0FFFh       ; keep low 12 bits.
                                    jmp     short unpackx
                        ; ---------------------------------------------------------------------------

                        unpack16:
                                    ;push   dx      ; 12/12/2023
                                    xor     dx, dx ; 0
                                    shl     si, 1           ; extend to 32 bit offset
                                    ;adc    dx, 0
                                    ; 12/12/2023
                                    rcl     dx, 1

                                    ; 12/12/2023
                                    ; ds = FAT buffer segment
                                    call    get_fat_sector
                                    ;pop    dx      ; 12/12/2023
                                    mov     bx, [bx]        ;
                                                            ; bx = new fat entry.
                        unpackx:
                                    ; 20/12/2023
                                    xor     si, si          ; high word of cluster number = 0
                                                            ; (FAT12 or FAT16)
                        getcl1:
                                    ; 29/12/2023
                                    pop     ax      ; 7* - cluster number lw
                                    ;pop    word [cs:ClusterH]
                                    pop     dx      ; 6* - cluster number hw

                                    ; 20/12/2023 - Retro DOS v5.0 (PCDOS 7.1 IBMBIO.COM)
                                    ; (this is a fast kernel loading method by the MSDOS programmer)
                                    ; ((consequtive clusters --> consequtive sectors))

                                    sub     ax, bx  ; previous - current (or current - new)
                                    ;sbb    [cs:ClusterH], si
                                    sbb     dx, si
                                    ;;cmp   [cs:ClusterH], -1 ; one apart? (current = previous+1)
                                    ;cmp    dx, -1
                                    ; 29/12/2023
                                    inc     dx ; -1 -> 0
                                    jnz     short not_consequental
                                    ;cmp    ax, -1          ; 0FFFFh ; is [ClusterH]:ax = -1 ?
                                    inc     ax ; -1 -> 0
                        not_consequental:
                                    pop     ax ; 5*         ; restore logical sector (low)
                                    pop     dx ; 4* ; * ; 12/12/2023
                                    pop     ds ; 3*

                                    ;; 12/12/2023
                                    ;; (this is a fast kernel loading method by the MSDOS programmer)
                                    ;; ((consequtive clusters --> consequtive sectors))
                                    ;; ds = cs
                                    ;sub    si, bx
                                    ;cmp    si, -1          ; one apart? (consequtive?)
                                    ;                       ; (current = previous+1)

                                    jnz     short getcl2   ; no, read [doscnt] sectors

                                    ;add    [cs:doscnt], cx ; (cx = sectors per cluster)
                                    add     [doscnt], cx ; 12/12/2023 ; add to read count
                                    jmp     unpack
                        ; ---------------------------------------------------------------------------

                        getcl2:
                                    push    si ; 20/12/2023
                                    push    bx
                                    ; bx = low word of the new cluster number
                                    ; 20/12/2023 - Retro DOS v5.0 (32 bit cluster numbers)
```

```
                        ; si = high word of the new cluster number
                        push    dx              ; sector to read (high word)
                        push    ax              ; sector to read (low word)

                        ; 12/12/2023
                        ; ds = cs
                        ;mov    ax, [cs:drvfat]        ; get drive and       fat spec
                        ;mov    cx, [cs:doscnt]
                        mov     ax, [drvfat]    ; get drive and       fat spec

                        ;;;
                        ; 20/12/2023 - Retro DOS v5.0 (PCDOS 7.1 IBMBIO.COM)
                        ;
                        ; dma and segment (64K boundary) overrun precaution
                        ; (sector count will be decreased if it is required)
                        mov     cx, di
                        not     cx              ; cx = 65535 - cx
                        shr     cx, 1           ; cx = cx/2
                        xor     cl, cl
                        xchg    cl, ch          ; cx = cx/256

                        ;cmp    cx, [cs:doscnt]
                                                ; if sector read count > cx, decrease it to cx
                        cmp     cx, [doscnt]
                        jbe     short getcl3
                        ;;;
                        ;mov    cx, [cs:doscnt]
                        mov     cx, [doscnt]
                getcl3:
                        pop     dx              ; sector to read for diskrd (low)
                        ;pop    word [cs:start_sec_h]
                        ; 12/12/2023
                        pop     word [start_sec_h]
                                                ; sector to read for diskrd (high)
                        ; 12/12/2023
                        ; ds = cs
                        ;push   ds
                        ;push   cs
                        ;pop    ds

                        push    cs              ; simulate far call

                        ; 20/12/2023
                        ; 17/10/2022
                        mov     bp, DISKRD      ; offset diskrd
                        ;mov    bp, 0A2Bh       ; 20/12/2023
                        ;       (PCDOS 7.1 IBMBIO.COM - BIOSCODE:0A2Bh ; 364h:0A2Bh)
                        ;mov    bp, 8E5h        ; 17/10/2022
                                                ; 2C7h:8E5h = 70h:2E55h

                        call    call_bios_code ; read the clusters

                        ;pop    ds
                        ; 12/12/2023
                        ; ds = cs
                        pop     bx              ; lw of the new cluster number
                        pop     si ; 20/12/2023 ; hw of the new cluster number

                        pop     di ; 2* - (kernel) load location (es:di)

                        ;mov    ax, [cs:doscnt]        ; get number of       sectors read
                        ; 12/12/2023
                        mov     ax, [doscnt]
                        xchg    ah, al          ; multiply by 256
                        shl     ax, 1           ; times 2 equal 512
                        add     di, ax          ; update load location

                        pop     cx ; 1*         ; restore sectors/cluster

                        retn

                ; ============== S U B   R O U T I N E =======================================

                ;function: find and read the corresponding fat sector into ds:0
                ;
                ;in). dx:si - offset value (starting from fat entry 0) of fat entry to find. M054
                ;     ds - fatloc segment
                ;     cs:drvfat - logical drive number, fat id
                ;     cs:md_sectorsize
                ;     cs:last_fat_secnum - last fat sector number read in.
                ;
                ;out). corresponding fat sector read in.
                ;      bx = offset value from fatlog segment.
                ;      other registers are saved.
                ;      zero flag set if the fat entry is splitted, i.e., when 12 bit fat entry
                ;      starts at the last byte of the fat sector. in this case, the caller
                ;      should save this byte, and read the next fat sector to get the rest
                ;      of the fat entry value. (this will only happen with the 12 bit fat.)

                        ; 17/10/2022
                get_fat_sector:
                        ; 12/12/2023
                        ; ds = fat buffer segment

                        ; 12/12/2023
                        ;push   ax ; (not necessary)
                        push    cx ; read count (sectors per cluster)
                        push    di ; IBMDOS.COM/MSDOS.SYS load offset
                        push    si ; FAT offset value (from fat entry 0)
                        push    es ; IBMDOS.COM/MSDOS.SYS load segment
                        push    ds ; FAT buffer segment

                        ; 12/12/2023
                        push    cs
                        pop     ds

                        mov     ax, si
                        ;;mov   cx, [cs:md_sectorsize] ; 512
                        ; 12/12/2023
                        ;mov    cx, [md_sectorsize] ; 512
                        ;div    cx              ; ax = sector number, dx = offset
                        ; 12/12/2023
                        ;nop

                        ; 12/12/2023
                        div     word [md_sectorsize] ; 512

                        ; ax = FAT sector (sequence/index) number
                        ; dx = cluster number offset

                        ; Get rid of the assumption that
                        ; there is only one reserved sector

                        ; 12/12/2023 ; *
```

```
                        ;push   es ; *
                        ;push   ds ; *
                        ;push   di ; *
                        push    ax
                        ;push   cs ; *
                        ;pop    ds ; *

                        ;mov    ax, [cs:drvfat]       ; get drive # and FAT id
                        ; 12/12/2023
                        mov     ax, [drvfat]   ; get drive # and FAT id
                        mov     bp, SETDRIVE
                        ;mov    bp, 5AEh  ; PCDOS 7.1 IBMBIO.COM
                        ;;mov   bp, 4D7h       ; setdrive
                                               ; at 2C7h:4D7h = 70h:2A47h
                        push    cs             ; simulate far call
                        call    call_bios_code ; get bds for drive
                        pop     ax             ; (sector number -without reserved and hidden sectors-)
                        add     ax, [es:di+9]  ; [es:di+BDS.resectors]
                                               ; add #reserved_sectors
                        ; 12/12/2023
                        ;pop    di ; *
                        ;pop    ds ; *
                        ;pop    es ; *

                        ; 12/12/2023
                        ; ds = cs
                        cmp     ax, [last_fat_sec_num]
                        ;cmp    ax, [cs:last_fat_sec_num]
                        jz      short gfs_split_chk ; don't need to read it again.
                        mov     [last_fat_sec_num], ax
                        ;mov    [cs:last_fat_sec_num], ax
                                               ; sector number
                                               ; (in the partition, without hidden sectors)
                        ; 13/12/2023
                        pop     es ; FAT buffer segment (DS on top of the stack)
                        push    es ; (put it on top of the stack again)

                        push    dx ; cluster number offset

                                               ; 12/12/2023
                                               xor     cx, cx
                                               mov     [start_sec_h], cx ;0
                                               ;mov    word [cs:start_sec_h], 0
                                                                       ; prepare to read the fat sector
                                                                       ; start_sec_h is always       0 for fat sector.
                                               mov     dx, ax
                                               ; 12/12/2023
                                               inc     cx ; cx = 1
                                               ;mov    cx, 1          ; 1 sector read
                                               ;mov    ax, [cs:drvfat]
                                               mov     ax, [drvfat]
                                               ;push   ds
                                               ;pop    es

                                               xor     di, di  ; 0     ; es:di -> fatloc segment:0

                                               ; 12/12/2023
                                               ;push   ds
                                               ;push   cs
                                               ;pop    ds

                                               push    cs             ; simulate far call
                                               mov     bp, DISKRD     ; 8E5h
                                               ;mov    bp, 8E5h       ; offset diskrd
                                                                      ; 2C7h:8E5h = 70h:2E55h
                                               call    call_bios_code

                                               ; 12/12/2023
                                               ;pop    ds
                                               ; ds = cs = biosdata segment

                                               pop     dx ; cluster number offset

                                       gfs_split_chk:
                                               ; 13/12/2023
                                               ;mov    cx, [cs:md_sectorsize] ; 512
                                               mov     cx, [md_sectorsize]
                                       ;gfs_split_chk:
                                               dec     cx             ; 511
                                               cmp     dx, cx         ; if offset points to the
                                                                      ; last byte of this sector,
                                                                      ; then splitted       entry.
                                               mov     bx, dx         ; set bx to dx

                                               ; 12/12/2023
                                               ; bx = dx = cluster number offset in the FAT buffer
                                               pop     ds ; FAT buffer segment
                                               pop     es ; IBMDOS.COM/MSDOS.SYS load segment
                                               pop     si ; FAT offset value (from fat entry 0)
                                               pop     di ; IBMDOS.COM/MSDOS.SYS load offset
                                               pop     cx ; read count (sectors per cluster)
                                               ;pop    ax

                                               retn
                                       ; 15/10/2022
                                       ;Bios_Data_Init    ends

                                       %endif
                                       ; ----------------------------------------------------------------------------
                                       ; ----------------------------------------------------------------------------

                                               ; 09/12/2022
                                               ;db 0

                                       numbertodiv equ ($-BData_start)
                                       numbertomod equ (numbertodiv % 16)

                                       %if (numbertomod>0) & (numbertomod<16) ; 17/09/2023
0000291A 00<rep 6h>                            times (16-numbertomod) db 0
                                       %endif

                                       ;align 16

                                       ; 09/12/2022
                                       IOSYSCODESEGOFF equ $ - BData_start
                                       ; 29/09/2023
                                       ;IOSYSCODESEGOFF equ $-$$
                                       IOSYSCODESEG       equ (IOSYSCODESEGOFF>>4)+(700h>>4)

                                       ; 28/09/2023
                                       S1SIZE equ $-$$

                                       ;--- End of DOSBIOS data segment ----------------------------------------------
                                       ; ----------------------------------------------------------------------------
```

```
                                            ;db 4 dup(0)
                                    ; 09/12/2022
                                    ;           times 4 db 0   ; 19/10/2022
                                    ; ----------------------------------------------------------------------------

                                    ;============================================================================
                                    ; DOS BIOS (IO.SYS) CODE SEGMENT
                                    ;============================================================================
                                    ; 09/12/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 IO.SYS)
                                    ; 30/12/2022 - Retro DOS v4.2 (Modified MSDOS 6.21 IO.SYS)

                                    section .BIOSCODE vstart=0

                                    ; 30/12/2022
                                    ; (BIOSCODE SEGMENT is 2CCh for MSDOS 6.21 IO.SYS)  -- ((25C0h+700h)>>4) --

                                    BCode_start:        ; 09/12/2022

                                    ; 02/10/2022

                                    ;--- DOSBIOS code segment ---------------------------------------------------
                                    ;----------------------------------------------------------------------------
                                    ; MSBIO1.ASM (MSDOS 6.0, 1991)
                                    ;----------------------------------------------------------------------------

                                    DOSBIOSEG_2C7h:    ;db 30h dup(0)        ; SEGMENT 2C7h (2C70h-700h=2570h)
00000000 00<rep 30h>                            times 48 db 0          ; 19/10/2022
00000030 7000                       BiosDataWord:      dw 70h

                                    ; 15/10/2022
                                    ;BIOSDATAWORD      equ BiosDataWord - DOSBIOSEG_2C7h
                                    ; 09/12/2022
                                    BIOSDATAWORD       equ BiosDataWord

                                    ; ----------------------------------------------------------------------------

                                    ; 15/10/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 IO.SYS)
                                    ; 20/12/2023 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMBIO.COM)

                                    ;****************************************************************************
                                    ;*                                                                          *
                                    ;*  seg_reinit is called with ax = our new code segment value,              *
                                    ;*    trashes di, cx, es                                                    *
                                    ;*                                                                          *
                                    ;*  cas -- should be made disposable!                                       *
                                    ;*                                                                          *
                                    ;****************************************************************************

                                        ; 20/09/2023
                                        ; 10/08/2023 - Retro DOS v4.2 IO.SYS (optimization)
                                        ; PCDOS 7.1 IBMBIO.COM - BIOSCODE:0032h

                                    _seg_reinit:
00000032 2E8E06[3000]                           mov    es, [cs:BIOSDATAWORD]
                                                                       ; at 2C7h:30h or 70h:25A0h
                                                ;mov   di, (offset cdev+2)
00000037 BF[0406]                               mov    di, cdev+2      ; 19/10/2022
                                                ;mov   cx, 4           ; (end_BC_entries - cdev)/4
                                                ; 10/08/2023
0000003A B90300                                 mov    cx, 3 ; (PCDOS 7.1)
                                    _seg_reinit_1:
0000003D AB                                     stosw                  ; modify Bios_Code entry points
0000003E 47                                     inc    di
0000003F 47                                     inc    di
00000040 E2FB                                   loop   _seg_reinit_1
                                                ; 10/08/2023 (PCDOS 7.1)
                                                ; (direct jump to i2f_handler from BIOSDATA:bios_i2f)
                                                ; (instead of 'bcode_i2f: dw i2f_handler, IOSYSCODESEG')
00000042 26A3[0800]                             mov     [es:bios_i2f_seg], ax ; actual BIOSCODE segment

00000046 CB                                     retf

                                    ; ----------------------------------------------------------------------------

                                    ; 15/10/2022

                                    ;****************************************************************************
                                    ;*                                                                          *
                                    ;*  chardev_entry - main device driver dispatch routine                     *
                                    ;*     called with a dummy parameter block on the stack                     *
                                    ;*     dw dispatch_table, dw prn/aux numbers (optional)                     *
                                    ;*                                                                          *
                                    ;*  will eventually take care of doing the transitions in                       *
                                    ;*     out of Bios_Code                                                     *
                                    ;*                                                                          *
                                    ;****************************************************************************

                                                ; 20/09/2023
                                    chardev_entry:                     ; 0070h:25B3h =       02C7h:0043h
00000047 56                                     push   si
00000048 50                                     push   ax
00000049 51                                     push   cx
0000004A 52                                     push   dx
0000004B 57                                     push   di
0000004C 55                                     push   bp
0000004D 1E                                     push   ds
0000004E 06                                     push   es
0000004F 53                                     push   bx
00000050 89E5                                   mov    bp, sp
00000052 8B7612                                 mov    si, [bp+18]    ; get return address (dispatch table)
                                                ;;mov  ds, word [cs:0030h]
                                                ;mov   ds, word ptr cs:BIOSDATAWORD ; [cs:0030h]
00000055 2E8E1E[3000]                           mov    ds, [cs:BIOSDATAWORD] ; 17/10/2022
                                                ; 20/09/2023 (PCDOS 7.1 - IBMBIO.COM - BIOSCODE:005Ah)
0000005A C434                                   les    si, [si]
                                                ;mov   ax, [si+2]     ; get the device number      if present
0000005C 8CC0                                   mov    ax, es
0000005E A2[2100]                               mov    [auxnum], al
00000061 8826[8004]                             mov    [printdev], ah
                                                ;mov   si, [si]       ; point to the device dispatch table
00000065 C41E[1200]                             les    bx, [ptrsav]   ; get pointer to i/o packet
00000069 268A4701                               mov    al, [es:bx+1]  ; [es:bx+unit] ; al = unit code
0000006D 268A670D                               mov    ah, [es:bx+13] ; [es:bx+media]       ; ah = media descrip
00000071 268B4F12                               mov    cx, [es:bx+18] ; [es:bx+count]       ; cx = count
00000075 268B5714                               mov    dx, [es:bx+20] ; [es:bx+start]       ; dx = start sector
                                                ; 17/10/2022
00000079 81FE[6F05]                             cmp    si, DSKTBL
                                                ;;cmp  si, 579h       ; (PCDOS 7.1 IBMBIO.COM)
                                                ;cmp   si, 4A2h       ; dsktbl
                                                                       ; at 2C7h:4A2h = 70h:2A12h
0000007D 7517                                   jnz    short no_sector32_mapping

                                    ; Special case for 32-bit start sector number:
                                    ;   if (si==dsktbl) /* if this is a disk device call */
```

```
10674                                  ;      set high 16 bits of secnum to 0
10675                                  ;      if (secnum == 0xffff) fetch 32 bit sector number
10676                                  ;
10677                                  ; pass high word of sector number in start_sec_h, low word in dx
10678                                  ;
10679                                  ; note: start_l and start_h are the offsets within the io_request packet
10680                                  ;     which contain the low and hi words of the 32 bit start sector if
10681                                  ;     it has been used.
10682                                  ;
10683                                  ; note: remember not to destroy the registers which have been set up before
10684
10685                                              ; 20/09/2023
10686                                              ;mov    ds:start_sec_h,       0 ; initialize to 0
10687 0000007F C706[9C04]0000                      mov     word [start_sec_h], 0
10688 00000085 83FAFF                             cmp     dx, 0FFFFh
10689 00000088 750C                               jnz     short no_sector32_mapping
10690 0000008A 268B571C                           mov     dx, [es:bx+28] ; [es:bx+start_h]
10691                                                     ; 32 bits dsk req
10692                                              ;mov    ds:start_sec_h,       dx ; start_sec_h = packet.start_h
10693 0000008E 8916[9C04]                         mov     [start_sec_h], dx
10694 00000092 268B571A                           mov     dx, [es:bx+26] ; [es:bx+start_l]
10695                                                     ; dx = packet.start_l
10696                                  no_sector32_mapping:
10697 00000096 97                                 xchg    ax, di
10698 00000097 268A4702                           mov     al, [es:bx+2]  ; [es:bx+cmd]
10699 0000009B 2E3A04                             cmp     al, [cs:si]
10700 0000009E 732B                               jnb     short command_error
10701 000000A0 98                                 cbw                    ; note that al <= 15 means ok
10702 000000A1 D1E0                               shl     ax, 1
10703 000000A3 01C6                               add     si, ax
10704 000000A5 97                                 xchg    ax, di
10705 000000A6 26C47F0E                           les     di, [es:bx+14] ; [es:bx+trans]
10706 000000AA FC                                 cld
10707                                             ; 17/10/2022
10708 000000AB 2EFF5401                           call    near [cs:si+1]
10709                                             ;call   word ptr cs:si+1
10710 000000AF 7202                               jb      short already_got_ah_status
10711 000000B1 B401                               mov     ah, 1
10712                                  already_got_ah_status:
10713                                             ;;mov   ds, [cs:0030h] ; 15/10/2022
10714                                             ;mov    ds, word ptr cs:BIOSDATAWORD ; [cs:0030h]
10715                                                                    ; cas note: shouldn't be needed!
10716 000000B3 2E8E1E[3000]                       mov     ds, [cs:BIOSDATAWORD] ; 17/10/2022
10717                                             ;lds    bx, ds:ptrsav
10718 000000B8 C51E[1200]                         lds     bx, [ptrsav]
10719 000000BC 894703                             mov     [bx+3], ax     ; [bx+status]
10720                                                                    ; mark operation complete
10721 000000BF 5B                                 pop     bx
10722 000000C0 07                                 pop     es
10723 000000C1 1F                                 pop     ds
10724 000000C2 5D                                 pop     bp
10725 000000C3 5F                                 pop     di
10726 000000C4 5A                                 pop     dx
10727 000000C5 59                                 pop     cx
10728 000000C6 58                                 pop     ax
10729 000000C7 5E                                 pop     si
10730                                             ;add    sp, 2          ; get rid of fake return address
10731                                             ; 20/09/2023 (PCDOS 7.1      - IBMBIO.COM - BIOSCODE:00C8h)
10732 000000C8 44                                 inc     sp
10733 000000C9 44                                 inc     sp
10734
10735                                             ; fall through into bc_retf
10736                                  ; ---------------------------------------------------------------------------
10737                                  bc_retf:
10738 000000CA CB                                 retf
10739                                  ; ---------------------------------------------------------------------------
10740
10741                                  command_error:
10742 000000CB E80700                             call    bc_cmderr
10743 000000CE EBE3                               jmp     short already_got_ah_status
10744                                  ; 15/10/2022
10745                                  ; 01/05/2019
10746
10747                                  ;---------------------------------------------------------------------------
10748                                  ; The following piece of hack is for supporting CP/M compatibility
10749                                  ; Basically at offset 5 we have a far call into 0:c0. But this does not call
10750                                  ; 0:c0 directly instead it call f01d:fef0, because it needs to support 'lhld 6'
10751                                  ; The following hack has to reside at ffff:d0 (= f01d:fef0) if BIOS is loaded
10752                                  ; high.
10753                                  ;---------------------------------------------------------------------------
10754
10755                                              ;db 7 dup(0)
10756
10757                                              ; 20/09/2023 - Retro DOS v4.2 IO.SYS (Optimization)
10758                                              ; (PCDOS 7.1 - IBMBIO.COM - BIOSCODE:0D0h)
10759                                              ; 15/10/2022
10760                                              ;dw 0          ; pad to bring offset to 0D0h
10761
10762 000000D0 00<rep 5h>              off_d0:     times 5 db 0   ; 5 bytes from 0:c0 will be copied onto here
10763                                                             ;  which is the CP/M call 5 entry point
10764
10765
10766                                  ; ---------------------------------------------------------------------------
10767
10768                                  ;   exit - all routines return through this path
10769
10770                                  bc_cmderr:
10771 000000D5 B003                               mov     al, 3          ; 2C7h:D5h = 70h:2645h
10772                                                                    ; unknown command error
10773
10774                                  ; =============== S U B   R O U T I N E =======================================
10775
10776                                  ;   now zero the count field by subtracting its current value,
10777                                  ;     which is still in cx, from itself.
10778
10779                                  ;   subtract the number of i/o's NOT YET COMPLETED from total
10780                                  ;     in order to return the number actually complete
10781
10782                                  bc_err_cnt:
10783                                              ;les    bx, ds:ptrsav
10784                                              ; 19/10/2022
10785 000000D7 C41E[1200]                         les     bx, [ptrsav]
10786 000000DB 26294F12                           sub     [es:bx+18], cx ; [es:bx+count]
10787                                                                    ; # of successful i/o's
10788 000000DF B481                               mov     ah, 81h        ; mark error return
10789 000000E1 F9                                 stc                    ; indicate abnormal end
10790 000000E2 C3                                 retn
10791
10792                                  ; 15/10/2022
10793
10794                                  ;Bios_Code ends
10795
10796                                  ;---------------------------------------------------------------------------
10797                                  ; MSCHAR.ASM - MSDOS 6.0 - 1991
```

```
                        ;--------------------------------------------------------------------------
                        ; 15/10/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 IO.SYS)
                        ; 10/01/2019 - Retro DOS v4.0

                        ; 30/04/2019

                        ;title     mschar - character and clock devices

                        ;MODE_CTRLBRK      equ     0FFh

                        ; BIOSCODE:00E4h (MSDOS 6.21, IO.SYS)

                        ;**************************************************************************
                        ;*                                                                        *
                        ;*  device driver dispatch tables                                         *
                        ;*                                                                        *
                        ;*  each table starts with a byte which lists the number of               *
                        ;*  legal functions, followed by that number of words. Each               *
                        ;*  word represents an offset of a routine in Bios_Code which             *
                        ;*  handles the function. The functions are terminated with               *
                        ;*  a near return. If carry is reset, a 'done' code is returned           *
                        ;*  to the caller. If carry is set, the ah/al registers are               *
                        ;*  returned as abnormal completion status. Notice that ds                *
                        ;*  is assumed to point to the Bios_Data segment throughout.              *
                        ;*                                                                        *
                        ;**************************************************************************

                                    ; 20/09/2023
                                    ; (PCDOS 7.1 - IBMBIO.COM - BIOSCODE:00E3h)
                                    ; 13/12/2022
000000E3 00                         db 0

                                    ; 13/12/2022
000000E4 0B             con_table:  db ((con_table_end - con_table)-1)/2 ; 11
                                                                ; 2C7h:0E4h = 70h:2654h
000000E5 [FA01]                     dw bc_exvec  ; 1FBh         ; bc_exvec at 2C7h:1FBh       = 70h:276Bh
                                                                ; 00 init
000000E7 [FA01]                     dw bc_exvec  ; 1FBh         ; 01
000000E9 [FA01]                     dw bc_exvec  ; 1FBh         ; 02
000000EB [D500]                     dw bc_cmderr ; 0D5h         ; bc_exvec at 2C7h:D5h = 70h:2645h
                                                                ; 03
000000ED [5C01]                     dw con_read  ; 15Ch         ; con_read at 2C7h:15Ch       = 70h:26CCh
                                                                ; 04
000000EF [9F01]                     dw con_rdnd  ; 19Fh         ; con_rdnd at 2C7h:19Fh       = 70h:270Fh
                                                                ; 05
000000F1 [FA01]                     dw bc_exvec  ; 1FBh         ; 06
000000F3 [0802]                     dw con_flush ; 209h         ; con_flush at 2C7h:209h = 70h:2779h
                                                                ; 07
000000F5 [FC01]                     dw con_writ  ; 1FDh         ; con_writ at 2C7h:1FDh       = 70h:276Dh
                                                                ; 08
000000F7 [FC01]                     dw con_writ  ; 1FDh         ; 09
000000F9 [FA01]                     dw bc_exvec  ; 1FBh         ; 0A
                        con_table_end:
000000FB 1A             prn_table:  db ((prn_table_end - prn_table)-1)/2 ; 26
                                                                ; 2C7h:0FBh = 70h:266Bh
000000FC [FA01]                     dw bc_exvec   ; 1FBh        ; bc_exvec
000000FE [FA01]                     dw bc_exvec   ; 1FBh        ; 01
00000100 [FA01]                     dw bc_exvec   ; 1FBh        ; 02
00000102 [D500]                     dw bc_cmderr  ;             0D5h    ; bc_cmderr
00000104 [1902]                     dw prn_input  ;             21Ah    ; prn_input
                                                                ; 04 indicate zero chars read
00000106 [C701]                     dw z_bus_exit ; 1C8h        ; z_bus_exit
                                                                ; 05 read non-destructive
00000108 [FA01]                     dw bc_exvec   ; 1FBh        ; 06
0000010A [FA01]                     dw bc_exvec   ; 1FBh        ; 07
0000010C [1E02]                     dw prn_writ   ;             21Fh    ; prn_writ
0000010E [1E02]                     dw prn_writ   ; 21Fh        ; 09
00000110 [4F02]                     dw prn_stat   ; 251h        ; prn_stat
00000112 [FA01]                     dw bc_exvec   ; 1FBh        ; 0B
00000114 [FA01]                     dw bc_exvec   ; 1FBh        ; 0C
00000116 [FA01]                     dw bc_exvec   ; 1FBh        ; 0D
00000118 [FA01]                     dw bc_exvec   ; 1FBh        ; 0E
0000011A [FA01]                     dw bc_exvec   ; 1FBh        ; 0F
0000011C [9402]                     dw prn_tilbusy ; 28Bh       ; prn_tilbusy
0000011E [FA01]                     dw bc_exvec   ; 1FBh        ; 11
00000120 [FA01]                     dw bc_exvec   ; 1FBh        ; 12
00000122 [C202]                     dw prn_genioctl ; 2BAh      ; prn_genioctl
00000124 [FA01]                     dw bc_exvec   ; 1FBh        ; 14
00000126 [FA01]                     dw bc_exvec   ; 1FBh        ; 15
00000128 [FA01]                     dw bc_exvec   ; 1FBh        ; 16
0000012A [FA01]                     dw bc_exvec   ; 1FBh        ; 17
0000012C [FA01]                     dw bc_exvec   ; 1FBh        ; 18
0000012E [F702]                     dw prn_ioctl_query ; 2F0h   ; prn_ioctl_query
                        prn_table_end:
00000130 0B             aux_table:  db ((aux_table_end - aux_table)-1)/2 ; 11
                                                                ; 2C7h:130h = 70h:26A0h
00000131 [FA01]                     dw bc_exvec   ; 1FBh        ; 00 - init
00000133 [FA01]                     dw bc_exvec   ; 1FBh        ; 01
00000135 [FA01]                     dw bc_exvec   ; 1FBh        ; 02
00000137 [D500]                     dw bc_cmderr  ;             0D5h    ; 03
00000139 [1203]                     dw aux_read   ; 30Dh        ; aux_read ; 04       - read
0000013B [3703]                     dw aux_rdnd   ; 335h        ; aux_rdnd - 05       - read non-destructive
0000013D [FA01]                     dw bc_exvec   ; 1FBh        ; 06
0000013F [7803]                     dw aux_flsh   ;             36Ch    ; aux_flsh
00000141 [7F03]                     dw aux_writ   ;             374h    ; aux_writ
00000143 [7F03]                     dw aux_writ   ;             374h    ; 09
00000145 [5703]                     dw aux_wrst   ;             355h    ; aux_wrst
                        aux_table_end:
00000147 0A             tim_table   db ((tim_table_end - tim_table)-1)/2 ; 10
                                                                ; 2C7h:147h = 70h:26B7h
00000148 [FA01]                     dw bc_exvec   ; 1FBh        ; 00
0000014A [FA01]                     dw bc_exvec   ; 1FBh        ; 01
0000014C [FA01]                     dw bc_exvec   ; 1FBh        ; 02
0000014E [D500]                     dw bc_cmderr  ;             0D5h    ; 03
00000150 [E404]                     dw tim_read   ;             435h    ; tim_read
00000152 [C701]                     dw z_bus_exit ; 1C8h        ; z_bus_exit
00000154 [FA01]                     dw bc_exvec   ; 1FBh        ; 06
00000156 [FA01]                     dw bc_exvec   ; 1FBh        ; 07
00000158 [E503]                     dw tim_writ   ; 3DBh        ; tim_writ
0000015A [E503]                     dw tim_writ   ; 3DBh        ; 09
                        tim_table_end:

                        ; --------------------------------------------------------------------------

                        ;**************************************************************************
                        ;*                                                                        *
                        ;*  con_read - read cx bytes from keyboard into buffer at es:di           *
                        ;*                                                                        *
                        ;**************************************************************************

                        con_read:                       ; 2C7h:15Ch = 70h:26CCh
                                    ;jcxz   short con_exit ; read cx bytes       from keyboard into buffer
0000015C E306                       jcxz    con_exit    ; 19/10/2022
                        con_loop:
```

```
10922 0000015E E80500                   call    chrin           ; get char in al
10923 00000161 AA                       stosb                   ; store char at es:di
10924 00000162 E2FA                     loop    con_loop
10925                           con_exit:
10926 00000164 F8                       clc
10927 00000165 C3                       retn
10928
10929                           ; =============== S U B   R O U T I N E =======================================
10930
10931                           ;****************************************************************************
10932                           ;*                                                                          *
10933                           ;*  chrin - input single char from keyboard into al                         *
10934                           ;*                                                                          *
10935                           ;*    we are going to issue extended keyboard function, if                  *
10936                           ;*    supported. the returning value of the extended keystroke              *
10937                           ;*    of the extended keyboard function uses 0E0h in al                     *
10938                           ;*    instead of 00h as in the conventional keyboard function.              *
10939                           ;*    this creates a conflict when the user entered real                    *
10940                           ;*    greek alpha charater (= 0E0h) to  distinguish the extended            *
10941                           ;*    keystroke and the greek alpha. this case will be handled              *
10942                           ;*    in the following manner:                                              *
10943                           ;*                                                                          *
10944                           ;*        ah = 16h                                                          *
10945                           ;*        int 16h                                                           *
10946                           ;*        if al == 0, then extended code (in ah)                            *
10947                           ;*        else if al == 0E0h, then                                          *
10948                           ;*        if ah <> 0, then extended code (in ah)                            *
10949                           ;*         else greek_alpha character.                                      *
10950                           ;*                                                                          *
10951                           ;*  also, for compatibility reason, if an extended code is                  *
10952                           ;*    detected, then we are going to change the value in al                 *
10953                           ;*    from 0E0h to 00h.                                                     *
10954                           ;*                                                                          *
10955                           ;****************************************************************************
10956
10957                                   ; 19/10/2022
10958                           chrin:
10959 00000166 8A26[7E04]               mov     ah, [keyrd_func] ; set by msinit. 0 or 10h
10960 0000016A 30C0                     xor     al, al
10961 0000016C 8606[0C00]               xchg    al, [altah]     ; get character       & zero altah
10962 00000170 08C0                     or      al, al
10963 00000172 752A                     jnz     short keyret
10964 00000174 CD16                     int     16h             ; KEYBOARD -
10965 00000176 09C0                     or      ax, ax
10966 00000178 74EC                     jz      short chrin
10967 0000017A 3D0072                   cmp     ax, 7200h       ; check for ctrl-prtsc
10968 0000017D 7504                     jnz     short alt_ext_chk
10969 0000017F B010                     mov     al, 10h
10970 00000181 EB1B                     jmp     short keyret
10971                           ; ---------------------------------------------------------------------------
10972
10973                           ;  if operation was extended function (i.e. keyrd_func != 0) then
10974                           ;    if character read was 0E0h then
10975                           ;      if extended byte was zero (i.e. ah == 0) then
10976                           ;    goto keyret
10977                           ;      else
10978                           ;    set al to zero
10979                           ;    goto alt_save
10980                           ;      endif
10981                           ;    endif
10982                           ;  endif
10983
10984                           alt_ext_chk:
10985 00000183 803E[7E04]00             cmp     byte [keyrd_func], 0
10986 00000188 740C                     jz      short not_ext
10987 0000018A 3CE0                     cmp     al, 0E0h
10988 0000018C 7508                     jnz     short not_ext
10989 0000018E 08E4                     or      ah, ah
10990 00000190 740C                     jz      short keyret
10991 00000192 30C0                     xor     al, al
10992 00000194 EB04                     jmp     short alt_save
10993                           ; ---------------------------------------------------------------------------
10994
10995                           not_ext:
10996 00000196 08C0                     or      al, al          ; special case?
10997 00000198 7504                     jnz     short keyret
10998                           alt_save:
10999 0000019A 8826[0C00]               mov     [altah], ah     ; store special key
11000                           keyret:
11001 0000019E C3                       retn
11002
11003                           ; ---------------------------------------------------------------------------
11004
11005                           ;****************************************************************************
11006                           ;*                                                                          *
11007                           ;*  con_rdnd - keyboard non destructive read, no wait                       *
11008                           ;*                                                                          *
11009                           ;*  pc-convertible-type machine: if bit 10 is set by the dos   *
11010                           ;*  in the status word of the request packet, and there is no  *
11011                           ;*  character in the input buffer, the driver issues a system  *
11012                           ;*  wait request to the rom. on return from the rom, it returns             *
11013                           ;*  a 'char-not-found' to the dos.                                          *
11014                           ;*                                                                          *
11015                           ;****************************************************************************
11016
11017                                   ; 19/10/2022
11018                           con_rdnd:
11019 0000019F A0[0C00]                 mov     al, [altah]
11020 000001A2 08C0                     or      al, al
11021 000001A4 754C                     jnz     short rdexit
11022 000001A6 8A26[7F04]               mov     ah, [keysts_func]
11023 000001AA CD16                     int     16h             ; KEYBOARD -
11024 000001AC 751D                     jnz     short gotchr
11025 000001AE 803E[7900]00             cmp     byte [fhavek09], 0
11026 000001B3 7412                     jz      short z_bus_exit
11027 000001B5 C41E[1200]               les     bx, [ptrsav]
11028                                   ; 12/12/2022
11029 000001B9 26F6470404               test    byte [es:bx+4], 04h
11030                                   ;test   word [es:bx+3], 400h ; [es:bx+status]
11031 000001BE 7407                     jz      short z_bus_exit
11032 000001C0 B80041                   mov     ax, 4100h
11033 000001C3 30DB                     xor     bl, bl
11034 000001C5 CD15                     int     15h             ; SYSTEM - WAIT       ON EXTERNAL EVENT (CONVERTIBLE)
11035                                                           ; AL = condition type, BH = condition compare or mask value
11036                                                           ; BL = timeout value times 55 milliseconds, 00h     means no timeout
11037                                                           ; DX = I/O port       address if AL bit 4 set
11038                           z_bus_exit:
11039 000001C7 F9                       stc                     ; 2C7h:1C8h = 70h:2738h
11040 000001C8 B403                     mov     ah, 3           ; indicate busy       status
11041 000001CA C3                       retn
11042                           ; ---------------------------------------------------------------------------
11043
11044                           gotchr:
11045 000001CB 09C0                     or      ax, ax
```

```
11046 000001CD 7508                     jnz     short notbrk   ; check for null after break
11047 000001CF 8A26[7E04]               mov     ah, [keyrd_func] ; issue keyboard read function
11048 000001D3 CD16                     int     16h            ; KEYBOARD -
11049 000001D5 EBC8                     jmp     short con_rdnd ; get a real status
11050                                   ; ---------------------------------------------------------------------------
11051
11052                                   notbrk:
11053 000001D7 3D0072                   cmp     ax, 7200h      ; check for ctrl-prtsc
11054 000001DA 7504                     jnz     short rd_ext_chk
11055 000001DC B010                     mov     al, 10h        ; ('P' & 1Fh) ; return control p
11056 000001DE EB12                     jmp     short rdexit
11057                                   ; ---------------------------------------------------------------------------
11058
11059                                   rd_ext_chk:
11060 000001E0 803E[7E04]00             cmp     byte [keyrd_func], 0 ; extended keyboard function?
11061 000001E5 740B                     jz      short rdexit
11062 000001E7 3CE0                     cmp     al, 0E0h       ; extended key value or      greek alpha?
11063 000001E9 7507                     jnz     short rdexit
11064 000001EB 80FC00                   cmp     ah, 0          ; scan code exist?
11065 000001EE 7402                     jz      short rdexit   ; yes. greek alpha char.
11066 000001F0 B000                     mov     al, 0          ; no. extended key stroke.
11067                                                          ; change it for       compatibility
11068                                   rdexit:
11069 000001F2 C41E[1200]               les     bx, [ptrsav]
11070 000001F6 2688470D                 mov     [es:bx+13], al ; [es:bx+media]
11071                                                          ; return keyboard character here
11072                                   bc_exvec:
11073 000001FA F8                       clc                    ; bc_exvec at 2C7h:1FBh      = 70h:276Bh
11074                                                          ; indicate normal termination
11075 000001FB C3                       retn
11076                                   ; ---------------------------------------------------------------------------
11077
11078                                   ;****************************************************************************
11079                                   ;*                                                                          *
11080                                   ;*  con_write - console write routine                                       *
11081                                   ;*                                                                          *
11082                                   ;*  entry:  es:di -> buffer                                                         *
11083                                   ;*          cx    =  count                                                  *
11084                                   ;*                                                                          *
11085                                   ;****************************************************************************
11086
11087                                   con_writ:
11088                                               ;jcxz   short bc_exvec
11089 000001FC E3FC                               jcxz    bc_exvec       ; 19/10/2022
11090                                             ; 12/12/2022
11091                                             ;jcxz   cc_ret
11092                                   con_lp:
11093 000001FE 268A05                             mov     al, [es:di]
11094 00000201 47                                 inc     di
11095 00000202 CD29                               int     29h            ; DOS 2+ internal - FAST PUTCHAR
11096                                                                    ; AL = character to display
11097 00000204 E2F8                               loop    con_lp
11098                                   cc_ret:
11099 00000206 F8                                 clc
11100 00000207 C3                                 retn
11101
11102                                   ; =============== S U B   R O U T I N E =======================================
11103
11104                                   ;****************************************************************************
11105                                   ;*                                                                          *
11106                                   ;*  con_flush - flush out keyboard queue                                    *
11107                                   ;*                                                                          *
11108                                   ;****************************************************************************
11109
11110                                   con_flush:
11111 00000208 C606[0C00]00                       mov     byte [altah], 0       ; clear out holding buffer
11112                                   flloop:                                 ; while (charavail()) charread();
11113 0000020D B401                               mov     ah, 1
11114 0000020F CD16                               int     16h            ; KEYBOARD - CHECK BUFFER, DO NOT CLEAR
11115                                                                    ; Return: ZF clear if character      in buffer
11116                                                                    ; AH = scan code, AL = character
11117                                                                    ; ZF set if no character in buffer
11118 00000211 74F3                               jz      short cc_ret
11119 00000213 30E4                               xor     ah, ah
11120 00000215 CD16                               int     16h            ; KEYBOARD - READ CHAR FROM BUFFER, WAIT IF EMPTY
11121                                                                    ; Return: AH = scan code, AL = character
11122 00000217 EBF4                               jmp     short flloop
11123
11124                                   ; ---------------------------------------------------------------------------
11125
11126                                   ; 15/10/2022
11127
11128                                   ;****************************************************************************
11129                                   ;*                                                                          *
11130                                   ;*  some equates for rom bios printer i/o                                       *
11131                                   ;*                                                                          *
11132                                   ;****************************************************************************
11133
11134                                   ; ibm rom status bits (i don't trust them, neither should you)
11135                                   ; warning!!! the ibm rom does not return just one bit. it returns a
11136                                   ; whole slew of bits, only one of which is correct.
11137
11138                                   ;notbusystatus     equ 10000000b         ; not busy
11139                                   ;nopaperstatus     equ 00100000b         ; no more paper
11140                                   ;prnselected       equ 00010000b         ; printer selected
11141                                   ;ioerrstatus       equ 00001000b         ; some kinda error
11142                                   ;timeoutstatus     equ 00000001b         ; time out.
11143                                   ;
11144                                   ;noprinter  equ 00110000b         ; no printer attached
11145
11146                                   ; 18/03/2019 - Retro DOS v4.0
11147                                   ;error_I24_out_of_paper   equ 9 ; MSDOS 6.0, ERR.INC, 1991
11148
11149                                   ; ---------------------------------------------------------------------------
11150
11151                                   ;****************************************************************************
11152                                   ;*                                                                          *
11153                                   ;*  prn_input - return with no error but zero chars read                        *
11154                                   ;*                                                                          *
11155                                   ;*  enter with cx = number of characters requested                          *
11156                                   ;*                                                                          *
11157                                   ;****************************************************************************
11158
11159                                   prn_input:                              ; 2C7h:21Ah = 70h:278Ah
11160 00000219 E8BBFE                             call    bc_err_cnt      ; reset count to zero
11161                                                                     ; (sub reqpkt.count,cx)
11162                                             ; 12/12/2022
11163                                   prn_done:
11164 0000021C F8                                 clc                     ; but return with carry      reset for no error
11165 0000021D C3                                 retn
11166                                   ; ---------------------------------------------------------------------------
11167
11168                                   ;****************************************************************************
11169                                   ;*                                                                          *
```

```
11170                                   ;*  prn_writ - write cx bytes from es:di to printer device           *
11171                                   ;*                                                                  *
11172                                   ;*  auxnum has printer number                                       *
11173                                   ;*                                                                  *
11174                                   ;********************************************************************
11175
11176                                   prn_writ:                       ; 2C7h:21Fh = 70h:278Fh
11177                                           ;jcxz  short prn_done ; no chars to output
11178 0000021E E3FC                       jcxz   prn_done       ; 19/10/2022
11179                                   prn_loop:
11180 00000220 BB0200                   mov    bx, 2          ; retry count
11181                                   prn_out:
11182 00000223 E83600                   call   prnstat        ; get status
11183 00000226 751D                     jnz    short TestPrnError
11184 00000228 268A05                   mov    al, [es:di]    ; get character       to print
11185 0000022B 30E4                     xor    ah, ah
11186 0000022D E82E00                   call   prnop          ; print to printer
11187 00000230 7419                     jz     short prn_con  ; no error - continue
11188 00000232 80FCFF                   cmp    ah, 0FFh       ; MODE_CTRLBRK
11189 00000235 7509                     jnz    short _prnwf
11190 00000237 B00C                     mov    al, 0Ch        ; error_I24_gen_failure
11191 00000239 C606[0C00]00             mov    byte [altah], 0
11192 0000023E EB08                     jmp    short pmessg
11193                                   ; ---------------------------------------------------------------------------
11194
11195                                   _prnwf:
11196 00000240 F6C401                   test   ah, 1          ; timeoutstatus
11197 00000243 7406                     jz     short prn_con
11198                                   TestPrnError:
11199 00000245 4B                       dec    bx             ; retry until count is exhausted.
11200 00000246 75DB                     jnz    short prn_out
11201                                   pmessg:
11202 00000248 E98CFE                   jmp    bc_err_cnt
11203                                   ; ---------------------------------------------------------------------------
11204
11205                                   prn_con:
11206 0000024B 47                       inc    di             ; point to next char and continue
11207 0000024C E2D2                     loop   prn_loop
11208                                   ;prn_done:
11209                                           ; 12/12/2022
11210                                   prn_done2:
11211                                           ;clc
11212                                           ; cf=0
11213 0000024E C3                       retn
11214                                   ; ---------------------------------------------------------------------------
11215
11216                                   ;********************************************************************
11217                                   ;*                                                                  *
11218                                   ;*  prn_stat - device driver entry to return printer status         *
11219                                   ;*                                                                  *
11220                                   ;********************************************************************
11221
11222                                   prn_stat:                       ; 2C7h:251h = 70h:27C1h
11223 0000024F E80A00                   call   prnstat        ; device in dx
11224 00000252 75F4                     jnz    short pmessg
11225 00000254 F6C480                   test   ah, 80h        ; notbusystatus
11226                                   ;jnz   short prn_done
11227                                   ; 12/12/2022
11228 00000257 75F5                     jnz    short prn_done2 ; cf=0
11229 00000259 E96BFF                   jmp    z_bus_exit
11230                                   ; ---------------------------------------------------------------------------
11231
11232                                   ;********************************************************************
11233                                   ;*                                                                  *
11234                                   ;*  prnstat - utility function to call ROM BIOS to check            *
11235                                   ;*           printer status. Return meaningful error code           *
11236                                   ;*                                                                  *
11237                                   ;********************************************************************
11238
11239                                   prnstat:
11240 0000025C B402                     mov    ah, 2          ; set command for get status
11241                                                          ; PRINTER - GET       STATUS
11242                                                          ; DX = printer port (0-3)
11243                                                          ; Return: AH = status
11244
11245                                   ; =============== S U B   R O U T I N E =======================================
11246
11247                                   ;********************************************************************
11248                                   ;*                                                                  *
11249                                   ;*  prnop - call ROM BIOS printer function in ah                    *
11250                                   ;*          return zero true if no error                            *
11251                                   ;*          return zero false if error, al = error code             *
11252                                   ;*                                                                  *
11253                                   ;********************************************************************
11254
11255                                   prnop:
11256                                           ; 20/12/2023 - Retro DOS v5.0
11257                                           ; PCDOS 7.1 IBMBIO.COM
11258
11259                                           ;mov   dx, [auxnum]   ; get printer number
11260                                           ;int   17h
11261
11262 0000025E 1E                           push   ds
11263 0000025F FF36[2100]                   push   word [auxnum]
11264 00000263 31D2                         xor    dx, dx ; 0
11265 00000265 8EDA                         mov    ds, dx
11266 00000267 5A                           pop    dx
11267 00000268 9C                           pushf                  ; simulate int 17h
11268 00000269 FA                           cli
11269                                       ;call  dword ptr ds:5ghghCh
11270 0000026A FF1E5C00                     call   far [17h*4]    ; 0:5Ch = INT 17h vector
11271 0000026E 1F                           pop    ds
11272
11273                                   ; This check was added to see if this is a case of no
11274                                   ; printer being installed. This tests checks to be sure
11275                                   ; the error is noprinter (30h)
11276
11277 0000026F 50                           push   ax
11278 00000270 80E430                       and    ah, 30h
11279 00000273 80FC30                       cmp    ah, 30h        ; noprinter
11280 00000276 58                           pop    ax
11281 00000277 7506                         jnz    short NextTest
11282 00000279 80E4DF                       and    ah, 0DFh       ; ~nopaperstatus
11283 0000027C 80CC08                       or     ah, 8          ; ioerrstatus
11284
11285                                   ; examine the status bits to see if an error occurred. unfortunately, several
11286                                   ; of the bits are set so we have to pick and choose. we must be extremely
11287                                   ; careful about breaking basic.
11288
11289                                   NextTest:
11290 0000027F F6C428                       test   ah, 28h        ; (ioerrstatus+nopaperstatus)
11291                                                             ; i/o error?
11292 00000282 740A                         jz     short checknotready ; no, try not ready
11293
```

```
                                ; at this point, we know we have an error. the converse is not true
                                
00000284 B009                           mov     al, 9           ; error_I24_out_of_paper
                                                                ; first, assume       out of paper
00000286 F6C420                         test    ah, 20h         ; out of paper set?
00000289 7502                           jnz     short ret1      ; yes, error is       set
0000028B FEC0                           inc     al              ; return al=10 (i/o error)
                                ret1:
0000028D C3                             retn
                                ; ---------------------------------------------------------------------------
                                
                                checknotready:
0000028E B002                           mov     al, 2           ; assume not-ready
00000290 F6C401                         test    ah, 1
00000293 C3                             retn
                                
                                ; ---------------------------------------------------------------------------
                                
                                ;****************************************************************************
                                ;*                                                                          *
                                ;*  prn_tilbusy - output until busy. Used by print spooler.                 *
                                ;*                this entry point should never block waiting for       *
                                ;*                device to come ready.                                   *
                                ;*                                                                        *
                                ;*  inputs: cx = count, es:di -> buffer                                   *
                                ;*  outputs: set the number of bytes transferred in the                   *
                                ;*           device driver request packet                                 *
                                ;*                                                                        *
                                ;****************************************************************************
                                
                                        ; 19/10/2022
                                prn_tilbusy:                            ; 2C7h:28Bh = 70h:27FBh
00000294 89FE                           mov     si, di          ; everything is       set for lodsb
                                prn_tilbloop:
00000296 51                             push    cx
00000297 53                             push    bx
00000298 30FF                           xor     bh, bh
0000029A 8A1E[8004]                     mov     bl, [printdev]
0000029E D1E3                           shl     bx, 1
                                        ;mov    cx, ds:wait_count[bx] ;       wait count times to come ready
000002A0 8B8F[8104]                     mov     cx, [wait_count+bx]
000002A4 5B                             pop     bx
                                prn_getstat:
000002A5 E8B4FF                         call    prnstat         ; get status
000002A8 7514                           jnz     short prn_bperr       ; error
000002AA F6C480                         test    ah, 80h         ; ready yet?
000002AD E1F6                           loope   prn_getstat     ; no, go for more
000002AF 59                             pop     cx              ; get original count
000002B0 740D                           jz      short prn_berr ; still not ready => done
000002B2 26                             es
000002B3 AC                             lodsb
                                        ;lods   byte ptr es:[si] ; es
                                                                ; lodsb
000002B4 30E4                           xor     ah, ah
000002B6 E8A5FF                         call    prnop
000002B9 7504                           jnz     short prn_berr ; error
000002BB E2D9                           loop    prn_tilbloop
                                        ; 12/12/2022
                                        ; cf=0 (prnop)
                                        ;clc                    ; normal no-error return
000002BD C3                             retn                    ;   from device driver
                                
                                ; ---------------------------------------------------------------------------
                                
                                prn_bperr:
000002BE 59                             pop     cx              ; restore transfer count from stack
                                prn_berr:
000002BF E915FE                         jmp     bc_err_cnt
                                ; ---------------------------------------------------------------------------
                                
                                ; 15/10/2022
                                
                                ;****************************************************************************
                                ;*                                                                          *
                                ;*  prn_genioctl - get/set printer retry count                              *
                                ;*                                                                          *
                                ;****************************************************************************
                                
                                ; IOCTL.INC (MSDOS 6.0, 1991)
                                ; 11/01/2019
                                
                                ;*********************************;*
                                ; CHARACTER DEVICES (PRINTERS)     ;*
                                ;*********************************;*
                                
                                ;;RAWIO SUB-FUNCTIONS
                                ;;get_retry_count equ 65h
                                ;;set_retry_count equ 45h
                                
                                ;;struc A_RETRYCOUNT
                                ;;.rc_count: resw 1
                                ;;endstruc
                                
                                ;ioc_pc equ 5
                                
                                ; ---------------------------------------------------------------------------
                                
                                        ; 19/10/2022
                                prn_genioctl:                           ; 2C7h:2BAh = 70h:282Ah
000002C2 C43E[1200]                     les     di, [ptrsav]
000002C6 26807D0D05                     cmp     byte [es:di+13], 5 ; [es:di+IOCTL_REQ.MAJORFUNCTION]
                                                                ; ioc_pc
000002CB 7403                           jz      short prnfunc_ok
                                
                                prnfuncerr:
000002CD E905FE                         jmp     bc_cmderr
                                ; ---------------------------------------------------------------------------
                                
                                prnfunc_ok:
000002D0 268A450E                       mov     al, [es:di+14] ; [es:di+IOCTL_REQ.MINORFUNCTION]
000002D4 26C47D13                       les     di, [es:di+19] ; [es:di+IOCTL_REQ.GENERICIOCTL_PACKET]
000002D8 30FF                           xor     bh, bh
                                        ;mov    bl, ds:printdev       ; get index into retry counts
000002DA 8A1E[8004]                     mov     bl, [printdev]
000002DE D1E3                           shl     bx, 1
                                        ;mov    cx, ds:wait_count[bx] ;       pull out retry count for device
000002E0 8B8F[8104]                     mov     cx, [wait_count+bx]
000002E4 3C65                           cmp     al, 65h         ; get_retry_count
000002E6 7407                           jz      short prngetcount
000002E8 3C45                           cmp     al, 45h         ; set_retry_count
000002EA 75E1                           jnz     short prnfuncerr
000002EC 268B0D                         mov     cx, [es:di]
                                prngetcount:
                                        ;mov    ds:wait_count[bx], cx
```

```
11418 000002EF 898F[8104]                       mov     [wait_count+bx], cx
11419 000002F3 26890D                           mov     [es:di], cx     ; [es:di+A_RETRYCOUNT.RC_COUNT]
11420                                                                   ; return current retry count
11421                                           ; 12/12/2022
11422                                           ; cf=0
11423                                           ;clc
11424 000002F6 C3                               retn
11425                                   ; ---------------------------------------------------------------------------
11426
11427                                   ;****************************************************************************
11428                                   ;*                                                                          *
11429                                   ;*  prn_ioctl_query                                                         *
11430                                   ;*                                                                          *
11431                                   ;*  Added for 5.00                                                          *
11432                                   ;****************************************************************************
11433
11434                                   prn_ioctl_query:                ; 2C7h:2F0h = 70h:2860h
11435 000002F7 C43E[1200]                           les     di, [ptrsav]
11436 000002FB 26807D0D05                           cmp     byte [es:di+13], 5 ; [es:di+IOCTL_REQ.MAJORFUNCTION]
11437                                                                   ; ioc_pc
11438 00000300 750D                               jnz     short prn_query_err
11439 00000302 268A450E                           mov     al, [es:di+14] ; [es:di+IOCTL_REQ.MINORFUNCTION]
11440 00000306 3C65                               cmp     al, 65h         ; GET_RETRY_COUNT
11441 00000308 7404                               jz      short IOCtlSupported
11442 0000030A 3C45                               cmp     al, 45h         ; SET_RETRY_COUNT
11443 0000030C 7501                               jnz     short prn_query_err
11444                                   IOCtlSupported:
11445                                             ; 12/12/2022
11446                                             ; cf=0
11447                                             ;clc
11448 0000030E C3                                 retn
11449                                   ; ---------------------------------------------------------------------------
11450
11451                                   prn_query_err:
11452                                             ; 12/12/2022
11453                                             ;stc
11454 0000030F E9C3FD                             jmp     bc_cmderr ; (bc_cmderr sets cf to 1)
11455                                   ; ---------------------------------------------------------------------------
11456
11457                                   ;****************************************************************************
11458                                   ;*                                                                          *
11459                                   ;*  aux port driver code -- "aux" == "com1"                                 *
11460                                   ;*                                                                          *
11461                                   ;*  the device driver entry/dispatch code sets up auxnum to                 *
11462                                   ;*  give the com port number to use (0=com1, 1=com2, 2=com3...)             *
11463                                   ;*                                                                          *
11464                                   ;****************************************************************************
11465
11466                                   ;   values in ah, requesting function of int 14h in rom bios
11467
11468                                   ;auxfunc_send       equ   1       ;transmit
11469                                   ;auxfunc_receive equ      2       ;read
11470                                   ;auxfunc_status     equ   3       ;request status
11471
11472                                   ;   error flags, reported by int 14h, reported in ah:
11473
11474                                   ;flag_data_ready equ      01h     ;data ready
11475                                   ;flag_overrun       equ   02h     ;overrun error
11476                                   ;flag_parity        equ   04h     ;parity error
11477                                   ;flag_frame  equ    08h     ;framing error
11478                                   ;flag_break  equ    10h     ;break detect
11479                                   ;flag_tranhol_emp equ     20h     ;transmit holding register empty
11480                                   ;flag_timeout       equ   80h     ;timeout
11481
11482                                   ;   these flags reported in al:
11483
11484                                   ;flag_cts    equ    10h     ;clear to send
11485                                   ;flag_dsr    equ    20h     ;data set ready
11486                                   ;flag_rec_sig       equ    80h     ;receive line signal detect
11487
11488                                   ; ---------------------------------------------------------------------------
11489
11490                                   ;****************************************************************************
11491                                   ;*                                                                          *
11492                                   ;*  aux_read - read cx bytes from [auxnum] aux port to buffer               *
11493                                   ;*             at es:di                                                     *
11494                                   ;*                                                                          *
11495                                   ;****************************************************************************
11496
11497                                   aux_read:                       ; 2C7h:30Dh = 70h:287Dh
11498                                             ;jcxz   short exvec2
11499 00000312 E311                               jcxz    exvec2          ; 19/10/2022
11500 00000314 E88000                             call    getbx           ; put address of auxbuf       in bx
11501 00000317 30C0                               xor     al, al
11502 00000319 8607                               xchg    al, [bx]
11503 0000031B 08C0                               or      al, al
11504 0000031D 7503                               jnz     short aux2
11505                                   aux1:
11506 0000031F E80500                             call    auxin           ; get character       from port
11507                                                                     ; won't return if error
11508                                   aux2:
11509 00000322 AA                                 stosb
11510 00000323 E2FA                               loop    aux1            ; if more characters, go around      again
11511                                   exvec2:
11512 00000325 F8                                 clc                     ; all done, successful exit
11513                                   auxin_retn: ; 18/12/2022
11514 00000326 C3                                 retn
11515                                   ; ---------------------------------------------------------------------------
11516
11517                                   ;****************************************************************************
11518                                   ;*                                                                          *
11519                                   ;*  auxin - call rom bios to read character from aux port                   *
11520                                   ;*          if error occurs, map the error and return one                   *
11521                                   ;*          level up to device driver exit code, setting                    *
11522                                   ;*          the number of bytes transferred appropriately                   *
11523                                   ;*                                                                          *
11524                                   ;****************************************************************************
11525
11526                                   auxin:
11527 00000327 B402                               mov     ah, 2           ; auxfunc_receive
11528 00000329 E83A00                             call    auxop
11529 0000032C F6C40E                             test    ah, 0Eh         ; flag_frame|flag_parity|flag_overrun
11530                                               ;jnz    short arbad     ; skip if any error bits set
11531                                               ;retn
11532                                             ; 25/06/2023 (BugFix)
11533 0000032F 74F5                               jz      short auxin_retn
11534                                   ; ---------------------------------------------------------------------------
11535
11536                                   arbad:
11537 00000331 58                                 pop     ax              ; remove return       address (near call)
11538                                             ;xor    al, al
11539                                             ;or     al, 0B0h        ; flag_rec_sig|       flag_dsr|flag_cts
11540                                             ; 11/08/2023
11541 00000332 B0B0                               mov     al, 0B0h        ; PCDOS 7.1 IBMBIO.COM - BIOSCODE:0334h
```

```
11542 00000334 E9A0FD                          jmp     bc_err_cnt
11543
11544                                  ; ---------------------------------------------------------------------------
11545
11546                                  ;****************************************************************************
11547                                  ;*                                                                          *
11548                                  ;*  aux_rdnd - non-destructive aux port read                                *
11549                                  ;*                                                                          *
11550                                  ;****************************************************************************
11551
11552                                  aux_rdnd:                               ; 2C7h:335h = 70h:28A5h
11553 00000337 E85D00                          call    getbx
11554 0000033A 8A07                            mov     al, [bx]        ; have bx point       to auxbuf
11555 0000033C 08C0                            or      al, al          ; if al is non-zero (char in buffer)
11556 0000033E 7511                            jnz     short auxdrx    ; then return character
11557 00000340 E82100                          call    auxstat         ; if not, get status of       aux device
11558 00000343 F6C401                          test    ah, 1           ; flag_data_ready - test data ready
11559 00000346 740C                            jz      short auxbus    ; then device is busy (not ready)
11560 00000348 A820                            test    al, 20h         ; flag_dsr - test data set ready
11561 0000034A 7408                            jz      short auxbus    ; then device is busy (not ready)
11562 0000034C E8D8FF                          call    auxin           ; else aux is ready, get character
11563 0000034F 8807                            mov     [bx], al
11564                                  auxdrx:
11565 00000351 E99EFE                          jmp     rdexit          ; return busy status
11566                                  ; ---------------------------------------------------------------------------
11567
11568                                  auxbus:
11569 00000354 E970FE                          jmp     z_bus_exit
11570                                  ; ---------------------------------------------------------------------------
11571
11572                                  ;****************************************************************************
11573                                  ;*                                                                          *
11574                                  ;*  aux_wrst - return aux port write status                                 *
11575                                  ;*                                                                          *
11576                                  ;****************************************************************************
11577
11578                                  aux_wrst:                               ; 2C7h:355h = 70h:28C5h
11579 00000357 E80A00                          call    auxstat         ; get status of       aux in ax
11580 0000035A A820                            test    al, 20h         ; test data set       ready
11581 0000035C 74F6                            jz      short auxbus    ; then device is busy (not ready)
11582 0000035E F6C420                          test    ah, 20h         ; flag_tranhol_emp - test transmit hold     reg empty
11583 00000361 74F1                            jz      short auxbus    ; then device is busy (not ready)
11584                                          ; 12/12/2022
11585                                          ; cf=0  ; (test instruction resets cf)
11586                                          ;clc
11587 00000363 C3                              retn
11588                                  ; ---------------------------------------------------------------------------
11589
11590                                  ;****************************************************************************
11591                                  ;*                                                                          *
11592                                  ;*  auxstat - call rom bios to determine aux port status                    *
11593                                  ;*                                                                          *
11594                                  ;*  exit:   ax = status                                                     *
11595                                  ;*          dx = [auxnum]                                                   *
11596                                  ;*                                                                          *
11597                                  ;****************************************************************************
11598
11599                                  auxstat:
11600 00000364 B403                            mov     ah, 3           ; auxfunc_status
11601
11602                                          ; fall into auxop
11603
11604                                  ; =============== S U B   R O U T I N E =======================================
11605
11606                                  ;****************************************************************************
11607                                  ;*                                                                          *
11608                                  ;*  auxop - perform rom-biox aux port interrupt                             *
11609                                  ;*                                                                          *
11610                                  ;*  entry:  ah = int 14h function number                                    *
11611                                  ;*  exit:   ax = results                                                    *
11612                                  ;*          dx = [auxnum]                                                   *
11613                                  ;*                                                                          *
11614                                  ;****************************************************************************
11615
11616                                  auxop:                  ; proc near
11617                                  ; 20/12/2023 - Retro DOS v5.0
11618                                  ;mov    dx, [auxnum]    ; ah=function code
11619                                  ;                       ; 0=init, 1=send, 2=receive, 3=status
11620                                  ;                       ; get port number
11621                                  ;
11622                                  ;int    14h             ; SERIAL I/O - GET USART STATUS
11623                                  ;                       ; DX = port number (0-3)
11624                                  ;                       ; Return: AX = port status code
11625                                  ; (PCDOS 7.1 IBMBIO.COM)
11626 00000366 1E                              push    ds
11627 00000367 FF36[2100]                      push    word [auxnum]
11628 0000036B 31D2                            xor     dx, dx ; 0
11629 0000036D 8EDA                            mov     ds, dx
11630 0000036F 5A                              pop     dx
11631 00000370 9C                              pushf                   ; simulate INT 14h
11632 00000371 FA                              cli
11633                                          ;call   dword ptr ds:50h
11634 00000372 FF1E5000                        call    far [14h*4]     ; INT 14h vector (14h*4 = 50h)
11635 00000376 1F                              pop     ds
11636 00000377 C3                              retn
11637
11638                                  ; ---------------------------------------------------------------------------
11639
11640                                  ;****************************************************************************
11641                                  ;*                                                                          *
11642                                  ;*  aux_flsh - flush aux input buffer - set contents of                     *
11643                                  ;*             auxbuf [auxnum] to zero                                      *
11644                                  ;*                                                                          *
11645                                  ;*  cas - shouldn't this code call the rom bios input function              *
11646                                  ;*        repeatedly until it isn't ready?  to flush out any                *
11647                                  ;*        pending serial input queue if there's a tsr like MODE             *
11648                                  ;*        which is providing interrupt-buffering of aux port?               *
11649                                  ;*                                                                          *
11650                                  ;****************************************************************************
11651
11652                                  aux_flsh:                               ; 2C7h:36Ch = 70h:28DCh
11653 00000378 E81C00                          call    getbx                   ; flush aux input buffer
11654 0000037B C60700                          mov     byte [bx], 0    ; get bx to point to auxbuf
11655                                                                  ; zero out buffer
11656                                          ;clc                    ; all done, successful return
11657                                          ; 12/12/2022
11658                                          ; cf=0 ('add' instruction in 'getbx')
11659 0000037E C3                              retn
11660                                  ; ---------------------------------------------------------------------------
11661
11662                                  ;****************************************************************************
11663                                  ;*                                                                          *
11664                                  ;*  aux_writ - write to aux device                                          *
11665                                  ;*                                                                          *
```

```
                        ;****************************************************************************
                        
                        aux_writ:                               ; 2C7h:374h = 70h:28E4h
                                ;jcxz   short exvec2    ; write to aux device (if cx > 0)
0000037F E3A4                   jcxz    exvec2          ; 19/10/2022
                        aux_loop:
00000381 268A05                 mov     al, [es:di]     ; get character       to be written
                                                        ; move di pointer to next character
00000384 47                     inc     di
00000385 B401                   mov     ah, 1           ; auxfunc_send - indicates a write
00000387 E8DCFF                 call    auxop           ; send character over aux port
0000038A F6C480                 test    ah, 80h         ; check for error
0000038D 7405                   jz      short awok      ; then no error
0000038F B00A                   mov     al, 10          ; else indicate       write fault
00000391 E943FD                 jmp     bc_err_cnt      ; call error routines
                        ; ---------------------------------------------------------------------------
                        
                        awok:
00000394 E2EB                   loop    aux_loop        ; if cx is non-zero,
                                                        ; still more character to print
                                ;clc                    ; all done, successful return
                                ; 12/12/2022
                                ; cf=0 (test instruction above)
00000396 C3                     retn
                        
                        ; =============== S U B   R O U TI N E =======================================
                        
                        ;****************************************************************************
                        ;*                                                                          *
                        ;*  getbx - return bx -> single byte input buffer for                       *
                        ;*          selected aux port ([auxnum])                                    *
                        ;*                                                                          *
                        ;****************************************************************************
                        
                        getbx:
00000397 8B1E[2100]             mov     bx, [auxnum]    ; return bx -> single byte input buffer
                                                        ; for selected aux port      ([auxnum])
                                ;add    bx, offset auxbuf
0000039B 81C3[1600]             add     bx, auxbuf      ; 19/10/2022
                                ; 12/12/2022
                                ; cf=0 (if [auxnum] is valid number)
0000039F C3                     retn
                        
                        ; ---------------------------------------------------------------------------
                        
                        ; 15/10/2022
                        
                        ;--------------------------------------------------------------
                        ;                                                              :
                        ;              clock device driver                             :
                        ;                                                              :
                        ;                                                              :
                        ;   this file contains the clock device driver.                :
                        ;                                                              :
                        ;   the routines in this files are:                                :
                        ;                                                              :
                        ;   routine             function                           :
                        ;   -------             --------                           :
                        ;   tim_writ            set the current time               :
                        ;   tim_read            read the current time              :
                        ;   time_to_ticks       convert time to corresponding      :
                        ;                         number of clock ticks            :
                        ;                                                          :
                        ; the clock ticks at the rate of:                          :
                        ;                                                          :
                        ;   1193180/65536 ticks/second (about 18.2 ticks per second):
                        ; see each routine for information on the use.                 :
                        ;                                                          :
                        ;--------------------------------------------------------------
                        
                        ; convert time to ticks
                        ; input : time in cx and dx
                        ; ticks returned in cx:dx
                        
                        ;19/07/2019
                        ;09/03/2019
                        
                        time_to_ticks:                          ; 0070h:2906h =       02C7h:0396h
                        
                        ; first convert from hour,min,sec,hund. to
                        ; total number of 100th of seconds
                        
000003A0 B03C                   mov     al, 60
000003A2 F6E5                   mul     ch              ; hours to minutes
000003A4 B500                   mov     ch, 0
000003A6 01C8                   add     ax, cx          ; total minutes
000003A8 B97017                 mov     cx, 6000        ; 60*100
000003AB 89D3                   mov     bx, dx          ; get out of the way of       the multiply
000003AD F7E1                   mul     cx              ; convert to 1/100 sec
000003AF 89C1                   mov     cx, ax
000003B1 B064                   mov     al, 100
000003B3 F6E7                   mul     bh              ; convert seconds to 1/100 sec
000003B5 01C1                   add     cx, ax          ; combine seconds with hours and min
000003B7 83D200                 adc     dx, 0           ; ripple carry
000003BA B700                   mov     bh, 0
000003BC 01D9                   add     cx, bx          ; combine 1/100       sec
000003BE 83D200                 adc     dx, 0
                        
                            ; dx:cx is time in 1/100 sec
                        
000003C1 92                     xchg    ax, dx
000003C2 91                     xchg    ax, cx          ; now time is in cx:ax
000003C3 BB0BE9                 mov     bx, 59659
000003C6 F7E3                   mul     bx              ; multiply low half
000003C8 87D1                   xchg    dx, cx
000003CA 92                     xchg    ax, dx          ; cx->ax, ax->dx, dx->cx
000003CB F7E3                   mul     bx              ; multiply high       half
000003CD 01C8                   add     ax, cx          ; combine overlapping products
000003CF 83D200                 adc     dx, 0
000003D2 92                     xchg    ax, dx          ; ax:dx=time*59659
000003D3 BB0500                 mov     bx, 5
000003D6 F6F3                   div     bl              ; divide high half by 5
000003D8 88C1                   mov     cl, al
000003DA B500                   mov     ch, 0
000003DC 88E0                   mov     al, ah          ; remainder of divide-by-5
000003DE 98                     cbw
000003DF 92                     xchg    ax, dx          ; use it to extend low half
000003E0 F7F3                   div     bx              ; divide low half by 5
000003E2 89C2                   mov     dx, ax          ; cx:dx is now number of ticks in time
000003E4 CB                     retf                    ; far return
                        
                        ; ---------------------------------------------------------------------------
                        
                        ; 17/10/2022
```

```
                                ; 15/10/2022
                                ;--------------------------------------------------------------------
                                ;
                                ; tim_writ sets the current time
                                ;
                                ; on entry es:[di] has the current time:
                                ;
                                ;   number of days since 1-1-80  (word)
                                ;   minutes (0-59)               (byte)
                                ;   hours (0-23)                 (byte)
                                ;   hundredths of seconds (0-99) (byte)
                                ;   seconds (0-59)               (byte)
                                ;
                                ; each number has been checked for the correct range.
                                ;
                                ;   NOTE: Any changes in this routine probably require corresponding
                                ;   changes in the version that is built with the power manager driver.
                                ;   See ptime.asm.
                                ;
                                ;--------------------------------------------------------------------

                                    ; 08/08/2023 - Retro DOS v4.2 IO.SYS (optimization)
                                    ; PCDOS 7.1 IBMBIO.COM - BIOSCODE:03EAh
                                tim_writ:                       ; 2C7h:3DBh = 70h:294Bh
000003E5 268B05                             mov     ax, [es:di]
000003E8 50                                 push    ax              ; daycnt. we need to set this at the very
                                                                    ; end to avoid tick windows.
000003E9 803E[8C04]00                       cmp     byte [havecmosclock], 0
                                            ;cmp    ds:havecmosclock, 0
000003EE 7423                               jz      short no_cmos_1
000003F0 268A4503                           mov     al, [es:di+3]  ; near indirect       calls
                                                                    ; get binary hours
                                                                    ; convert to bcd
                                            ;call   far [bintobcd]
                                            ;;call  ds:bintobcd    ; call far [bintobcd]
                                            ; 08/08/2023
000003F4 E8E800                             call    bintobcd
000003F7 88C5                               mov     ch, al         ; ch = bcd hours
000003F9 268A4502                           mov     al, [es:di+2]  ; get binary minutes
                                            ;call   far [bintobcd]
                                            ;;call  ds:bintobcd    ; convert to bcd
000003FD E8DF00                             call    bintobcd
00000400 88C1                               mov     cl, al         ; cl = bcd minutes
00000402 268A4505                           mov     al, [es:di+5]  ; get binary seconds
                                            ;call   far [bintobcd]
                                            ;;call  ds:bintobcd
00000406 E8D600                             call    bintobcd

00000409 88C6                               mov     dh, al         ; dh = bcd seconds
0000040B B200                               mov     dl, 0          ; dl = 0 (st) or 1 (dst)
0000040D FA                                 cli
0000040E B403                               mov     ah, 3
00000410 CD1A                               int     1Ah            ; CLOCK - SET REAL TIME       CLOCK (AT,XT286,CONV,PS)
                                                                    ; CH = hours in       BCD, CL = minutes in BCD
                                                                    ;  DH = seconds in BCD,DL = 01h       if daylight savings, 00h if standard
time
                                                                    ; Return: CMOS clock set
00000412 FB                                 sti
                                no_cmos_1:
00000413 268B4D02                           mov     cx, [es:di+2]
00000417 268B5504                           mov     dx, [es:di+4]
                                            ; 17/10/2022
0000041B FF1E[0606]                         call    far [ttticks]
                                            ;call   dword ptr ds:ttticks ; call far      [ttticks]
                                                                    ; convert time to ticks
                                                                    ; cx:dx now has time in ticks
0000041F FA                                 cli                     ; turn off timer
00000420 B401                               mov     ah, 1
00000422 CD1A                               int     1Ah             ; CLOCK - SET TIME OF DAY
                                                                    ; CX:DX = clock count
                                                                    ; Return: time of day set
                                            ;pop    ds:daycnt
00000424 8F06[8904]                         pop     word [daycnt]
00000428 FB                                 sti
                                            ;cmp    ds:havecmosclock, 0
00000429 803E[8C04]00                       cmp     byte [havecmosclock], 0
0000042E 7409                               jz      short no_cmos_2

                                            ; 08/08/2023
                                            ;call   far [daycnttoday]
                                            ;;call  ds:daycnttoday ; call far [daycnttoday]
                                                                    ; convert to bcd format
00000430 E80700                             call    daycnttoday

00000433 FA                                 cli
00000434 B405                               mov     ah, 5
00000436 CD1A                               int     1Ah             ; CLOCK - SET DATE IN REAL TIME       CLOCK (AT,XT286,CONV,PS)
                                                                    ; DL = day in BCD, DH =       month in BCD, CL = year       in BCD
                                                                    ; CH = century (19h or 20h)
                                                                    ; Return: CMOS clock set
00000438 FB                                 sti
                                no_cmos_2:
                                            ; 12/12/2022
                                            ; cf=0
                                            ;clc
00000439 C3                                 retn

                                ; --------------------------------------------------------------------------

                                ; 08/08/2023 - Retro DOS v4.2 IO.SYS (optimization)
                                ; PCDOS 7.1 IBMBIO.COM - BIOSCODE:0440h
                                %if 1

                                ; CMOS Clock setting support routines used by MSCLOCK.
                                ; Warning!!! This code will be dynamically relocated by MSINIT.

                                daycnttoday:        ; proc near

                                ; entry: [daycnt] = number of days since 1-1-80
                                ;
                                ; return: ch - century in bcd
                                ;     cl - year in bcd
                                ;     dh - month in bcd
                                ;     dl - day in bcd

                                        ; 20/12/2023 - Retro DOS v5.0

                                        ; 08/08/2023 (ds:) (near proc)
                                        ; 16/10/2022 (cs:) (far proc)
0000043A FF36[8904]                     push    word [daycnt] ; save daycnt
0000043E 813E[8904]891C                 cmp     word [daycnt], 7305 ; (365*20+(20/4))
                                                                    ; # days from 1-1-1980 to 1-1-2000
00000444 7308                           jnb     short century20
```

```
                                        ;mov    byte [base_century], 19
                                        ;mov    byte [base_year], 80
                                        ; 08/08/2023
00000446 C706[8D04]1350                 mov     word [base_century], 5013h
0000044C EB0C                           jmp     short years
                                ; ---------------------------------------------------------------------------

                                century20:
                                        ;mov    byte [base_century], 20
                                        ;mov    byte [base_year], 0
                                        ; 08/08/2023
0000044E C706[8D04]1400                 mov     word [base_century], 20
00000454 812E[8904]891C                 sub     word [daycnt], 7305 ; (365*20+(20/4))
                                                                    ; adjust daycnt
                                years:
0000045A 31D2                           xor     dx, dx
0000045C A1[8904]                       mov     ax, [daycnt]
0000045F BBB505                         mov     bx, 1461        ; (366+365*3)
                                                                ; # of days in a Leap year block
00000462 F7F3                           div     bx              ; AX = # of leap block,       DX = daycnt
00000464 8916[8904]                     mov     [daycnt], dx    ; save daycnt left
00000468 B304                           mov     bl, 4
0000046A F6E3                           mul     bl              ; AX = # of years. Less       than 100
0000046C 0006[8E04]                     add     [base_year], al ; So, ah = 0. Adjust year
00000470 FF06[8904]                     inc     word [daycnt]  ; set daycnt to       1 base
                                        ; 08/08/2023
00000474 BB6E01                         mov     bx, 366
00000477 B90300                         mov     cx, 3
                                        ;cmp    word [daycnt], 366 ; daycnt=remainder of leap year
0000047A 391E[8904]                     cmp     [daycnt], bx ; 366
0000047E 7619                           jbe     short leapyear ; within 366+355+355+355 days.
00000480 FE06[8E04]                     inc     byte [base_year] ; if daycnt <= 366, then leap year
                                        ;sub    word [daycnt], 366 ; else daycnt--, base_year++ ;
00000484 291E[8904]                     sub     [daycnt], bx ; 366 ; 08/08/2023
                                        ;mov    cx, 3           ; And next three years are normal
                                        ; 08/08/2023
00000488 4B                             dec     bx ; 365
                                regularyear:        ; 20/12/2023
                                        ;cmp    word [daycnt], 365 ; for(i=1; i>3 or daycnt <=365; i++)
00000489 391E[8904]                     cmp     [daycnt], bx ; 365 ; 08/08/2023
0000048D 760F                           jbe     short yeardone ; {if (daycnt >       365)
0000048F FE06[8E04]                     inc     byte [base_year] ; { daycnt -=       365
                                        ;sub    word [daycnt], 365 ; }
00000493 291E[8904]                     sub     [daycnt], bx ; 365 ; 08/08/2023
00000497 E2F0                           loop    regularyear    ; }
                                                                ;
                                                                ; should never fall through loop
                                leapyear:
00000499 C606[9004]1D                   mov     byte [february], 29 ; 08/08/2023
                                        ;mov    byte [month_tab+1], 29 ; leap year.
                                                                ; change month table.
                                yeardone:
0000049E 31DB                           xor     bx, bx
000004A0 31D2                           xor     dx, dx
000004A2 A1[8904]                       mov     ax, [daycnt]
                                        ;mov    si, offset month_tab
000004A5 BE[8F04]                       mov     si, month_tab  ; 19/10/2022
                                        ;mov    cx, 12
                                        ; 08/08/2023
000004A8 B10C                           mov     cl, 12
                                months:
000004AA FEC3                           inc     bl
                                        ; 08/08/2023
000004AC 8A14                           mov     dl, [si]        ; PCDOS 7.1 IBMBIO.COM - BIOSCODE:04B7h
000004AE 39D0                           cmp     ax, dx          ; cmp daycnt for each month till fit
                                                                ; dh=0
000004B0 7605                           jbe     short month_done
000004B2 46                             inc     si              ; next month
000004B3 29D0                           sub     ax, dx          ; adjust daycnt
000004B5 E2F3                           loop    months          ;
                                                                ; should never fall through loop
                                month_done:
000004B7 C606[9004]1C                   mov     byte [february], 28 ; 08/08/2023
                                        ;mov    byte [month_tab+1], 28
                                                                ; restore month table value
000004BC 88DA                           mov     dl, bl
000004BE 8A36[8E04]                     mov     dh, [base_year]
000004C2 8A0E[8D04]                     mov     cl, [base_century] ; al=day,dl=month,dh=year,cl=cntry
000004C6 E81600                         call    bintobcd        ; convert "day"       to bcd
                                                                ; dl = bcd day,       al = month
000004C9 86D0                           xchg    dl, al
000004CB E81100                         call    bintobcd        ; dh = bcd month, al = year
000004CE 86F0                           xchg    dh, al
000004D0 E80C00                         call    bintobcd        ; cl = bcd year, al = century
000004D3 86C8                           xchg    cl, al
000004D5 E80700                         call    bintobcd        ; ch = bcd century
000004D8 88C5                           mov     ch, al
000004DA 8F06[8904]                     pop     word [daycnt] ; restore original value
000004DE C3                             retn

                                ;---------------------------------------------------------------------------

                                bintobcd:   ; proc near           ; real time clock support

                                ;convert a binary input in al (less than 63h or 99 decimal)
                                ;into a bcd value in al. ah destroyed.

000004DF D40A                           aam                     ; AH = AL/10, AL = AL MOD 10
000004E1 D510                           aad     10h               ; db 0D5h,10h
                                                                ; AL = (AH*10H)+AL, AH = 0
000004E3 C3                             retn
                                %endif

                                ;---------------------------------------------------------------------------

                                ; 15/10/2022

                                ;---------------------------------------------------------------------------
                                ; gettime reads date and time
                                ; and returns the following information:
                                ;
                                ;   es:[di]  =count of days since 1-1-80
                                ;   es:[di+2]=hours
                                ;   es:[di+3]=minutes
                                ;   es:[di+4]=seconds
                                ;   es:[di+5]=hundredths of seconds
                                ;
                                ;   NOTE: Any changes in this routine probably require corresponding
                                ;   changes in the version that is built with the power manager driver.
                                ;   See ptime.asm.
                                ;---------------------------------------------------------------------------

                                tim_read:                       ; 2C7h:435h = 70h:29A5h
000004E4 E84A00                         call    GetTickCnt
```

```
000004E7 8B36[8904]                     mov     si, [daycnt]

                                ; we now need to convert the time in tick to the time in 100th of
                                ; seconds. the relation between tick and seconds is:
                                ;
                                ;          65,536 seconds
                                ;         ----------------
                                ;         1,193,180 tick
                                ;
                                ; to get to 100th of second we need to multiply by 100. the equation is:
                                ;
                                ;   ticks from clock  * 65,536 * 100
                                ;      ---------------------------------- = time in 100th of seconds
                                ;           1,193,180
                                ;
                                ; fortunately this formula simplifies to:
                                ;
                                ;   ticks from clock * 5 * 65,536
                                ;      ---------------------------------- = time in 100th of seconds
                                ;           59,659
                                ;
                                ; the calculation is done by first multipling tick by 5. next we divide by
                                ; 59,659. in this division we multiply by 65,536 by shifting the dividend
                                ; my 16 bits to the left.
                                ;
                                ; start with ticks in cx:dx
                                ; multiply by 5

000004EB 89C8                           mov     ax, cx
000004ED 89D3                           mov     bx, dx          ; start with ticks in cx:dx
                                                                ; multiply by 5
000004EF D1E2                           shl     dx, 1
000004F1 D1D1                           rcl     cx, 1           ; times 2
000004F3 D1E2                           shl     dx, 1
000004F5 D1D1                           rcl     cx, 1           ; times 4
000004F7 01DA                           add     dx, bx
000004F9 11C8                           adc     ax, cx          ; times 5
000004FB 92                             xchg    ax, dx

                                ; now have ticks * 5 in    dx:ax
                                ; we now need to multiply by 65536 and divide by 59659 d.

000004FC B90BE9                         mov     cx, 59659       ; get divisor
000004FF F7F1                           div     cx              ; dx now has remainder
                                                                ; ax has high word of final quotient

                                        ; 08/08/2023 - Retro DOS v4.2 IO.SYS (optimization)
                                        ;mov    bx, ax          ; put high word         in safe place
00000501 93                             xchg    bx, ax
00000502 31C0                           xor     ax, ax          ; this is the multiply by 65536
00000504 F7F1                           div     cx              ; bx:ax now has time in 100th of seconds

                                ; rounding based on the    remainder may be added here
                                ; the result in    bx:ax is time in 1/100 second.

00000506 89DA                           mov     dx, bx
00000508 B9C800                         mov     cx, 200         ; extract 1/100's

                                ; division by 200 is necessary to ensure no overflow--max result
                                ; is number of seconds in a day/2 = 43200.

0000050B F7F1                           div     cx
0000050D 80FA64                         cmp     dl, 100         ; remainder over 100?
00000510 7203                           jb      short noadj
00000512 80EA64                         sub     dl, 100         ; keep 1/100's less than 100
                                noadj:
00000515 F5                             cmc                     ; if we subtracted 100,         carry is now set
00000516 88D3                           mov     bl, dl          ; save 1/100's

                                ; to compensate    for dividing by         200 instead of 100, we now multiply
                                ; by two, shifting a one in if the remainder had exceeded 100.

00000518 D1D0                           rcl     ax, 1
0000051A B200                           mov     dl, 0
0000051C D1D2                           rcl     dx, 1
                                        ;mov    cx, 60          ; divide out seconds
                                        ; 20/12/2023
0000051E B13C                           mov     cl, 60
00000520 F7F1                           div     cx
00000522 88D7                           mov     bh, dl          ; save the seconds
00000524 F6F1                           div     cl              ; break into hours and minutes
00000526 86C4                           xchg    al, ah

                                ; time is now in ax:bx (hours, minutes, seconds, 1/100 sec)

                                        ; 08/08/2023
                                        ;push   ax
                                        ;mov    ax, si          ; daycnt
00000528 96                             xchg    ax, si
00000529 AB                             stosw
                                        ;pop    ax
0000052A 96                             xchg    ax, si          ; al = hours, ah = minutes
0000052B AB                             stosw
0000052C 89D8                           mov     ax, bx
0000052E AB                             stosw
0000052F F8                             clc                     ; [es:di] = count of days since 1-1-80
                                                                ;   [es:di+2] = hours
                                                                ;   [es:di+3] = minutes
                                                                ;   [es:di+4] = seconds
                                                                ;   [es:di+5] = hundredths of seconds
00000530 C3                             retn

                                ; =============== S U B   R O U T I N E =======================================

                                ; 15/10/2022

                                ;---------------------------------------------------------------------------
                                ;
                                ; procedure : GetTickCnt
                                ;
                                ;           Returns the tick count in CX:DX. Takes care of DayCnt in case
                                ;           of rollover [except when power management driver is in use].
                                ;           Uses the following logic for updating Daycnt
                                ;
                                ;           if ( rollover ) {
                                ;                   if ( t_switch )
                                ;                           daycnt++ ;
                                ;                   else
                                ;                           daycnt += rollover ;
                                ;           }
                                ;
                                ; USES : AX
                                ;
                                ; RETURNS : CX:DX - tick count
```

```
                                ; MODIFIES : daycnt
                                ;
                                ;------------------------------------------------------------------------------

                                        ; 17/10/2022
                                GetTickCnt:
00000531 30E4                           xor     ah, ah
00000533 CD1A                           int     1Ah             ; CLOCK - GET TIME OF DAY
                                                                ; Return: CX:DX       = clock count
                                                                ; AL = 00h if clock was       read or written (via AH=0,1) since the previous
                                                                ; midnight
                                                                ; Otherwise, AL       > 0
                                        ; 20/12/2023
00000535 30E4                           xor     ah, ah
00000537 3826[8B04]                     cmp     byte [t_switch], ah ; 0
                                        ;cmp    byte [t_switch], 0 ; use old method ? (>0 is yes)
0000053B 7505                           jnz     short inc_case ; old method assumes that Int 1Ah returns rollover flag
                                        ;xor    ah, ah          ; new method assumes that Int 1Ah returns roll over count
                                                                ; and not flag
0000053D 0106[8904]                     add     [daycnt], ax
00000541 C3                             retn
                                ; ------------------------------------------------------------------------------

                                inc_case:
00000542 08C0                           or      al, al
00000544 7404                           jz      short no_rollover
00000546 FF06[8904]                     inc     word [daycnt]
                                no_rollover:
0000054A C3                             retn

                                ; ------------------------------------------------------------------------------
                                ; ------------------------------------------------------------------------------
                                ; 03/10/2023 - Retro DOS v5.0 (Modified PCDOS 7.1)
                                ; PCDOS 7.1 IBMBIO.COM - BIOSCODE:0556h

                                %if 1

0000054B 4641543132202020       fat_12_id:  db 'FAT12   '
00000553 4641543136202020       fat_16_id:  db 'FAT16   '
0000055B 4641543332202020       fat_32_id:  db 'FAT32   '
00000563 4E4F204E414D452020-    nul_vid:    db 'NO NAME    '
0000056C 2020

                                %endif

                                ;------------------------------------------------------------------------------
                                ; MSDISK.ASM - MSDOS 6.0 - 1991
                                ;------------------------------------------------------------------------------
                                ; 15/10/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 IO.SYS)
                                ; 09/03/2019 - Retro DOS v4.0

                                ; MSDISK.ASM - MSDOS 3.3 - 02/02/1988
                                ; 26/05/2018 - Retro DOS v3.0
                                ; 23/03/2018 - Retro DOS v2.0

                                ;error_unknown_media equ  7      ; for use in BUILD BPB call

                                ;struc BPB_TYPE
                                ;.SECSIZE:  resw 1
                                ;.SECALL:   resb 1
                                ;.RESNUM:   resw 1
                                ;.FATNUM:   resb 1
                                ;.DIRNUM:   resw 1
                                ;.SECNUM:   resw 1
                                ;.FATID:    resb 1
                                ;.FATSIZE:  resw 1
                                ;.SLIM:             resw 1
                                ;.HLIM:             resw 1
                                ;.HIDDEN:   resw 1
                                ;.size:
                                ;endstruc

                                ;--------------------------------------------------------------
                                ;   disk interface routines
                                ;--------------------------------------------------------------

                                ; device attribute bits:
                                ;   bit 6 - get/set map for logical drives and generic ioctl.

                                ;MAXERR             equ     5
                                ;MAX_HD_FMT_ERR     equ     2

                                ;LSTDRV     equ 504h

                                ; some floppies do not have changeline. as a result, media-check would
                                ; normally return i-don't-know, the dos would continually reread the fat and
                                ; discard cached data. we optimize this by implementing a logical door-latch:
                                ; it is physically impossible to change a disk in under 2 seconds. we retain
                                ; the time of the last successful disk operation and compare it with the current
                                ; time during media-check. if < 2 seconds and at least 1 timer tick has passed,
                                ; the we say no change. if > 2 seconds then we say i-don't-know. finally,
                                ; since we cannot trust the timer to be always available, we record the number
                                ; of media checks that have occurred when no apparent time has elapsed. while
                                ; this number is < a given threshold, we say no change. when it exceeds that
                                ; threshold, we say i-don't-know and reset the counter to 0. when we store
                                ; the time of last successful access, if we see that time has passed too,
                                ; we reset the counter.

                                accessmax   equ     5

                                ; due to various bogosities, we need to continually adjust what the head
                                ; settle time is.  the following algorithm is used:
                                ;
                                ;   get the current head settle value.
                                ;   if it is 0, then
                                ;   set slow = 15
                                ;   else
                                ;   set slow = value
                                ;   ...
                                ;*********************************************
                                ;************ old algorithm ******************
                                ;*   if we are seeking and writing then
                                ;*   use slow
                                ;*   else
                                ;*   use fast
                                ;*********************************************
                                ;*********** ibm's requested logic ***********
                                ;   if we are seeking and writing and not on an at then
                                ;   use slow
                                ;   else
                                ;   use fast
                                ;   ...
                                ;   restore current head settle value
                                ;
```

```
                                ;
                                ;---------------------------------------
                                multrk_on   equ     10000000b       ;user spcified mutitrack=on, or system turns
                                                                    ; it on after handling config.sys file as a
                                                                    ; default value, if multrk_flag = multrk_off1.
                                multrk_off1 equ     00000000b       ;initial value. no "multitrack=" command entered.
                                multrk_off2 equ     00000001b       ;user specified multitrack=off.

                                ; close data segment, open Bios_Code segment

                                ; 15/10/2022

                                ; BIOSCODE:04A2h (MSDOS 6.21, IO.SYS)

                                ;--------------------------------------------------------------------
                                ;   command jump table
                                ;--------------------------------------------------------------------

0000056E 00                             db 0

                                ; 11/12/2022
                                %if 0

                                dsktbl:             db 26                   ; 2C7h:4A2h = 70h:2A12h
                                                                    ; ((dtbl_siz-1)/2) ; this is the size of the table ; 26
                                                    dw 1742h                ; dsk_init
                                                    dw 4EBh                 ; media_chk
                                                    dw 592h                 ; get_bpb
                                                    dw 0D5h                 ; bc_cmderr
                                                    dw 857h                 ; dsk_read
                                                    dw 83Dh                 ; x_bus_exit
                                                    dw 558h                 ; ret_carry_clear
                                                    dw 558h                 ; ret_carry_clear
                                                    dw 849h                 ; dsk_writ
                                                    dw 841h                 ; dsk_writv
                                                    dw 558h                 ; ret_carry_clear
                                                    dw 558h                 ; ret_carry_clear
                                                    dw 0D5h                 ; bc_cmderr
                                                    dw 80Ah                 ; dsk_open
                                                    dw 81Ah                 ; dsk_close
                                                    dw 831h                 ; dsk_rem
                                                    dw 558h                 ; ret_carry_clear
                                                    dw 558h                 ; ret_carry_clear
                                                    dw 558h                 ; ret_carry_clear
                                                    dw 0C6Bh                ; do_generic_ioctl
                                                    dw 558h                 ; ret_carry_clear
                                                    dw 558h                 ; ret_carry_clear
                                                    dw 558h                 ; ret_carry_clear
                                                    dw 1124h                ; ioctl_getown
                                                    dw 1142h                ; ioctl_setown
                                                    dw 129Ah                ; ioctl_support_query

                                ;dtbl_siz equ $-dsktbl

                                %endif

                                ; 21/12/2023 - Retro DOS v5.0 (Modified PCDOS 7.1)
                                ; PCDOS 7.1 IBMBIO.COM - BIOSCODE:0579h

                                                    ; 21/12/2023 - Retro DOS v5.0
                                                    ; 11/12/2022
0000056F 1A                     dsktbl:             db (dtbl_siz-1)/2     ; 26 ; this is the size of the table
00000570 [4A1A]                                     dw dsk_init
00000572 [B805]                                     dw media_chk
00000574 [5706]                                     dw get_bpb
                                                    ;dw bc_cmderr
00000576 [4A0E]                                     dw ioctl_input ; PCDOS 7 ; 21/12/2023
00000578 [7209]                                     dw dsk_read
0000057A [5809]                                     dw x_bus_exit
0000057C [2206]                                     dw ret_carry_clear
0000057E [2206]                                     dw ret_carry_clear
00000580 [6409]                                     dw dsk_writ
00000582 [5C09]                                     dw dsk_writv
00000584 [2206]                                     dw ret_carry_clear
00000586 [2206]                                     dw ret_carry_clear
                                                    ;dw bc_cmderr
00000588 [F70D]                                     dw ioctl_output ; PCDOS 7 ; 21/12/2023
0000058A [2909]                                     dw dsk_open
0000058C [3809]                                     dw dsk_close
0000058E [4E09]                                     dw dsk_rem
00000590 [2206]                                     dw ret_carry_clear
00000592 [2206]                                     dw ret_carry_clear
00000594 [2206]                                     dw ret_carry_clear
00000596 [CC0E]                                     dw do_generic_ioctl
00000598 [2206]                                     dw ret_carry_clear
0000059A [2206]                                     dw ret_carry_clear
0000059C [2206]                                     dw ret_carry_clear
0000059E [A713]                                     dw ioctl_getown
000005A0 [C413]                                     dw ioctl_setown
000005A2 [1C15]                                     dw ioctl_support_query

                                dtbl_siz equ $-dsktbl

                                ; ============== S U B   R O U T I N E =======================================

                                ; ---------------------------------------------------------------------------
                                ; setdrive scans through the data structure of bdss, and returns a pointer to
                                ; the one that belongs to the drive specified. carry is set if none exists
                                ; for the drive. Pointer is returned in es:[di]
                                ;
                                ;  AL contains the logical drive number.
                                ; ---------------------------------------------------------------------------

                                SetDrive:
                                                    ;les    di, dword ptr ds:start_bds ; Point es:di to first bds
000005A4 C43E[1901]                                 les     di, [start_bds] ; 19/10/2022
                                X_Scan_Loop:
000005A8 26384505                                   cmp     [es:di+5], al
000005AC 7409                                       jz      short X_SetDrv
000005AE 26C43D                                     les     di, [es:di]     ; [es:di+BDS.link] ; Go      to next bds
000005B1 83FFFF                                     cmp     di, 0FFFFh
000005B4 75F2                                       jnz     short X_Scan_Loop
000005B6 F9                                         stc
                                X_SetDrv:
000005B7 C3                                         retn

                                ; ---------------------------------------------------------------------------

                                ; 15/10/2022

                                    ; 21/12/2023 - Retro DOS v5.0 (Modified PCDOS 7.1)
                                    ; PCDOS 7.1 IBMBIO.COM - BIOSCODE:05C2h

                                ; ---------------------------------------------------------------------------
```

```
12408                                   ; if id is f9, have a 96tpi disk else
12409                                   ; if bit 2 is 0 then media is not removable and could not have changed
12410                                   ;  otherwise if within 2 secs of last disk operation media could not
12411                                   ;    have changed, otherwise dont know if media has changed
12412                                   ; ---------------------------------------------------------------------
12413
12414                                   media_chk:                      ; 2C7h:4EBh = 70h:2A5Bh
12415 000005B8 E8E9FF                              call    SetDrive
12416 000005BB BE0100                            mov     si, 1
12417                                             ; 21/12/2023
12418 000005BE 26F6454001                        test    byte [es:di+40h], 1
12419                                             ;test   byte [es:di+24h], 1 ; [es:di+BDS.flags+1]
12420                                                                     ; fchanged_by_format
12421 000005C3 7415                              jz      short WeAreNotFakingIt
12422                                             ; 21/12/2023
12423 000005C5 26806540FE                        and     byte [es:di+40h], 0FEh
12424                                             ; 12/12/2022
12425                                             ;and    byte [es:di+24h], 0FEh ; ~fchanged_by_format
12426                                             ;;and   word [es:di+23h], 0FEFFh ; [es:di+BDS.flags]
12427                                                                     ; ~fchanged_by_format ;       reset flag
12428 000005CA C606[1E01]FF                      mov     byte [tim_drv], 0FFh ; -1
12429                                                                     ; Ensure that we ask the rom if media has changed
12430                                             ; 21/12/2023
12431 000005CF 26F6453F01                        test    byte [es:di+3Fh], 1
12432                                             ;test   byte [es:di+23h], 1 ; [es:di+BDS.flags]
12433                                                                     ; fnon_removable
12434 000005D4 740B                              jz      short wehaveafloppy
12435                                             ;mov    si, 0FFFFh      ; Indicate media changed
12436                                             ; 11/08/2023
12437 000005D6 F7DE                              neg     si              ; PCDOS 7.1 IBMBIO.COM - BIOSCODE:05E0h
12438 000005D8 EB2B                              jmp     short Media_Done ; Media_Done
12439                                   ; ---------------------------------------------------------------------------
12440
12441                                   WeAreNotFakingIt:
12442                                             ; 21/12/2023
12443 000005DA 26F6453F01                        test    byte [es:di+3Fh], 1
12444                                             ;test   byte [es:di+BDS.flags], fnon_removable
12445                                             ;test   byte [es:di+23h], 1
12446 000005DF 7524                              jnz     short Media_Done
12447                                   wehaveafloppy:
12448                                             ;xor    si, si ; 0      ; Presume "I don't know"
12449                                             ; 11/08/2023
12450 000005E1 4E                                dec     si ; 0          ; PCDOS 7.1 IBMBIO.COM - BIOSCODE:05EBh
12451
12452                                             ; If we have a floppy with changeline support, we ask the ROM
12453                                             ; to determine if media has changed. We do not perform the
12454                                             ; 2 second check for these drives.
12455
12456 000005E2 803E[7700]00                      cmp     byte [fhave96], 0      ; Do we have changeline       support?
12457 000005E7 740A                              jz      short mChk_NoChangeLine       ; Brif not
12458 000005E9 E83615                            call    mediacheck      ;  Call into removable routine
12459 000005EC 7236                              jb      short err_exitj
12460 000005EE E89A16                            call    haschange
12461 000005F1 7512                              jnz     short Media_Done
12462                                   mChk_NoChangeLine:
12463                                             ; If we come here, we have a floppy with no changeline support
12464
12465 000005F3 BE0100                            mov     si, 1           ; Presume no change
12466 000005F6 A0[1E01]                          mov     al, [tim_drv]   ; Last drive accessed
12467 000005F9 263A4504                          cmp     al, [es:di+4]   ; [es:di+BDS.drivenum]
12468                                                                     ; Is drive of last access the same?
12469 000005FD 7505                              jnz     short Media_Unk         ; No, then "i don't know"
12470 000005FF E82800                            call    Check_Time_Of_Access
12471 00000602 EB01                              jmp     short Media_Done
12472                                   ; ---------------------------------------------------------------------------
12473
12474                                   Media_Unk:
12475 00000604 4E                                dec     si              ; 0 ; Return "I don't know"
12476
12477                                             ; SI now contains the correct value for media change.
12478                                             ; Clean up the left overs
12479                                   Media_Done:
12480                                             ; 19/10/2022
12481 00000605 06                                push    es
12482 00000606 C41E[1200]                        les     bx, [ptrsav]
12483 0000060A 2689770E                          mov     [es:bx+0Eh], si         ; [es:bx+trans]
12484 0000060E 07                                pop     es
12485 0000060F 09F6                              or      si, si
12486 00000611 790F                              jns     short ret_carry_clear ;       volidok
12487 00000613 803E[7700]00                      cmp     byte [fhave96], 0
12488 00000618 7403                              jz      short mChk1_NoChangeLine ; Brif      no changeline support
12489 0000061A E80016                            call    media_set_vid
12490                                   mChk1_NoChangeLine:
12491 0000061D C606[1E01]FF                      mov     byte [tim_drv], 0FFh ; -1
12492                                                                     ; Make sure we ask rom for media check
12493                                   ret_carry_clear:
12494 00000622 F8                                clc                     ; volidok
12495 00000623 C3                                retn
12496                                   ; ---------------------------------------------------------------------------
12497
12498                                   err_exitj:
12499 00000624 E88207                            call    maperror        ; guaranteed to       set carry
12500                                   ret81:
12501 00000627 B481                              mov     ah, 81h         ; return error status
12502 00000629 C3                                retn                    ; return with carry set
12503
12504                                   ; =============== S U B   R O U T I N E =======================================
12505
12506                                   ; ---------------------------------------------------------------------------
12507                                   ; perform a check on the time passed since the last access for this physical
12508                                   ; drive.
12509                                   ; we are accessing the same drive. if the time of last successful access was
12510                                   ; less than 2 seconds ago, then we may presume that the disk was not changed.
12511                                   ; returns in si:
12512                                   ;   0 - if time of last access was >= 2 seconds
12513                                   ;   1 - if time was < 2 seconds (i.e no media change assumed)
12514                                   ; registers affected ax,cx,dx, flags.
12515                                   ;
12516                                   ;   assume es:di -> bds, ds->Bios_Data
12517                                   ; ---------------------------------------------------------------------------
12518
12519                                             ; 21/12/2023 - Retro DOS v5.0 IBMBIO.COM
12520                                             ; 19/10/2022
12521                                   Check_Time_Of_Access:
12522 0000062A BE0100                            mov     si, 1           ; presume no change.
12523 0000062D E801FF                            call    GetTickCnt      ; cx:dx is the elapsed time
12524                                             ; 21/12/2023
12525 00000630 268B4579                          mov     ax, [es:di+79h]
12526                                             ;mov    ax, [es:di+47h]       ; [es:di+BDS.tim_lo]
12527                                                                     ; get stored time
12528 00000634 29C2                              sub     dx, ax
12529                                             ; 21/12/2023
12530 00000636 268B457B                          mov     ax, [es:di+7Bh]
12531                                             ;mov    ax, [es:di+49h]       ; [es:di+BDS.tim_hi]
```

```
12532 0000063A 19C1                             sbb     cx, ax
12533                                           ; 11/08/2023
12534                                           ; PCDOS 7.1 IBMBIO.COM - BIOSCODE:0646h
12535                                           ;mov    al, [accesscount]
12536 0000063C 7515                             jnz     short timecheck_unk ; cx<>0 => >1 hour
12537 0000063E 09D2                             or      dx, dx          ; time must pass
12538 00000640 750C                             jnz     short timepassed ; yes, examine max value
12539                                           ; 11/08/2023
12540                                           ;inc    al
12541                                           ;cmp    al, 5
12542                                           ;;inc   byte [accesscount]
12543                                           ;;cmp   byte [accesscount], 5
12544                                           ;                       ; if count is less than threshold, ok
12545                                           ;jb     short timecheck_ret
12546                                           ;;dec   byte [accesscount] ; don't let the count wrap
12547                                           ; 11/08/2023
12548                                           ;dec    al
12549                                           ;jmp    short timecheck_unk ; "i don't know" if media changed
12550                                           ; 11/08/2023
12551 00000642 803E[1D01]04                     cmp     byte [accesscount], 4
12552 00000647 730A                             jnb     short timecheck_unk
12553 00000649 FE06[1D01]                       inc     byte [accesscount]
12554 0000064D C3                               retn
12555
12556                                   ; ---------------------------------------------------------------------------
12557
12558                                   timepassed:
12559 0000064E 83FA24                             cmp     dx, 36          ; 18*2 ; 18.2 tics per second.
12560                                                                     ; min elapsed time? (2 seconds)
12561 00000651 7601                               jbe     short timecheck_ret ; yes, presume no change
12562
12563                                             ; everything indicates that we do not know what has happened.
12564                                   timecheck_unk:
12565 00000653 4E                                 dec     si              ; presume i don't know
12566                                   timecheck_ret:
12567                                             ; 11/08/2023
12568                                             ;mov    [accesscount], al
12569 00000654 C3                                 retn
12570
12571                                   ; ---------------------------------------------------------------------------
12572                                   ; 15/10/2022
12573                                   Err_Exitj2:
12574 00000655 EBCD                               jmp     short err_exitj
12575
12576                                   ; ---------------------------------------------------------------------------
12577
12578                                   ; 15/10/2022
12579
12580                                   ; ============================================================================
12581                                   ; Build a valid bpb for the disk in the drive.
12582                                   ; ============================================================================
12583
12584                                             ; 21/12/2023 - Retro DOS v5.0 IBMBIO.COM
12585                                             ; 19/10/2022
12586                                   get_bpb:                          ; 2C7h:592h = 70h:2B02h
12587 00000657 268A25                             mov     ah, [es:di]     ; get fat id byte read by dos
12588 0000065A E847FF                             call    SetDrive        ; get the correct bds for the drive
12589                                             ; 21/12/2023
12590 0000065D 26F6453F01                         test    byte [es:di+3Fh], 1
12591                                             ;test   byte [es:di+23h], 1 ; [es:di+BDS.flags]
12592                                                                     ; fnon_removable
12593 00000662 7523                               jnz     short already_gotbpb ; no need to build     for fixed disks
12594
12595                                             ; let's set the default value for volid,vol_serial,
12596                                             ; filesys_id in bds table
12597
12598 00000664 E83600                             call    clear_ids
12599                                             ;mov    ds:set_id_flag,       1 ; indicate to       set system id in bds
12600 00000667 C606[9B04]01                       mov     byte [set_id_flag], 1
12601 0000066C E86700                             call    GetBp           ; build a bpb if necessary
12602 0000066F 72B6                               jb      short ret81
12603                                             ;cmp    ds:set_id_flag,       2 ; already, volume_label set from boot
12604 00000671 803E[9B04]02                       cmp     byte [set_id_flag], 2
12605                                             ;mov    ds:set_id_flag,       0 ; record to bds table?
12606 00000676 C606[9B04]00                       mov     byte [set_id_flag], 0
12607 0000067B 740A                               jz      short already_gotbpb ; do not set it again from     root dir
12608                                                                     ; otherwise, conventional boot record
12609                                             ;cmp    ds:fhave96, 0  ; do we have changeline       support?
12610 0000067D 803E[7700]00                       cmp     byte [fhave96], 0
12611 00000682 7403                               jz      short already_gotbpb ; brif not
12612 00000684 E80A16                             call    set_volume_id
12613                                   already_gotbpb:
12614 00000687 83C706                             add     di, 6           ; BDS.BPB
12615                                                                     ; return the bpb from the current bds
12616
12617                                   ;           fall into setptrsav, es:di -> result
12618
12619                                   ; ---------------------------------------------------------------------------
12620
12621                                   ; 15/10/2022
12622
12623                                   ; ============================================================================
12624                                   ;Setptrsav is also jumped to from dsk_init (msbio2.asm). In both cases, the
12625                                   ;pointer to be returned is in es:di. We were incorrectly returning ds:di.
12626                                   ;Note that this works in most cases because most pointers are in Bios_Data.
12627                                   ;It fails, for instance, when we install an external drive using driver.sys
12628                                   ;because then the BDS segment is no longer Bios_Data.
12629                                   ;NB: It is fine to corrupt cx because this is not a return value and anyway
12630                                   ;this returns to Chardev_entry (msbio1.asm) where all registers are
12631                                   ;restored before returning to the caller.
12632                                   ; ============================================================================
12633
12634                                   ; 21/12/2023
12635                                   %if 0
12636                                             ; 19/10/2022
12637                                   SetPtrSav: ; return point for dsk_init
12638                                             mov     cx, es          ; save es
12639                                             ;les    bx, ds:ptrsav
12640                                             les     bx, [ptrsav]
12641                                             mov     [es:bx+0Dh], ah       ; [es:bx+media]
12642                                             mov     [es:bx+12h], di       ; [es:bx+count]
12643                                             mov     [es:bx+14h], cx       ; [es:bx+count+2]
12644                                             clc
12645                                             retn
12646                                   %endif
12647                                             ; 21/12/2023 - Retro DOS v5.0 (Modified PCDOS 7.1)
12648                                             ; PCDOS 7.1 IBMBIO.COM - BIOSCODE:0698h
12649                                   SetPtrSav:
12650                                             ; return point for dsk_init
12651 0000068A 1E                                 push    ds
12652                                             ;lds    bx, ds:ptrsav
12653 0000068B C51E[1200]                         lds     bx, [ptrsav]
12654 0000068F 886F0D                             mov     [bx+0Dh], ah   ; [bx+media]
12655 00000692 897F12                             mov     [bx+12h], di   ; [bx+count]
```

```
12656 00000695 8C4714                          mov     [bx+14h], es   ; [bx+count+2]
12657 00000698 1E                              push    ds
12658 00000699 07                              pop     es
12659 0000069A 1F                              pop     ds
12660 0000069B F8                              clc
12661 0000069C C3                              retn
12662
12663                                  ; ============== S U B   R O U T I N E =======================================
12664
12665                                  ; 15/10/2022
12666
12667                                  ; ------------------------------------------------------
12668                                  ; clear ids in bds table. only applied for floppies.
12669                                  ;input:  es:di -> bds table
12670                                  ;   assumes ds: -> Bios_Data
12671                                  ;output: volid set to "NO NAME    "
12672                                  ;    vol_serial set to 0.
12673                                  ;    filesys_id set to "FAT12   " or "FAT16   "
12674                                  ;      depending on the flag fatsize in bds.
12675                                  ;
12676                                  ;   trashes si, cx
12677                                  ; ------------------------------------------------------
12678
12679                                  ;size_of_EXT_BOOT_VOL_LABEL equ 11
12680                                  ;size_of_EXT_SYSTEM_ID equ 8
12681
12682                                  ; 11/09/2023
12683                                  ; 14/08/2023
12684                                  ;BDS.fatsiz equ 1Fh
12685                                  ; 21/12/2023
12686                                  ;BDS.fatsiz equ 59
12687
12688                                              ; 22/12/2023
12689                                              ; 21/12/2023 - Retro DOS v5.0 (Modified PCDOS 7.1)
12690                                  clear_ids:
12691                                              ;mov    al, [es:di+1Fh] ; mov al,[es:di+BDS.fatsiz]
12692                                              ; 21/12/2023 - Retro DOS v5.0 (PCDOS 7.1 IBMBIO.COM BugFix)
12693 0000069D 268A5D3B                          mov     bl, [es:di+3Bh] ; mov bl,[es:di+BDS.fatsiz]; *+
12694                                  clear_ids_x:
12695                                              ; 21/12/2023
12696                                              ; (PCDOS 7.1 IBMBIO.COM - BIOSCODE:06ABh)
12697                                              ; 11/09/2023
12698                                              ; (MSDOS 5.0 IO.SYS - BIOSCODE:05D9h)
12699 000006A1 57                               push    di
12700 000006A2 31C9                              xor     cx, cx         ; no serial number
12701                                            ; 21/12/2023
12702 000006A4 26898D8900                        mov     [es:di+89h], cx        ; [es:di+BDS.vol_serial]
12703 000006A9 26898D8B00                        mov     [es:di+8Bh], cx        ; [es:di+BDS.vol_serial+2]
12704                                            ;mov    [es:di+57h], cx        ; [es:di+BDS.vol_serial]
12705                                            ;mov    [es:di+59h], cx        ; [es:di+BDS.vol_serial+2]
12706
12707                                            ; BUGBUG - there's a lot in common here and with
12708                                            ; mov_media_ids.. see if we can save some space by
12709                                            ; merging them... jgl
12710
12711                                            ;mov    cx, 11         ; size_of_EXT_BOOT_VOL_LABEL
12712                                            ; 10/12/2022
12713 000006AE B10B                            mov     cl, 11 ; cx = 11
12714
12715                                            ;;mov   si, offset vol_no_name ; "NO NAME    "
12716                                            ;mov    si, vol_no_name        ; 19/10/2022
12717                                            ; 22/12/2023
12718                                            ;mov    si, offset nul_vid ; "NO NAME    "
12719 000006B0 BE[6305]                         mov     si, nul_vid
12720
12721                                            ; 21/12/2023
12722 000006B3 83C77D                          add     di, 125
12723                                          ;add    di, 75         ; BDS.volid
12724
12725                                          ;rep movsb
12726                                          ; 21/12/2023
12727                                          ;rep movs byte ptr es:[di], byte ptr cs:[si] ; cs rep movsb
12728                                          ; 26/12/2023
12729                                          ;cs     ; vol_no_name is in BIOSCODE segment
12730                                          ;rep movsb
12731 000006B6 F3                             rep
12732 000006B7 2E                             cs
12733 000006B8 A4                             movsb
12734
12735                                          ; 11/09/2023 (BugFix, DI is not start addr of BDS structure here)
12736                                          ;;test  byte [es:di+BDS.fatsiz], fbig
12737                                          ; (MSDOS 5.0 IO.SYS - BIOSCODE:05EFh)
12738                                          ;test   byte [es:di+1Fh], 40h
12739                                          ; 21/12/2023 - Retro DOS v5.0
12740                                          ; (PCDOS 7.1 IBMBIO.COM - BIOSCODE:06C3h)
12741                                          ;test   byte [es:di+59], 20h
12742                                                  ; (here, es:di points to the BDS offset +136)
12743                                                  ; purpose: test byte [es:di+BDS.fatsiz], fbigbig
12744                                                  ; applied: test byte [es:BDS.fatsiz+136], fbigbig -BUG!-
12745
12746                                                  ; (PCDOS 7.1 BUG note: 26/06/2023 - Erdogan Tan)
12747                                                  ;; ! NOTE - 11/08/2023 - Erdogan Tan (Retro DOS v4.2 IO.SYS bugfix)
12748                                                  ; Microsoft/IBM code has a bug here because the BDS's
12749                                                  ; .volid and .filesys_id fields will be reset
12750                                                  ; (to their default text) according to 'BDS.fatsiz' flags
12751                                                  ; at the BDS offset 59 but current (this) code checks flags
12752                                                  ; at ES:DI+59 while DI points the BDS offset 136!? ; (PCDOS 7.1)
12753                                                  ;; at the BDS offset 31 but current (this) code checks flags
12754                                                  ;; at ES:DI+31 while DI points the BDS offset 86!? ; (MSDOS 6.22)
12755                                                  ;
12756                                                  ; Correct Code:
12757                                                  ; ;test byte [ES:59],20h or [ES:BDS.fatsiz],fbigbig  ; (PCDOS 7.1)
12758                                                  ; ;;test byte [ES:31],40h or [ES:BDS.fatsiz],fbig  ; (MSDOS 6.22)
12759                                                  ; 11/09/2023
12760                                                  ; (before 'rep movsb') 'mov al,[es:di+BDS.Fatsiz]' and then
12761                                                  ; (after 'rep movsb') 'test al,fbig' (AL is free/proper to use here)
12762                                                  ;
12763                                                  ; Same BUG is existing in MSDOS 6.22 IO.SYS - BIOSCODE:05EFh
12764                                                  ; and in Windows ME IO.SYS - BIOSCODE:0E1Ah as 'test byte [es:di+59],20h'
12765
12766                                                  ;
12767                                                  ; (Why this bug did not affect MSDOS and PCDOS 7.x applications:
12768                                                  ; 'clear_ids' is used for floppy disks only and the default
12769                                                  ; option of 'clear_ids' is FAT12 volid and filesys_id text
12770                                                  ; when the flag bit has wrong value for FAT16/40h or FAT32/20h.)
12771
12772                                          ; 21/12/2023 - Retro DOS v5.0
12773                                          ;mov    si, offset fat_32_id ; "FAT32   "
12774 000006B9 BE[5B05]                       mov     si, fat_32_id
12775
12776                                          ; 21/12/2023
12777                                          ; BugFix (of the PCDOS 7.1 IBMBIO.COM BUG) ; *+
12778                                          ;test   bl, fbigbig ; FAT32 flag
12779 000006BC F6C320                         test    bl, 20h ; * ; BL = [es:BDS.fatsiz] = [es:59]
```

```
12780 000006BF 750B                             jnz     short ci_bigfat
12781
12782                                           ;mov    si, offset fat_16_id ; "FAT16 "
12783 000006C1 BE[5305]                         mov     si, fat_16_id  ; 19/10/2022
12784
12785                                           ; 21/12/2023
12786                                           ; !BUG! (PCDOS 7.1 IBMBIO.COM BIOSCODE:06CDh)
12787                                           ;test  byte [es:di+59], 40h ; [es:di+BDS.fatsiz], fbig
12788                                           ; BugFix ; *+
12789                                           ;test  bl, fbig ; FAT16 flag
12790 000006C4 F6C340                          test    bl, 40h ; * ; Retro DOS v5.0
12791                                           ;;test al, 40h ; * ; Retro DOS v4.2
12792 000006C7 7503                             jnz     short ci_bigfat
12793
12794                                           ;mov    si, offset fat_12_id ; "FAT12 "
12795 000006C9 BE[4B05]                         mov     si, fat_12_id  ; 19/10/2022
12796                                   ci_bigfat:
12797                                           ;mov    cx, 8          ; size_of_EXT_SYSTEM_ID
12798                                           ; 10/12/2022
12799 000006CC B108                             mov     cl, 8 ; cx = 8
12800 000006CE 83C705                           add     di, 5          ; (BDS.filesys_id-BDS.volid)-size_of_EXT_BOOT_VOL_LABEL
12801                                                                  ; filesys_id field
12802                                           ;rep movsb
12803                                           ; 21/12/2023 - Retro DOS v5.0
12804                                           ;rep movs byte ptr es:[di], byte ptr cs:[si] ; 0F3h,2Eh,0A4h
12805                                           ; 26/12/2023
12806                                           ;cs     ; fat32_id, fat16_id and fat12_id are in BIOSCODE segment
12807                                           ;rep movsb
12808 000006D1 F3                              rep
12809 000006D2 2E                              cs
12810 000006D3 A4                              movsb
12811
12812 000006D4 5F                               pop     di             ; restore bds pointer
12813                                   getret_exit:            ; 21/12/2023
12814 000006D5 C3                               retn
12815
12816                                   ; =============== S U B   R O U T I N E =======================================
12817
12818                                   ; 15/10/2022
12819
12820                                   ; ---------------------------------------------------------------------------
12821                                   ;   getbp - return bpb from the drive specified by the bds.
12822                                   ;       if the return_fake_bpb flag is set, then it does nothing.
12823                                   ;       note that we never come here for fixed disks.
12824                                   ;       for all other cases,
12825                                   ;         - it reads boot sector to pull out the bpb
12826                                   ;         - if no valid bpb is found, it then reads the fat sector,
12827                                   ;           to get the fat id byte to build the bpb from there.
12828                                   ;
12829                                   ;   inputs: es:di point to correct bds.
12830                                   ;
12831                                   ;   outputs:       fills in bpb in current bds if valid bpb or fat id on disk.
12832                                   ;           carry set, and al=7 if invalid disk.
12833                                   ;           carry set and error code in al if other error.
12834                                   ;           if failed to recognize the boot record, then will set the
12835                                   ;           set_id_flag to 0.
12836                                   ;           this routine will only work for a floppy diskette.
12837                                   ;                for a fixed disk, it will just return.
12838                                   ;
12839                                   ;   ****** Note: getbp is a clone of getbp which uses the newer
12840                                   ;     segment definitions. It should be migrated towards.
12841                                   ;      now es:di has the bds, ds: has Bios_Data
12842                                   ; ---------------------------------------------------------------------------
12843
12844                                           ; 29/12/2023
12845                                           ; 21/12/2023 - Retro DOS v5.0 (Modified PCDOS 7.1)
12846                                   GetBp:
12847                                           ; if returning fake bpb then return bpb as is.
12848                                           ; 21/12/2023
12849 000006D6 26F6453F05                       test    byte [es:di+3Fh], 5 ; PCDOS 7.1
12850                                           ;test  byte [es:di+BDS.flags], return_fake_bpb|fnon_removable
12851                                           ;test  byte [es:di+23h], 5 ; MSDOS 6.22 (& MSDOS 5.0)
12852                                           ;jz     short getbp1   ; getbp1
12853                                           ;jmp    getret_exit
12854                                           ; 21/12/2023
12855 000006DB 75F8                            jnz     short getret_exit
12856                                   ; ---------------------------------------------------------------------------
12857                                   getbp1:
12858 000006DD 51                               push    cx
12859 000006DE 52                               push    dx
12860 000006DF 53                               push    bx
12861
12862                                           ; attempt to read in boot sector and determine bpb.
12863                                           ; we assume that the 2.x and greater dos disks all
12864                                           ; have a valid boot sector.
12865
12866 000006E0 E8CF00                           call    readbootsec
12867 000006E3 720A                             jb      short getbp_err_ret_brdg ; carry set if there was error.
12868 000006E5 09DB                             or      bx, bx         ; bx is 0 if boot sector is valid.
12869 000006E7 7509                             jnz     short dofatbpb
12870 000006E9 E81401                           call    movbpb         ; move bpb into       registers
12871                                           ;jmp    short Has1
12872                                           ; 21/12/2023 - Retro DOS v5.0 (PCDOS 7.1) IBMBIO.COM
12873 000006EC E9B500                           jmp     getret
12874                                   ; ---------------------------------------------------------------------------
12875
12876                                   getbp_err_ret_brdg:
12877 000006EF E9B600                           jmp     getbp_err_ret
12878                                   ; ---------------------------------------------------------------------------
12879
12880                                           ; we have a 1.x diskette. In this case read in the fat ID byte
12881                                           ; and fill in bpb from there.
12882                                   dofatbpb:
12883 000006F2 E8B401                           call    readfat        ; puts media descriptor      byte in ah
12884 000006F5 72F8                             jb      short getbp_err_ret_brdg
12885                                           ;cmp    ds:fhave96, 0  ;  changeline support available?
12886 000006F7 803E[7700]00                     cmp     byte [fhave96], 0 ; 19/10/2022
12887 000006FC 7403                             jz      short bpb_nochangeline ; brif not
12888 000006FE E83115                           call    hidensity      ; may not return! May add sp, 2      and
12889                                                                  ; jump to has1!!!!!! or      has720K
12890                                   bpb_nochangeline:         ; test for a valid 3.5" medium
12891                                           ; 21/12/2023 - Retro DOS v5.0
12892 00000701 26807D3E02                       cmp     byte [es:di+3Eh], 2
12893                                           ;cmp    byte [es:di+22h], 2 ; [es:di+BDS.formfactor]
12894                                                                  ; ffSmall
12895 00000706 7512                             jnz     short is_floppy
12896 00000708 80FCF9                           cmp     ah, 0F9h       ; is it a valid fat id byte for       3.5" ?
12897 0000070B 7512                             jnz     short got_unknown_medium
12898                                   Has720K:
12899                                           ; 21/12/2023
12900                                           ;;mov   bx, offset sm92 ; pointer to correct bpb
12901                                           ;mov    bx, sm92       ; 19/10/2022
12902
12903                                           ; es points to segment of bds. the following should be modified
```

```
                                        ; to get spf,csec,spau,spt correctly. it had been wrong if
                                        ; driver.sys is loaded since the bds is inside the driver.sys.

                                        ; 21/12/2023
                                        ;; 10/12/2022
                                        ;;mov   al, [bx+0]     ; [bx+bpbtype.spf]
                                        ;; 21/12/2022
                                        ;mov    al, [bx]
                                        ;mov    cx, [bx+3]     ; [bx+bpbtype.csec]
                                        ;mov    dx, [bx+5]     ; [bx+bpbtype.spau]
                                        ;mov    bx, [bx+1]     ; [bx+bpbtype.spt]
                                        ;; 19/10/2022 - Temporary !
                                        ;;db    8Ah, 87h, 0, 0 ; mov al, [bx+0]
                                        ;;db    8Bh, 8Fh, 3, 0 ; mov cx, [bx+3]
                                        ;;db    8Bh, 97h, 5, 0 ; mov dx, [bx+5]
                                        ;;db    8Bh, 9Fh, 1, 0 ; mov bx, [bx+1]

                                        ; 21/12/2023 - Retro DOS v5.0
0000070D B003                           mov     al, 3          ; bpbtype.sbf = 3
0000070F B9A005                         mov     cx, 1440       ; bpbtype.csec = 1440
00000712 BA0202                         mov     dx, 202h       ; dl = bpbtype.spau = 2
                                                               ; dh = bpbtype.chead = 2
00000715 BB0970                         mov     bx, 7009h      ; bl = bpbtype.spt = 9
                                                               ; bh = bpbtype.dire = 112
00000718 EB30                           jmp     short Has1
                                ; ---------------------------------------------------------------------------

                                is_floppy:                     ; must be a 5.25" floppy if we come here
0000071A 80FCF8                         cmp     ah, 0F8h       ; valid media?? (0F8h-0FFh)
                                        ;jb     short got_unknown_medium
                                        ; 21/12/2023
0000071D 730A                           jnb     short chk_160K
                                ; ---------------------------------------------------------------------------
                                        ; 21/12/2023
                                        ; we have a 3.5" diskette for which we cannot build a bpb.
                                        ; we do not assume any type of bpb for this medium.
                                got_unknown_medium:
                                        ;mov    ds:set_id_flag,       0
0000071F C606[9B04]00                   mov     byte [set_id_flag], 0
00000724 B007                           mov     al, 7
00000726 F9                             stc
00000727 EB7B                           jmp     short getret
                                ; ---------------------------------------------------------------------------
                                chk_160K:
00000729 B001                           mov     al, 1          ; set number of       fat sectors
0000072B BB0840                         mov     bx, 16392      ; 64*256+8
                                                               ; set dir entries and sector max
0000072E B94001                         mov     cx, 320        ; 40*8
                                                               ; set size of drive
00000731 BA0101                         mov     dx, 257        ; 01*256+1
                                                               ; set head limit and sec/all unit
                                        ; 21/12/2023
                                        ;mov    al, 1          ; bpbtype.sbf = 1
                                        ;mov    bx, 4008h      ; bl = bpbtype.spt = 8
                                        ;                      ; bh = bpbtype.dire = 64
                                        ;mov    cx, 140h       ; bpbtype.csec = 320
                                        ;mov    dx, 101h       ; dl = bpbtype.spau = 1
                                        ;                      ; dh = bpbtype.chead = 1

00000734 F6C402                         test    ah, 2          ; test for 8 or       9 sector
00000737 7505                           jnz     short has8     ; nz = has 8 sectors

                                        ; 29/12/2023
                                        ;inc    al      ; 2    ; inc number of       fat sectors
                                        ;inc    bl      ; 9    ; inc sector max
00000739 40                             inc     ax
0000073A 43                             inc     bx

                                        ;add    cx, 40         ; increase size       (to 360)
                                        ; 18/12/2022
0000073B 80C128                         add     cl, 40 ; 28h   ; 180K (360 sectors)
                                has8:
0000073E F6C401                         test    ah, 1          ; test for 1 or       2 heads
00000741 7407                           jz      short Has1     ; jz = 1 head
00000743 01C9                           add     cx, cx         ; double size of disk
00000745 B770                           mov     bh, 112        ; increase number of directory entries
00000747 FEC6                           inc     dh      ; 2    ; inc sec/all unit
                                        ; 29/12/2023
                                        ;inc    dl      ; 2    ; inc head limit
00000749 42                             inc     dx
                                Has1:
                                        ; 02/09/2023 (PCDOS 7.1, IBMBIO.COM - BIOSCODE:0754h)
0000074A 1E                             push    ds
0000074B 06                             push    es
0000074C 1F                             pop     ds

                                        ;mov    [es:di+8], dh  ; [es:di+BDS.secperclus]
                                        ;mov    [es:di+0Ch], bh       ; [es:di+BDS.direntries]
                                        ;mov    [es:di+0Eh], cx       ; [es:di+BDS.totalsecs16]
                                        ;mov    [es:di+10h], ah       ; [es:di+BDS.media]
                                        ;mov    [es:di+11h], al       ; [es:di+BDS.fatsecs]
                                        ;mov    [es:di+13h], bl       ; [es:di+BDS.secpertrack]
                                        ;mov    [es:di+15h], dl       ; [es:di+BDS.heads]

0000074D 887508                         mov     [di+8], dh     ; [di+BDS.secperclus]
00000750 30F6                           xor     dh, dh
00000752 895515                         mov     [di+15h], dx   ; [di+BDS.heads]
00000755 88FA                           mov     dl, bh
00000757 89550C                         mov     [di+0Ch], dx   ; [di+BDS.direntries]
0000075A 894D0E                         mov     [di+0Eh], cx   ; [di+BDS.totalsecs16]
0000075D 894D1B                         mov     [di+1Bh], cx   ; [di+BDS.totalsecs32]
00000760 886510                         mov     [di+10h], ah   ; [di+BDS.media]
00000763 88C2                           mov     dl, al
00000765 895511                         mov     [di+11h], dx   ; [di+BDS.fatsecs]
00000768 88DA                           mov     dl, bl
0000076A 895513                         mov     [di+13h], dx   ; [di+BDS.secpertrack]

                                        ; the BDS_BPB.BPB_HIDDENSECTORS+2 field and the
                                        ; BDS_BPB.BPB_BIGTOTALSECTORS field need to be set
                                        ; to 0 since this code is for floppies

                                        ; 18/12/2022
                                        ;mov    word [es:di+19h], 0 ; [es:di+BDS.hiddensecs+2]
                                        ;mov    word [es:di+17h], 0 ; [es:di+BDS.hiddensecs]
                                        ;mov    word [es:di+1Dh], 0 ; [es:di+BDS.totalsecs32+2]
                                        ; 18/12/2022
0000076D 29C9                           sub     cx, cx ; 0
                                        ;mov    [es:di+19h], cx ; 0 ; [es:di+BDS.hiddensecs+2]
                                        ;mov    [es:di+17h], cx ; 0 ; [es:di+BDS.hiddensecs]
                                        ;mov    [es:di+1Dh], cx ; 0 ; [es:di+BDS.totalsecs32+2]

                                        ; 02/09/2023
0000076F 894D19                         mov     [di+19h], cx ; 0 ; [di+BDS.hiddensecs+2]
00000772 894D17                         mov     [di+17h], cx ; 0 ; [di+BDS.hiddensecs]
```

```
13028 00000775 894D1D                          mov     [di+1Dh], cx ; 0 ; [di+BDS.totalsecs32+2]
13029
13030                                          ; 21/12/2023 - Retro DOS v5.0
13031 00000778 894D1F                          mov     [di+1Fh], cx    ; [di+BDS.fatsecs32] ; BPB_FATSz32
13032 0000077B 894D21                          mov     [di+21h], cx    ; [di+BDS.fatsecs32+2]
13033 0000077E 894D27                          mov     [di+27h], cx    ; [di+BDS.rootdirclust]
13034 00000781 894D29                          mov     [di+29h], cx    ; [di+BDS.rootdirclust+2]
13035 00000784 894D2F                          mov     [di+2Fh], cx    ; [di+BDS.reserved]
13036                                                                  ;     BPB_Reserved (12 zero bytes)
13037 00000787 894D31                          mov     [di+31h], cx
13038 0000078A 894D33                          mov     [di+33h], cx
13039 0000078D 894D35                          mov     [di+35h], cx
13040 00000790 894D37                          mov     [di+37h], cx
13041 00000793 894D39                          mov     [di+39h], cx
13042 00000796 894D23                          mov     [di+23h], cx    ; [di+BDS.extflags] ; BPB_ExtFlags
13043 00000799 894D25                          mov     [di+25h], cx    ; [di+BDS.fsver] ; BPB_FSVer
13044
13045 0000079C 49                              dec     cx              ; -1 ; 0FFFFFFFFh
13046 0000079D 894D2B                          mov     [di+2Bh], cx    ; [di+BDS.fsinfo] ; BPB_FSInfo
13047 000007A0 894D2D                          mov     [di+2Dh], cx    ; [di+BDS.bkbootsec] ; BPB_BkBootSec
13048
13049 000007A3 1F                              pop     ds ; 02/09/2023
13050                                  getret:
13051 000007A4 5B                              pop     bx
13052 000007A5 5A                              pop     dx
13053 000007A6 59                              pop     cx
13054                                  ;getret_exit:              ; 21/12/2023
13055 000007A7 C3                              retn
13056                                  ; ---------------------------------------------------------------------------
13057
13058                                  getbp_err_ret:     ; before doing anything else, set set_id_flag to 0.
13059                                              ;mov   ds:set_id_flag,       0
13060                                              ; 19/10/2022
13061 000007A8 C606[9B04]00                     mov     byte [set_id_flag], 0
13062 000007AD E8F905                           call    maperror
13063 000007B0 EBF2                             jmp     short getret
13064                                  ; ---------------------------------------------------------------------------
13065                                  ; 21/12/2023
13066                                  ;           ; we have a 3.5" diskette for which we cannot build a bpb.
13067                                  ;           ; we do not assume any type of bpb for this medium.
13068                                  ;
13069                                  ;got_unknown_medium:
13070                                  ;           ;mov    ds:set_id_flag,       0
13071                                  ;           mov     byte [set_id_flag], 0
13072                                  ;           mov     al, 7
13073                                  ;           stc
13074                                  ;           jmp     short getret
13075
13076                                  ; =============== S U B   R O U T I N E =======================================
13077
13078                                  ; 15/10/2022 - Retro DOS 4.0 (Modified MSDOS 5.0 IO.SYS)
13079
13080                                  ; ---------------------------------------------------------------
13081                                  ; read in the boot sector. set carry if error in reading sector.
13082                                  ; bx is set to 1 if the boot sector is invalid, otherwise it is 0.
13083                                  ;
13084                                  ;   assumes es:di -> bds, ds-> Bios_Data
13085                                  ; ---------------------------------------------------------------
13086
13087                                  ; 10/03/2019 - Retro DOS v4.0
13088
13089                                  ; 30/12/2022 - Retro DOS v4.2
13090                                  ; (MSDOS 6.21 IO.SYS, BIOSCODE:06C3h)
13091                                  ; ((MSDOS 6.22 IO.SYS, BIOSCODE:06C3h)) ; 22/12/2023
13092
13093                                  ; 22/12/2023 - Retro DOS v5.0
13094                                  ; (PCDOS 7.1 IBMBIO.COM, BIOSCODE:07C6h)
13095
13096                                  readbootsec:
13097 000007B2 B600                             mov     dh, 0           ; head 0
13098 000007B4 B90100                           mov     cx, 1           ; cylinder 0, sector 1
13099 000007B7 E8FC00                           call    read_sector
13100 000007BA 7243                             jb      short err_ret
13101 000007BC 31DB                             xor     bx, bx          ; assume valid boot sector
13102
13103                                                   ; put a sanity check for the boot sector in here to detect
13104                                                   ; boot sectors that do not have valid bpbs. we examine the
13105                                                   ; first two bytes - they must contain a long jump (69h) or a
13106                                                   ; short jump (EBh) followed by a nop (90h), or a short jump
13107                                                   ; (E9h). if this test is passed, we further check by examining
13108                                                   ; the signature at the end of the boot sector for the word
13109                                                   ; AA55h. if the signature is not present, we examine the media
13110                                                   ; descriptor byte to see if it is valid. for dos 3.3, this
13111                                                   ; logic is modified a little bit. we are not going to check
13112                                                   ; signature. instead we are going to sanity check the media
13113                                                   ; byte in bpb regardless of the validity of signature. this is
13114                                                   ; to save the already developed commercial products that have
13115                                                   ; good jump instruction and signature but with the false bpb
13116                                                   ; informations
13117
13118                                  ; that will crash the diskette drive operation. (for example, symphony diskette).
13119
13120                                                   ; 02/09/2023
13121                                                   ; 19/10/2022
13122                                                   ;cmp    byte [disksector], 69h ; is it a direct jump?
13123                                                   ;jz     short check_bpb_mediabyte ; don't need to find a nop
13124                                                   ;cmp    byte [disksector], 0E9h ; dos 2.0 jump?
13125                                                   ;jz     short check_bpb_mediabyte ; no need for     nop
13126                                                   ;cmp    byte [disksector], 0EBh ; how about a short jump?
13127                                                   ;jnz    short invalidbootsec
13128                                                   ;cmp    byte [disksector+2], 90h ; is next one a nop?
13129                                                   ;jnz    short invalidbootsec
13130
13131                                                   ; 02/09/2023 (PCDOS 7.1)
13132 000007BE A0[5201]                                mov     al, [disksector]
13133 000007C1 3C69                                    cmp     al, 69h         ; is it a direct jump?
13134 000007C3 740F                                    je      short check_bpb_mediabyte
13135                                                                          ; don't need to find a nop
13136 000007C5 3CE9                                    cmp     al, 0E9h        ; dos 2.0 jump?
13137 000007C7 740B                                    je      short check_bpb_mediabyte
13138                                                                          ; no need for nop
13139 000007C9 3CEB                                    cmp     al, 0EBh        ; how about a short jump?
13140 000007CB 7530                                    jne     short invalidbootsec
13141 000007CD 803E[5401]90                            cmp     byte [disksector+2], 90h ; is next one a nop?
13142 000007D2 7529                                    jne     short invalidbootsec
13143
13144                                  ; 15/10/5022 - Retro DOS v4.0 (Modified MSDOS 5.0 IO.SYS)
13145                                  ;
13146                                  ;; 10/03/2019
13147                                  ;; (MSDOS 3.3, MSDISK.ASM, 1988)
13148                                  ;;
13149                                  ;; Don't have to perform the following signature check since
13150                                  ;; we need to check the media byte even with the good signatured diskette.
13151                                  ;;
```

```
                                ;;check_signature:
                                ;;          cmp     word [cs:disksector+1FEh],0AA55h ; see if non-ibm
                                ;;                                                   ; disk or 1.x media.
                                ;;          jz      short checksinglesided ; go see if singled sided medium.
                                ;;                                         ; may need some special handling

                                ; check for non-ibm disks which do not have the signature AA55h at the
                                ; end of the boot sector, but still have a valid boot sector. this is done
                                ; by examining the media descriptor in the boot sector.

                                          ; 19/10/2022
                                check_bpb_mediabyte:
000007D4 A0[6701]                         mov     al, [disksector+15h]
                                                                   ; [disksector+EXT_BOOT.BPB+EBPB.MEDIADESCRIPTOR]
000007D7 50                               push    ax ; 02/09/2023
000007D8 24F0                             and     al, 0F0h
000007DA 3CF0                             cmp     al, 0F0h        ; allow for strange media
000007DC 58                               pop     ax ; 02/09/2023
000007DD 751E                             jnz     short invalidbootsec

                                ; there were some (apparently a lot of them) diskettes that had been formatted
                                ; under dos 3.1 and earlier versions which have invalid bpbs in their boot
                                ; sectors. these are specifically diskettes that were formatted in drives
                                ; with one head, or whose side 0 was bad. these contain bpbs in the boot
                                ; sector that have the sec/clus field set to 2 instead of 1, as is standard
                                ; in dos. in order to support them, we have to introduce a "hack" that will
                                ; help our build bpb routine to recognise these specific cases, and to
                                ; set up out copy of the bpb accordingly.
                                ; we do this by checking to see if the boot sector is off a diskette that
                                ; is single-sided and is a pre-dos 3.20 diskette. if it is, we set the
                                ; sec/clus field to 1. if not, we carry on as normal.

                                checksinglesided:
                                          ;mov   al, [disksector+15h]
                                          ; 02/09/2023
                                          ; al = [disksector+15h]
000007DF 3CF0                             cmp     al, 0F0h
000007E1 741B                             jz      short gooddsk
000007E3 A801                             test    al, 1
000007E5 7517                             jnz     short gooddsk
000007E7 813E[5A01]332E                   cmp     word [disksector+8], 2E33h ; "3."
000007ED 7507                             jnz     short mustbeearlier
000007EF 803E[5C01]32                     cmp     byte [disksector+0Ah], 32h ; "2"
000007F4 7308                             jnb     short gooddsk

                                ; we must have a pre-3.20 diskette. set the sec/clus field to 1

                                mustbeearlier:
000007F6 C606[5F01]01                     mov     byte [disksector+0Dh], 1
                                                                   ; [disksector+EXT_BOOT.BPB+EBPB.SECTORSPERCLUSTER]
000007FB EB01                             jmp     short gooddsk
                                ; ---------------------------------------------------------------------------

                                invalidbootsec:
000007FD 43                               inc     bx              ; indicate that boot sector invalid
                                          ; 10/12/2022
                                movbpb_ret:
                                gooddsk:
000007FE F8                               clc
                                err_ret:
000007FF C3                               retn
                                ; ---------------------------------------------------------------------------

                                          ; 10/12/2022
                                ;err_ret:
                                          ;retn

                                ; =============== S U B   R O U T I N E =======================================

                                ; 15/10/2022
                                ; ---------------------------------------------------------------------------
                                ; 'movbpb' moves the bpb read from the boot sector into registers for use by
                                ; getbp routine at has1
                                ;
                                ; if the set_id_flag is 1, and if an extended boot record, then set volume
                                ; serial number, volume label, file system id in bds according to
                                ; the boot record. after that, this routine will set the set_id_flag to 2
                                ; to signal that volume label is set already from the extended boot record
                                ; (so, don't set it again by calling "set_volume_id" routine which uses
                                ; the volume label in the root directory.)
                                ; ---------------------------------------------------------------------------

                                ; 10/03/2019 - Retro DOS v4.0

                                ; 22/12/2023
                                %if 0
                                          ; 19/10/2022
                                movbpb:
                                          mov     dh, [disksector+0Dh]
                                                                   ; disksector+EXT_BOOT.BPB+EBPB.SECTORSPERCLUSTER]
                                                                   ; sectors per unit
                                          mov     bh, [disksector+11h]
                                                                   ; [disksector+EXT_BOOT.BPB+EBPB.ROOTENTRIES]
                                                                   ; number of directory entries
                                          mov     cx, [disksector+13h]
                                                                   ; [disksector+EXT_BOOT.BPB+EBPB.TOTALSECTORS]
                                                                   ; size of drive
                                          mov     ah, [disksector+15h]
                                                                   ; [disksector+EXT_BOOT.BPB+EBPB.MEDIADESCRIPTOR]
                                                                   ; media descriptor
                                          mov     al, [disksector+16h];
                                                                   ; [disksector+EXT_BOOT.BPB+EBPB.SECTORSPERFAT]
                                                                   ; number of fat       sectors
                                          mov     bl, [disksector+18h]
                                                                   ; [disksector+EXT_BOOT.BPB+EBPB.SECTORSPERTRACK]
                                                                   ; sectors per track
                                          mov     dl, [disksector+1Ah]
                                                                   ; [disksector+EXT_BOOT.BPB+EBPB.HEADS]
                                                                   ; number of heads
                                %else
                                          ; 29/12/2023
                                          ; 22/12/2023 - Retro DOS v5.0
                                          ; (PCDOS 7.1 IBMBIO.COM, BIOSCODE:0814h)
                                          ;;;
                                movbpb:
00000800 57                               push    di
00000801 83C706                           add     di, 6           ; BDS+6 = BDS.BPB
00000804 8D36[5D01]                       lea     si, [disksector+0Bh]
00000808 B93500                           mov     cx, 53          ; copy bios parameters block
                                                                   ; from BPB_BytsPerSec to (FAT32) BS_DrvNum (excluded)
0000080B FC                               cld
0000080C F3A4                             rep movsb
0000080E 8B4CD3                           mov     cx, [si-45]     ; si = disksector+64 -> 64-45 = 19
                                                                   ; disksektor+19 = BPB_TotSec16
```

```
13276 00000811 31C0                             xor     ax, ax
13277 00000813 E308                             jcxz    movbpb_bigdisk
13278 00000815 26894DE0                         mov     [es:di-32], cx ; write 16 bit total sectors
13279                                                                  ; to 32 bit total sectors field
13280 00000819 268945E2                         mov     [es:di-30], ax ; BPB_TotalSec32+2 (BDS offset 29, BPB offset 23)
13281                                   movbpb_bigdisk:
13282 0000081D 3944D6                           cmp     [si-42], ax     ; BPB_FATSz16 = disksector+22
13283 00000820 7410                             jz      short movbpb_fat32
13284                                   movbpb_fat:
13285 00000822 83EF1C                           sub     di, 28          ; di = BDS offset 31 (BPB offset 25)
13286                                           ; 29/12/2023
13287 00000825 B90C00                           mov     cx, 12          ; clear 12 byte extended BDS (FAT32) fields
13288                                                                   ; (which are used only for FAT32 disks)
13289 00000828 F3AA                             rep stosb
13290 0000082A 48                               dec     ax              ; -1 ; 0FFFFh
13291 0000082B AB                               stosw                   ; set BDS offset 43 (dword) to -1
13292                                                                   ; dword [BDS.BPB_FSInfo] = 0FFFFFFFFh
13293 0000082C AB                               stosw
13294 0000082D 40                               inc     ax              ; ax = 0
13295 0000082E B10C                             mov     cl, 12
13296                                           ;mov    cx, 12          ; clear BDS offset 47 to 59
13297                                                                   ; (BPB offset 41 to 53) (disksector offset 52 to 64)
13298 00000830 F3AA                             rep stosb
13299                                   movbpb_fat32:
13300 00000832 5F                               pop     di
13301                                   %endif
13302                                           ;;;
13303
13304 00000833 803E[9B04]01                     cmp     byte [set_id_flag], 1 ; called by get_bpb?
13305 00000838 75C4                             jnz     short movbpb_ret
13306 0000083A E81200                           call    mov_media_ids
13307 0000083D 7205                             jb      short movbpb_conv ; conventional boot record?
13308 0000083F C606[9B04]02                     mov     byte [set_id_flag], 2 ; signals that volume id is set
13309                                   movbpb_conv:
13310 00000844 803E[7700]01                     cmp     byte [fhave96], 1
13311 00000849 75B3                             jnz     short movbpb_ret
13312 0000084B E83714                           call    resetchanged    ; reset flags in bds to       not fchanged.
13313                                           ; 10/12/2022
13314                                           ; cf = 0
13315                                   ;movbpb_ret:
13316                                           ;clc
13317 0000084E C3                               retn
13318
13319                                   ; =============== S U B   R O U T I N E =======================================
13320
13321                                   ;copy the boot_serial number, volume id, and filesystem id from the
13322                                   ;***extended boot record*** in ds:disksector to the bds table pointed
13323                                   ;by es:di.
13324
13325                                   ;in.) es:di -> bds
13326                                   ;     ds:disksector = valid extended boot record.
13327                                   ;out.) vol_serial, bds_volid and bds_system_id in bds are set according to
13328                                   ;       the boot record information.
13329                                   ;     carry flag set if not an extended bpb.
13330                                   ;     all registers saved except the flag.
13331
13332                                   ; 22/12/2023
13333                                   %if 0
13334                                           ; 19/10/2022
13335                                   mov_media_ids:
13336                                           cmp     byte [disksector+26h], 29h
13337                                                                   ; [disksector+EXT_BOOT.SIG],
13338                                                                   ; EXT_BOOT_SIGNATURE
13339                                           jnz     short mmi_not_ext
13340                                           push    cx
13341                                           mov     cx, [disksector+27h]
13342                                                                   ; [disksector+EXT_BOOT.SERIAL]
13343                                           mov     [es:di+57h], cx       ; [es:di+BDS.vol_serial]
13344                                           mov     cx, [disksector+29h]
13345                                                                   ; [disksector+EXT_BOOT.SERIAL+2]
13346                                           mov     [es:di+59h], cx       ; [es:di+BDS.vol_serial+2]
13347                                           push    di
13348                                           push    si
13349                                           mov     cx, 11          ; size_of_EXT_BOOT_VOL_LABEL
13350                                           mov     si, disksector+2Bh
13351                                           ;mov    si, (offset disksector+2Bh) ;
13352                                                                   ; disksector+EXT_BOOT.VOL_LABEL
13353                                           add     di, 75          ; BDS.volid
13354                                           rep movsb
13355                                           ;mov    cx, 8           ; size_of_EXT_SYSTEM_ID
13356                                           ; 10/12/2022
13357                                           mov     cl, 8 ; cx = 8
13358                                           mov     si, disksector+36h
13359                                           ;mov    si, (offset disksector+36h) ; disksector+EXT_BOOT.SYSTEM_ID
13360                                           add     di, 5           ; (BDS.filesys_id-BDS.volid)-size_of_EXT_BOOT_VOL_LABEL
13361                                           rep movsb
13362                                           pop     si
13363                                           pop     di
13364                                           pop     cx
13365                                           ; 10/12/2022
13366                                           ; cf = 0
13367                                           ;clc            ; this clc is not required (16/06/2019 - Erdogan Tan)
13368                                                           ; (20/09/2022)
13369                                           retn
13370                                   %else
13371                                           ; 22/12/2023 - Retro DOS v5.0
13372                                           ; (PCDOS 7.1 IBMBIO.COM, BIOSCODE:0865h)
13373                                           ;;;
13374                                   mov_media_ids:
13375 0000084F 833E[6801]00                     cmp     word [disksector+16h], 0 ; BPB.FATSz16
13376 00000854 7507                             jnz     short mmi_chk_fat
13377 00000856 803E[9401]29                     cmp     byte [disksector+42h], 29h
13378                                                                   ; [disksector+FAT32_EXT_BOOT.SIG],
13379                                                                   ; EXT_BOOT_SIGNATURE
13380 0000085B EB05                             jmp     short mmi_chk_fat32
13381                                   mmi_chk_fat:
13382 0000085D 803E[7801]29                     cmp     byte [disksector+26h], 29h
13383                                                                   ; [disksector+EXT_BOOT.SIG],EXT_BOOT_SIGNATURE
13384                                   mmi_chk_fat32:
13385 00000862 7543                             jnz     short mmi_not_ext
13386 00000864 51                               push    cx
13387 00000865 50                               push    ax
13388 00000866 57                               push    di
13389 00000867 56                               push    si
13390 00000868 1E                               push    ds
13391 00000869 833E[6801]00                     cmp     word [disksector+16h], 0 ; BPB.FATSz16
13392 0000086E 750C                             jnz     short mmi_fat
13393
13394                                   mmi_fat32:                      ; FAT32 file system
13395                                           ;lds    cx, dword ptr ds:disksector+43h
13396 00000870 C50E[9501]                       lds     cx, [disksector+43h]  ; BS_FAT32_VolID
13397 00000874 BE[9901]                         mov     si, disksector+47h    ; BS_FAT32_VolLab
13398 00000877 B8[A401]                         mov     ax, disksector+52h    ; BS_FAT32_FilSysType
13399 0000087A EB0A                             jmp     short mmi_do
```

```
13400
13401                                   mmi_fat:
13402                                           ;lds    cx, dword ptr ds:disksector+27h
13403 0000087C C50E[7901]                       lds     cx, [disksector+27h]  ; BS_VolID
13404 00000880 BE[7D01]                         mov     si, disksector+2Bh    ; BS_VolLab
13405 00000883 B8[8801]                         mov     ax, disksector+36h    ; BS_FilSysType
13406                                   mmi_do:
13407 00000886 26898D8900                       mov     [es:di+89h], cx       ; [es:di+BDS.vol_serial]
13408                                                                         ; (BDS offset 137)
13409 0000088B 268C9D8B00                       mov     [es:di+8Bh], ds       ; [es:di+BDS.vol_serial+2]
13410 00000890 1F                               pop     ds
13411 00000891 B90B00                           mov     cx, 11
13412 00000894 83C77D                           add     di, 125         ; di = di+125 = BDS.volid
13413 00000897 F3A4                             rep movsb
13414 00000899 B108                             mov     cl, 8           ; di = di+136
13415 0000089B 89C6                             mov     si, ax          ; BS_FilSysType or BS_FAT32_FilSysType
13416 0000089D 83C705                           add     di, 5           ; di = di+141 = BDS.filesys_id
13417 000008A0 F3A4                             rep movsb
13418 000008A2 5E                               pop     si
13419 000008A3 5F                               pop     di
13420 000008A4 58                               pop     ax
13421 000008A5 59                               pop     cx
13422                                           ;clc    ; this clc is not required (16/06/2019 - Erdogan Tan)
13423                                                   ; (20/09/2022 - 27/06/2023) MSDOS 6.21 .. PCDOS 7.1
13424 000008A6 C3                               retn
13425                                   %endif
13426                                           ;;;
13427
13428                                   ; ---------------------------------------------------------------------------
13429
13430                                   mmi_not_ext:
13431 000008A7 F9                               stc
13432 000008A8 C3                               retn
13433
13434                                   ; =============== S U B   R O U T I N E =======================================
13435
13436                                   ; 15/10/2022
13437                                   ; ---------------------------------------------------------------
13438                                   ; read in the fat sector and get the media byte from it.
13439                                   ; input : es:di -> bds
13440                                   ; output:
13441                                   ;     carry set if an error occurs, ax contains error code.
13442                                   ;     otherwise, ah contains media byte on exit
13443                                   ; ---------------------------------------------------------------
13444
13445                                   readfat:
13446                                           ;mov    dh, 0
13447                                           ; 10/12/2022
13448 000008A9 30F6                             xor     dh, dh
13449 000008AB B90200                           mov     cx, 2           ; head 0
13450                                                                   ; cylinder 0, sector 2
13451 000008AE E80500                           call    read_sector
13452 000008B1 7202                             jb      short bad_fat_ret
13453 000008B3 8A27                             mov     ah, [bx]        ; media byte
13454                                   bad_fat_ret:
13455 000008B5 C3                               retn
13456
13457                                   ; =============== S U B   R O U T I N E =======================================
13458
13459                                   ; 15/10/2022
13460
13461                                   ; ---------------------------------------------------------------------------
13462                                   ; read a single sector into the temp buffer.
13463                                   ; perform three retries in case of error.
13464                                   ;   inputs: es:[di].bds_drivenum has physical drive to use
13465                                   ;           cx has sector and cylinder
13466                                   ;           dh has head
13467                                   ;           es:di has bds
13468                                   ;           ds has Bios_Data
13469                                   ;
13470                                   ;   outputs:        carry clear
13471                                   ;                Bios_Data:bx point to sector
13472                                   ;                   (note: some callers assume location of buffer)
13473                                   ;
13474                                   ;           carry set
13475                                   ;               ax has rom error code
13476                                   ;
13477                                   ; register bp is preserved.
13478                                   ; ---------------------------------------------------------------------------
13479
13480                                   ; 10/03/2019 - Retro DOS v4.0
13481                                   ; 22/12/2023 - Retro DOS v5.0
13482
13483                                           ; 19/10/2022
13484                                   read_sector:
13485 000008B6 55                               push    bp
13486 000008B7 BD0300                           mov     bp, 3           ; make 3 attempts
13487 000008BA 268A5504                         mov     dl, [es:di+4]  ; [es:di+BDS.drivenum]
13488 000008BE BB[5201]                         mov     bx, disksector ; get es:bx to point to      buffer
13489                                   rd_ret:
13490 000008C1 06                               push    es
13491 000008C2 1E                               push    ds
13492 000008C3 07                               pop     es
13493 000008C4 B80102                           mov     ax, 201h
13494 000008C7 CD13                             int     13h             ; DISK - READ SECTORS INTO MEMORY
13495                                                                   ; AL = number of sectors to read, CH = track, CL = sector
13496                                                                   ; DH = head, DL       = drive, ES:BX -> buffer to fill
13497                                                                   ; Return: CF set on error, AH =      status, AL = number of sectors read
13498 000008C9 07                               pop     es
13499 000008CA 734A                             jnb     short okret2
13500                                   rd_rty:
13501 000008CC E81205                           call    again           ; reset disk, decrement       bp, preserve ax
13502 000008CF 7442                             jz      short err_rd_ret
13503
13504                                           ; 22/12/2023 - Retro DOS v5.0 (PCDOS 7.1 IBMBIO.COM)
13505 000008D1 26F6453F01                       test    byte [es:di+3Fh], 1
13506                                           ;test   byte [es:di+23h], 1
13507                                           ;;test  byte ptr [es:di+23h], 1       ; [es:di+BDS.flags]
13508                                                                   ; fnon_removable
13509 000008D6 75E9                             jnz     short rd_ret
13510 000008D8 803E[A905]00                     cmp     byte [media_set_for_format], 0
13511 000008DD 7510                             jnz     short rd_skip1_dpt
13512 000008DF 50                               push    ax
13513 000008E0 1E                               push    ds              ; for retry, set the head settle time to 0Fh
13514 000008E1 C536[2D01]                       lds     si, [dpt]
13515                                           ;mov    al, [si+9]      ; [si+DISK_PARMS.DISK_HEAD_STTL]
13516                                           ;mov    byte [si+9], 15 ; [si+DISK_PARMS.DISK_HEAD_STTL]
13517                                           ;                       ; NORMSETTLE
13518                                           ; 12/12/2022
13519 000008E5 B00F                             mov     al, 15
13520 000008E7 864409                           xchg    al, [si+9]
13521                                           ;
13522 000008EA 1F                               pop     ds
13523 000008EB A2[2A01]                         mov     [save_head_sttl], al
```

```
13524 000008EE 58                          pop     ax
13525                                  rd_skip1_dpt:
13526 000008EF 06                          push    es
13527 000008F0 1E                          push    ds
13528 000008F1 07                          pop     es
13529 000008F2 B80102                      mov     ax, 201h
13530 000008F5 CD13                        int     13h             ; DISK - READ SECTORS INTO MEMORY
13531                                                              ; AL = number of sectors to read, CH = track, CL = sector
13532                                                              ; DH = head, DL       = drive, ES:BX -> buffer to fill
13533                                                              ; Return: CF set on error, AH =      status, AL = number of sectors read
13534 000008F7 07                          pop     es
13535 000008F8 9C                          pushf
13536 000008F9 803E[A905]00                cmp     byte [media_set_for_format], 0
13537 000008FE 750E                        jnz     short rd_skip2_dpt
13538 00000900 50                          push    ax
13539 00000901 A0[2A01]                    mov     al, [save_head_sttl]
13540 00000904 1E                          push    ds
13541 00000905 C536[2D01]                  lds     si, [dpt]
13542 00000909 884409                      mov     [si+9], al      ; [si+DISK_PARMS.DISK_HEAD_STTL]
13543 0000090C 1F                          pop     ds
13544 0000090D 58                          pop     ax
13545                                  rd_skip2_dpt:
13546 0000090E 9D                          popf
13547 0000090F 7305                        jnb     short okret2
13548 00000911 EBB9                        jmp     short rd_rty
13549                                  ; ---------------------------------------------------------------------------
13550
13551                                  err_rd_ret:
13552 00000913 B2FF                        mov     dl, 0FFh        ; make sure we ask rom if media      has changed
13553                                                              ; return error
13554 00000915 F9                          stc
13555
13556                                  ; update information pertaining to last drive accessed, time of access, last
13557                                  ; track accessed in that drive.
13558
13559                                  okret2:
13560 00000916 8816[7600]                  mov     [step_drv], dl ; set up for head settle logic in disk
13561 0000091A 8816[1E01]                  mov     [tim_drv], dl  ; save drive last accessed
13562
13563                                      ; 22/12/2023 - Retro DOS v5.0 (PCDOS 7.1 IBMBIO.COM)
13564 0000091E 26886D78                    mov     [es:di+78h], ch
13565                                      ;mov    [es:di+46h], ch       ; [es:di+BDS.track]
13566                                                              ; save last track accessed on this drive
13567                                                              ; preserve flags in case error occurred
13568 00000922 9C                          pushf
13569 00000923 E89B04                      call    set_tim
13570 00000926 9D                          popf                    ; restore flags
13571 00000927 5D                          pop     bp
13572 00000928 C3                          retn
13573
13574                                  ;---------------------------------------------------------------------------
13575                                  ;   disk open/close routines
13576                                  ;---------------------------------------------------------------------------
13577
13578                                  dsk_open:                       ; 2C7h:80Ah = 70h:2D7Ah
13579 00000929 803E[7700]00                cmp     byte [fhave96], 0
13580 0000092E 7407                        jz      short dsk_open_exit ; done if no changeline support
13581 00000930 E871FC                      call    SetDrive        ; get bds for drive
13582                                      ; 22/12/2023 - Retro DOS v5.0 (PCDOS 7.1 IBMBIO.COM)
13583 00000933 26FF453C                    inc     word [es:di+3Ch] ; [es:di+BDS.opcnt] ; BDS offset 60
13584                                      ;inc    word [es:di+20h] ; [es:di+BDS.opcnt]
13585                                  dsk_open_exit:
13586                                      ; 10/12/2022
13587                                      ; cf = 0
13588                                      ;clc            ; CF is already ZERO here (18/09/2022, MSDOS 5.0 IO.SYS)
13589                                                      ; (19/07/2019 -       Erdogan Tan - MSDOS 6.0       IO.SYS - retrodos4.s)
13590 00000937 C3                          retn
13591                                  ; ---------------------------------------------------------------------------
13592
13593                                  dsk_close:                      ; 2C7h:81Ah = 70h:2D8Ah
13594 00000938 803E[7700]00                cmp     byte [fhave96], 0
13595 0000093D 740E                        jz      short exitjx    ; done if no changeline      support
13596 0000093F E862FC                      call    SetDrive        ; get bds for drive
13597                                      ; 22/12/2023 - Retro DOS v5.0 (PCDOS 7.1 IBMBIO.COM)
13598 00000942 26837D3C00                  cmp     word [es:di+3Ch], 0 ; [es:di+BDS.opcnt] ; BDS off 60
13599                                      ;cmp    word [es:di+20h], 0 ; [es:di+BDS.opcnt]
13600 00000947 7404                        jz      short exitjx    ; watch out for wrap
13601                                      ; 22/12/2023
13602 00000949 26FF4D3C                    dec     word [es:di+3Ch]
13603                                      ;dec    word [es:di+20h]
13604                                  exitjx:
13605                                      ; 10/12/2022
13606                                      ; cf = 0
13607                                      ;clc            ; CF is already ZERO here (18/09/2022, MSDOS 5.0 IO.SYS)
13608                                                      ; (19/07/2019 -       Erdogan Tan - MSDOS 6.0       IO.SYS - retrodos4.s)
13609 0000094D C3                          retn
13610
13611                                  ;---------------------------------------------------------------------------
13612                                  ;           disk removable routine
13613                                  ;---------------------------------------------------------------------------
13614
13615                                      ; al is unit #
13616                                  dsk_rem:                        ; 2C7h:831h = 70h:2DA1h
13617 0000094E E853FC                      call    SetDrive        ; get bds for this drive
13618                                      ; 22/12/2023 - Retro DOS v5.0 (PCDOS 7.1 IBMBIO.COM)
13619                                      ;test   byte [es:di+BDS.flags], fnon_removable
13620 00000951 26F6453F01                  test    byte [es:di+3Fh], 1 ; [es:di+BDS.flags], fnon_removable
13621 00000956 74F5                        jz      short exitjx
13622                                      ;test   byte [es:di+23h], 1 ; [es:di+BDS.flags], fnon_removable
13623                                      ;;jnz   short x_bus_exit ; non_rem
13624                                      ;jnz    short non_rem  ; 15/10/2022
13625                                      ;; 10/12/2022
13626                                      ;; cf = 0
13627                                      ;;clc                  ; CF is already ZERO here
13628                                      ;                      ; 15/10/2022
13629                                      ;retn
13630                                  ; ---------------------------------------------------------------------------
13631
13632                                  non_rem:
13633                                  x_bus_exit:
13634 00000958 B403                        mov     ah, 3           ; 2C7h:83Dh = 0070h:2DADh
13635                                                              ; return busy status
13636 0000095A F9                          stc
13637                                  dsk_ret:
13638 0000095B C3                          retn
13639
13640                                  ;---------------------------------------------------------------------------
13641                                  ;           disk i/o routines
13642                                  ;---------------------------------------------------------------------------
13643
13644                                  dsk_writv:                      ; 2C7h:841h = 70h:2DB1h
13645                                      ;mov    word [wrtverify], 103h
13646                                      ; 19/10/2022
13647 0000095C C706[2001]0301              mov     word [rflag], 103h
```

```
13648                                   ;mov    word ptr ds:rflag, 103h      ; write and verify
13649 00000962 EB06                     jmp     short dsk_cl
13650                                 ; ---------------------------------------------------------------------------
13651
13652                                 dsk_writ:                               ; 2C7h:849h = 70h:2DB9h
13653                                   ;mov    word [wrtverify], 3
13654                                   ; 19/10/2022
13655 00000964 C706[2001]0300           mov     word [rflag], 3
13656                                   ;mov    word ptr ds:rflag, 3 ; romwrite
13657                                 dsk_cl:
13658 0000096A E8A400                   call    diskio          ; romwrite
13659                                 ; ---------------------------------------------------------------------------
13660
13661                                 dsk_io:
13662 0000096D 73EC                     jnb     short dsk_ret
13663 0000096F E965F7                   jmp     bc_err_cnt
13664                                 ; ---------------------------------------------------------------------------
13665
13666                                 dsk_read:                               ; ; 2C7h:857h =       70h:2DC7h
13667 00000972 E89700                   call    diskrd
13668 00000975 EBF6                     jmp     short dsk_io
13669
13670                                 ; =============== S U B   R O U T I N E =======================================
13671
13672                                 ; 15/10/2022
13673                                 ; 10/03/2019 - Retro DOS v4.0
13674                                 ; 22/12/2023 - Retro DOS v5.0
13675
13676                                 ;--------------------------------------------------------
13677                                 ; miscellaneous odd jump routines.
13678                                 ; moved out of mainline for speed.
13679
13680                                 ; if we have a system where we have virtual drives, we need
13681                                 ; to prompt the user to place the correct disk in the drive.
13682                                 ;
13683                                 ;   assume es:di -> bds, ds:->Bios_Data
13684                                 ;--------------------------------------------------------
13685
13686                                   ; 19/10/2022
13687                                 checksingle:
13688 00000977 50                       push    ax
13689 00000978 53                       push    bx
13690                                   ; 22/12/2023 - Retro DOS v5.0 (PCDOS 7.1 IBMBIO.COM)
13691 00000979 268B5D3F                 mov     bx, [es:di+3Fh]         ; [es:di+BDS.flags]
13692                                   ;mov    bx, [es:di+23h]         ; [es:di+BDS.flags]
13693
13694                                 ; if hard drive, cannot change disk.
13695                                 ; if current owner of physical drive, no need to change diskette.
13696
13697 0000097D F6C321                   test    bl, 21h         ; fnon_removable|fi_own_physical
13698 00000980 7573                     jnz     short singleret
13699 00000982 F6C310                   test    bl, 10h         ; fi_am_mult
13700                                                           ; is there a drive sharing this      physical drive?
13701 00000985 746E                     jz      short singleret
13702
13703                                 ; look for the previous owner of this physical drive
13704                                 ; and reset its ownership flag.
13705
13706 00000987 268A4504                 mov     al, [es:di+4]  ; [es:di+BDS.drivenum]
13707                                                           ; get physical drive number
13708 0000098B 06                       push    es              ; preserve pointer to current bds
13709 0000098C 57                       push    di
13710 0000098D C43E[1901]               les     di, [start_bds] ; get first bds
13711                                 scan_list:
13712 00000991 26384504                 cmp     [es:di+4], al
13713 00000995 7553                     jnz     short scan_skip         ; Not our drive. Try next bds.
13714 00000997 B320                     mov     bl, 20h ; ' '   ; fi_own_physical ; test ownership flag
13715                                   ; 22/12/2023 - Retro DOS v5.0 (PCDOS 7.1 IBMBIO.COM)
13716 00000999 26845D3F                 test    [es:di+3Fh], bl         ; [es:di+BDS.flags]
13717                                   ;test   [es:di+23h], bl
13718 0000099D 744B                     jz      short scan_skip         ; he doesn't own it either. continue
13719 0000099F 26305D3F                 xor     [es:di+3Fh], bl
13720                                   ;xor    [es:di+23h], bl         ; reset ownership flag
13721 000009A3 5F                       pop     di              ; restore pointer to current bds
13722 000009A4 07                       pop     es
13723 000009A5 26085D3F                 or      [es:di+3Fh], bl
13724                                   ;or     [es:di+23h], bl         ; ; set ownership flag
13725
13726                                 ; we examine the fsetowner flag. if it is set, then we are using the code in
13727                                 ; checksingle to just set the owner of a drive. we must not issue the prompt
13728                                 ; in this case.
13729 000009A9 803E[7A00]01              cmp     byte [fsetowner], 1
13730 000009AE 7517                     jnz     short not_fsetowner
13731                                   ;cmp    byte ptr es:[di+4], 0 ;       are we handling       drive number 0 ?
13732 000009B0 26807D0400               cmp     byte [es:di+4], 0
13733 000009B5 753E                     jnz     short singleret
13734 000009B7 268A4505                 mov     al, [es:di+5]
13735                                   ;mov    al, es:[di+5]  ; [es:di+BDS.drivelet]
13736                                                           ; get the DOS drive letter
13737 000009BB 06                       push    es
13738 000009BC 8E06[1A00]               mov     es, [zeroseg]
13739 000009C0 26A20405                 mov     [es:LSTDRV], al
13740                                   ;mov    es:504h, al     ; [es:LSTDRV]
13741                                                           ; set up sdsb
13742 000009C4 07                       pop     es              ; restore bds pointer
13743 000009C5 EB2E                     jmp     short singleret
13744                                 ; ---------------------------------------------------------------------------
13745
13746                                 ; to support "backward" compatibility with ibm's "single drive status byte"
13747                                 ; we now check to see if we are in a single drive system and the application
13748                                 ; has "cleverly" diddled the sdsb
13749
13750                                 not_fsetowner:
13751 000009C7 803E[7800]02              cmp     byte [single], 2 ; if (single_drive_system)
13752 000009CC 7517                     jnz     short ignore_sdsb
13753 000009CE 50                       push    ax
13754 000009CF 268A4505                 mov     al, [es:di+5]  ; if (curr_drv == req_drv)
13755 000009D3 88C4                     mov     ah, al
13756 000009D5 06                       push    es
13757 000009D6 8E06[1A00]               mov     es, [zeroseg]
13758 000009DA 2686060405               xchg    al, [es:LSTDRV]
13759                                   ;xchg   al, es:504h     ; [es:LSTDRV]
13760                                                           ; then swap(curr_drv,req_drv)
13761 000009DF 07                       pop     es
13762 000009E0 38C4                     cmp     ah, al          ; else
13763 000009E2 58                       pop     ax              ; swap(curr_drv,req_drv)
13764 000009E3 7410                     jz      short singleret       ; issue swap_dsk_msg
13765                                 ignore_sdsb:
13766 000009E5 E8B310                   call    swpdsk
13767 000009E8 EB0B                     jmp     short singleret
13768                                 ; ---------------------------------------------------------------------------
13769
13770                                 scan_skip:
13771 000009EA 26C43D                   les     di, [es:di]
```

```
13772                                           ;les    di, es:[di]    ; [es:di+BDS.link]
13773                                                                  ; go to next bds
13774 000009ED 83FFFF                          cmp     di, 0FFFFh ; -1        ; end of list?
13775 000009F0 759F                            jnz     short scan_list        ; continue until hit end of list
13776 000009F2 F9                              stc
13777 000009F3 5F                              pop     di             ; restore current bds
13778 000009F4 07                              pop     es
13779                                   singleret:
13780 000009F5 5B                              pop     bx
13781 000009F6 58                              pop     ax
13782 000009F7 C3                              retn
13783
13784                                   ; 22/12/2023
13785                                   %if 0
13786                                   ; ---------------------------------------------------------------------------
13787
13788                                   baddrive:
13789                                           mov     al, 8          ; sector not found
13790                                           jmp     short baddrive_ret
13791                                   %endif
13792
13793                                   ; ---------------------------------------------------------------------------
13794
13795                                   unformatteddrive:
13796 000009F8 B007                             mov     al, 7          ; unknown media
13797                                   ;baddrive_ret:
13798 000009FA F9                               stc
13799                                   ; ---------------------------------------------------------------------------
13800
13801                                   ioret:
13802 000009FB C3                               retn
13803
13804                                   ; ---------------------------------------------------------------------------
13805
13806                                           ; 22/12/2023 - Retro DOS v5.0
13807                                           ; PCDOS 7.1 IBMBIO.COM - BIOSCODE:0A1Bh
13808
13809 000009FC 10                      LBA_Packet: db 16                ; ...
13810                                                                    ; DAP buffer
13811 000009FD 00                                  db 0
13812 000009FE 0000                    dap_block_cnt:     dw 0                 ; ...
13813 00000A00 00000000                dap_trans_buf:     dd 0                 ; ...
13814 00000A04 00000000                dap_lba_value:     dd 0                 ; ...
13815 00000A08 00000000                           dd 0
13816
13817                                   ; ---------------------------------------------------------------------------
13818
13819                                   ; 15/10/2022
13820
13821                                   ; ---------------------------------------------------------------------------
13822                                   ;   disk i/o handler
13823                                   ;
13824                                   ;   al = drive number (0-6)
13825                                   ;   ah = media descriptor
13826                                   ;   cx = sector count
13827                                   ;   dx = first sector (low)
13828                                   ;   [start_sec_h] = first sector (high)  32 bit calculation.
13829                                   ;   ds = cs
13830                                   ;   es:di = transfer address
13831                                   ;   [rflag]=operation (2=read, 3=write)
13832                                   ;   [verify]=1 for verify after write
13833                                   ;
13834                                   ;   if successful carry flag = 0
13835                                   ;     else cf=1 and al contains error code
13836                                   ; ---------------------------------------------------------------------------
13837
13838                                           ; 12/12/2023
13839                                           ; ds = biosdata segment (cs = bioscode segment)
13840                                   diskrd:
13841                                           ;mov    ds:rflag, 2    ; romread
13842                                           ; 19/10/2022
13843 00000A0C C606[2001]02                    mov     byte [rflag], 2 ; romread
13844
13845                                   ; =============== S U B   R O U T I N E =======================================
13846
13847                                           ; 22/12/2023 - Retro DOS v5.0
13848                                           ; PCDOS 7.1 IBMBIO.COM - BIOSCODE:0A30h
13849                                   ; 22/12/2023
13850                                   %if 0
13851                                           ; 19/10/2022
13852                                   diskio:
13853                                           mov     bx, di         ; es:bx = transfer address
13854                                           mov     [xfer_seg], es ; save transfer       segment
13855                                           call    SetDrive
13856                                           mov     al, [es:di+10h]        ; [es:di+BDS.media]
13857                                           mov     [medbyt], al
13858                                           ;jcxz   short ioret
13859                                           jcxz    ioret
13860
13861                                   ; see if the media is formatted or not by checking the flags field in
13862                                   ; in the bds. if it is unformatted we cannot allow i/o, so we should
13863                                   ; go to the error exit at label unformatteddrive.
13864
13865                                           test    byte [es:di+24h], 2
13866                                           ;test   byte ptr es:[di+24h], 2       ; [es:di+BDS.flags+1]
13867                                                                   ; unformatted_media
13868                                           jnz     short unformatteddrive
13869                                           mov     [seccnt], cx   ; save sector count
13870                                           mov     [spsav], sp    ; save sp
13871
13872                                   ; ensure that we are trying to access valid sectors on the drive
13873
13874                                           mov     ax, dx
13875                                           xor     si, si ; 0
13876                                           add     dx, cx
13877                                           ;adc    si, 0
13878                                           ; 02/09/2023 (PCDOS 7.1)
13879                                           rcl     si, 1
13880                                           cmp     word [es:di+0Eh], 0 ; [es:di+BDS.totalsecs16]
13881                                                                   ; 32 bit sector ?
13882                                           jz      short sanity32
13883                                           ;cmp    si, 0
13884                                           ; 02/09/2023
13885                                           or      si, si
13886                                           jnz     short baddrive
13887                                           cmp     dx, [es:di+0Eh]        ; [es:di+BDS.totalsecs16]
13888                                           ja      short baddrive
13889                                           jmp     short sanityok
13890                                   ; ---------------------------------------------------------------------------
13891
13892                                   sanity32:
13893                                           add     si, [start_sec_h]
13894                                           cmp     si, [es:di+1Dh]        ; [es:di+BDS.totalsecs32+2]
13895                                           jb      short sanityok
```

```
                                            ja      short baddrive
                                            cmp     dx, [es:di+1Bh]       ; [es:di+BDS.totalsecs32]
                                            ja      short baddrive
                                    sanityok:
                                            mov     dx, [start_sec_h]
                                            add     ax, [es:di+17h]       ; [es:di+BDS.hiddensecs]
                                            adc     dx, [es:di+19h]       ; [es:di+BDS.hiddensecs+2]

                                    ; now dx;ax have the physical first sector.
                                    ; since the following procedures is going to destroy ax, let's
                                    ; save it temporarily to saved_word.

                                            mov     [saved_word], ax ; save the sector number (low)

                                    ; set up pointer to disk base table in [dpt]. we cannot assume that iosetup
                                    ; will do it because we will skip the set up stuff with hard disks.

                                            push    es
                                            ;mov    es, [zeroseg]
                                            ; 02/09/2023
                                            xor     si, si ; 0
                                            mov     es, si
                                            les     si, [es:DSKADR]
                                            ;les    si, es:78h     ; [es:DSKADR]
                                                                   ; current disk parm table
                                            mov     [dpt], si
                                            mov     [dpt+2], es
                                            pop     es
                                            test    byte [es:di+23h], 1 ; [es:di+BDS.flags]
                                                                   ; fnon_removable
                                            jnz     short skip_setup
                                            call    checksingle

                                    ; check to see if we have previously noted a change line. the routine
                                    ; returns if everything is ok. otherwise, it pops off the stack and returns
                                    ; the proper error code.

                                            cmp     byte [fhave96], 0 ; do we have changeline support?
                                            jz      short diskio_nochangeline ; brif not
                                            call    checklatchio   ; will do a sneaky pop stack return
                                                                   ; if a disk error occurs
                                    diskio_nochangeline:
                                            call    iosetup        ; set up tables       and variables for i/o

                                    ; now the settle values are correct for the following code

                                    skip_setup:

                                    ; 32 bit sector calculation.
                                    ; dx:[saved_word] = starting sector number.

                                            mov     ax, dx
                                            xor     dx, dx
                                            ;div    word [es:di+13h] ; [es:di+BDS.secpertrack]
                                                                    ; divide by sec per track
                                            ; 02/09/2023
                                            mov     cx, [es:di+13h]
                                            div     cx
                                            mov     [temp_h], ax
                                            mov     ax, [saved_word]
                                            div     cx ; 02/09/2023
                                            ;div    word [es:di+13h] ; [es:di+BDS.secpertrack]
                                                                   ; now, [temp_h]:ax = track #, dx = sector
                                            ;inc    dl             ; sector number       is 1 based.
                                            ; 18/12/2022
                                            inc     dx
                                            mov     [cursec], dl   ; save current sector
                                            mov     cx, [es:di+15h]       ; es:di+BDS.heads]
                                                                   ; get number of       heads
                                            push    ax
                                            xor     dx, dx
                                            mov     ax, [temp_h]   ; divide tracks       by heads per cylinder
                                            div     cx
                                            mov     [temp_h], ax
                                            pop     ax
                                            div     cx             ; now, [temp_h]:ax = cylinder #, dx = head
                                            cmp     word [temp_h], 0
                                            ja      short baddrive_brdg
                                            cmp     ax, 1024       ; 2^10 currently maxium      for track #.
                                            ja      short baddrive_brdg
                                            mov     [curhd], dl    ; save current head
                                            mov     [curtrk], ax   ; save current track

                                    ; we are now set up for the i/o. normally, we consider the dma boundary
                                    ; violations here. not true. we perform the operation as if everything is
                                    ; symmetric; let the int 13 handler worry about the dma violations.

                                            mov     ax, [seccnt]
                                            call    block          ; (cas - call/ret)
                                            ;call   done
                                            ;retn
                                            ; 18/12/2022
                                            jmp     done
                                    %else
                                            ;;;     ; 22/12/2023
                                    diskio:
00000A11 89FB                               mov     bx, di         ; al = drive number
                                                                   ; cx = sector count
                                                                   ; dx = first sector (low)
                                                                   ; [start_sec_h] = first sector (high)
                                                                   ;
                                                                   ; es:bx = transfer address
00000A13 8C06[A804]                         mov     [xfer_seg], es ; save transfer segment
00000A17 E88AFB                             call    SetDrive
00000A1A 268A4510                           mov     al, [es:di+10h]       ; [es:di+BDS.media]
00000A1E A2[1F01]                           mov     [medbyt], al
00000A21 E3D8                               jcxz    ioret

                                    ; see if the media is formatted or not by checking the flags field in
                                    ; in the bds. if it is unformatted we cannot allow i/o, so we should
                                    ; go to the error exit at label unformatteddrive.

00000A23 26F6454002                         test    byte [es:di+40h], 2 ; [es:di+BDS.flags+1]
                                                                   ; unformatted_media
00000A28 75CE                               jnz     short unformatteddrive
00000A2A 890E[2201]                         mov     [seccnt], cx   ; save sector count
00000A2E 8926[3501]                         mov     [spsav], sp    ; save sp

                                    ; ensure that we are trying to access valid sectors on the drive

00000A32 89D0                               mov     ax, dx
00000A34 31F6                               xor     si, si ; 0
00000A36 01CA                               add     dx, cx
00000A38 D1D6                               rcl     si, 1
```

```
14020 00000A3A 26837D0E00                       cmp     word [es:di+0Eh], 0 ; [es:di+BDS.totalsecs16]
14021                                                                       ; > 32 bit sector ?
14022 00000A3F 740E                             jz      short sanity32
14023 00000A41 09F6                             or      si, si
14024 00000A43 7506                             jnz     short baddrive
14025 00000A45 263B550E                         cmp     dx, [es:di+0Eh]         ; [es:di+BDS.totalsecs16]
14026                                           ;ja     short baddrive
14027                                           ;jmp    short sanityok
14028                                           ; 22/12/2023
14029 00000A49 7616                             jna     short sanityok
14030                                   ; 29/12/2023
14031                                   ; 22/12/2023
14032                                   ;%if 1
14033                                   ; ---------------------------------------------------------------------------
14034
14035                                   baddrive:
14036 00000A4B B008                             mov     al, 8           ; sector not found
14037                                           ;jmp    short baddrive_ret
14038                                   ; ---------------------------------------------------------------------------
14039                                   ;unformatteddrive:
14040                                           ;mov    al, 7           ; unknown media
14041                                   baddrive_ret:
14042 00000A4D F9                               stc
14043                                   ;ioret:
14044 00000A4E C3                               retn
14045                                   ;%endif
14046
14047                                   ; ---------------------------------------------------------------------------
14048
14049                                   sanity32:
14050 00000A4F 0336[9C04]                       add     si, [start_sec_h]
14051 00000A53 263B751D                         cmp     si, [es:di+1Dh]         ; [es:di+BDS.totalsecs32+2]
14052 00000A57 7208                             jb      short sanityok
14053 00000A59 77F0                             ja      short baddrive
14054 00000A5B 263B551B                         cmp     dx, [es:di+1Bh]         ; [es:di+BDS.totalsecs32]
14055 00000A5F 77EA                             ja      short baddrive
14056                                   sanityok:
14057 00000A61 8B16[9C04]                       mov     dx, [start_sec_h]
14058 00000A65 26034517                         add     ax, [es:di+17h]         ; [es:di+BDS.hiddensecs]
14059 00000A69 26135519                         adc     dx, [es:di+19h]         ; [es:di+BDS.hiddensecs+2]
14060
14061                                   ; now dx;ax have the physical first sector.
14062                                   ; since the following procedures is going to destroy ax, let's
14063                                   ; save it temporarily to saved_word.
14064
14065 00000A6D A3[9E04]                          mov     [saved_word], ax ; save the sector number (low)
14066
14067                                   ; set up pointer to disk base table in [dpt]. we cannot assume that iosetup
14068                                   ; will do it because we will skip the set up stuff with hard disks.
14069
14070 00000A70 06                                push    es
14071 00000A71 31F6                             xor     si, si ; 0
14072 00000A73 8EC6                             mov     es, si
14073                                           ;les    si, dword ptr es:78h
14074 00000A75 26C4367800                       les     si, [es:78h]    ; INT 1Eh vector address
14075                                                                   ; [es:DSKADR] - current disk parm table
14076 00000A7A 8936[2D01]                       mov     [dpt], si
14077 00000A7E 8C06[2F01]                       mov     [dpt+2], es
14078 00000A82 07                               pop     es
14079 00000A83 26F6453F01                       test    byte [es:di+3Fh], 1 ; [es:di+BDS.flags], fnon_removable
14080 00000A88 7510                             jnz     short chk_13h_ext_flag
14081 00000A8A E8EAFE                           call    checksingle
14082
14083                                   ; check to see if we have previously noted a change line. the routine
14084                                   ; returns if everything is ok. otherwise, it pops off the stack and returns
14085                                   ; the proper error code.
14086
14087 00000A8D 803E[7700]00                      cmp     byte [fhave96], 0 ; do we have changeline support?
14088 00000A92 7403                             jz      short diskio_nochangeline ; brif not
14089 00000A94 E8D210                           call    checklatchio    ; will do a sneaky pop stack return
14090                                                                   ; if a disk error occurs
14091                                   diskio_nochangeline:
14092 00000A97 E8E000                           call    iosetup         ; set up tables and variables for i/o
14093
14094                                   chk_13h_ext_flag:
14095 00000A9A 26F6454004                       test    byte [es:di+40h], 4 ; [es:di+BDS.flags+1], fLBArw
14096                                                                   ; LBA read/write flag
14097 00000A9F 7539                             jnz     short set_lbarw_1
14098                                           ;jmp    skip_setup
14099                                           ; 22/12/2023
14100                                   ; ---------------------------------------------------------------------------
14101
14102                                   ; now the settle values are correct for the following code
14103
14104                                   skip_setup:
14105
14106                                   ; 32 bit sector calculation.
14107                                   ; dx:[saved_word] = starting sector number.
14108
14109                                           ;push   bp ; ! (not necessary) ; 22/12/2023
14110 00000AA1 92                               xchg    ax, dx ; mov ax,dx
14111 00000AA2 31D2                             xor     dx, dx
14112 00000AA4 268B4D13                         mov     cx, [es:di+13h]         ; [es:di+BDS.secpertrack]
14113                                                                   ; divide by sec per track
14114 00000AA8 F7F1                             div     cx
14115 00000AAA 95                               xchg    ax, bp ; mov bp,ax
14116 00000AAB A1[9E04]                         mov     ax, [saved_word]
14117 00000AAE F7F1                             div     cx              ; [es:di+BDS.secpertrack]
14118                                                                   ; now, bp:ax = track #, dx = sector
14119                                                                   ; sector number is 1 based.
14120 00000AB0 42                               inc     dx
14121 00000AB1 8816[3101]                       mov     [cursec], dl    ; save current sector
14122 00000AB5 268B4D15                         mov     cx, [es:di+15h] ; [es:di+BDS.heads]
14123                                                                   ; get number of heads
14124                                           ; 22/12/2023
14125                                           ;push   ax ; *
14126 00000AB9 31D2                             xor     dx, dx
14127 00000ABB 95                               xchg    ax, bp ; bp = *         ; divide tracks by heads per cylinder
14128 00000ABC F7F1                             div     cx
14129 00000ABE 95                               xchg    ax, bp ; ax = *, bp = **
14130                                           ;pop    ax ; *
14131 00000ABF F7F1                             div     cx              ; now, bp:ax = cylinder #, dx = head
14132 00000AC1 09ED                             or      bp, bp ; ** = 0 ?
14133                                           ;pop    bp ; ! ; 22/12/2023
14134                                           ;jnz    short baddrive_brdg
14135                                           ; 22/12/2023
14136 00000AC3 7586                             jnz     short baddrive
14137
14138                                           ;cmp    ax, 1024        ; 2^10 currently maximum for track #.
14139                                           ;jnb    short baddrive_brdg
14140                                           ; 22/12/2023
14141 00000AC5 80FC04                           cmp     ah, 4   ; if ax >= 4*256 (1024)
14142 00000AC8 7381                             jnb     short baddrive
14143
```

```
00000ACA 8816[3201]                     mov     [curhd], dl     ; save current head
00000ACE A3[3301]                       mov     [curtrk], ax    ; save current track

                                ; we are now set up for the i/o. normally, we consider the dma boundary
                                ; violations here. not true. we perform the operation as if everything is
                                ; symmetric; let the int 13 handler worry about the dma violations.

00000AD1 A1[2201]                       mov     ax, [seccnt]
00000AD4 E81F01                         call    block
                                        ;call   done
                                        ;retn
                                        ; 22/12/2023
00000AD7 E9E500                         jmp     done

                                ; ---------------------------------------------------------------------------

                                set_lbarw_1:
00000ADA A1[9E04]                       mov     ax, [saved_word] ; check for mini disk
                                                                 ; (logical dos drive/partition)
00000ADD 26837D7901                     cmp     word [es:di+79h], 1 ; [di+BDS.bdsm_ismini]
                                                                    ; logical dos partition
00000AE2 750F                           jnz     short set_lbarw_2 ; not a logical dos partition/drive
00000AE4 26837D7B00                     cmp     word [es:di+7Bh], 0 ; [di+BDS.bdsm_hidden_trks] (> 0)
00000AE9 7408                           jz      short set_lbarw_2
00000AEB 26034517                       add     ax, [es:di+17h]       ; [es:di+BDS.hiddensecs]
00000AEF 26135519                       adc     dx, [es:di+19h]       ; [es:di+BDS.hiddensecs+2]

                                set_lbarw_2:
00000AF3 2EA3[040A]                     mov     [cs:dap_lba_value], ax
00000AF7 2E8916[060A]                   mov     [cs:dap_lba_value+2], dx
00000AFC 2E891E[000A]                   mov     [cs:dap_trans_buf], bx
00000B01 A1[A804]                       mov     ax, [xfer_seg]
00000B04 2EA3[020A]                     mov     [cs:dap_trans_buf+2], ax
00000B08 A1[2201]                       mov     ax, [seccnt]
00000B0B 2EA3[FE09]                     mov     [cs:dap_block_cnt], ax
00000B0F BD0500                         mov     bp, 5
00000B12 892E[A304]                     mov     [vretry_cnt], bp ; verify op. retry cnt for write-verify
00000B16 892E[A504]                     mov     [soft_ecc_cnt], bp ; soft ecc error retry count

                                set_lbarw_3:
00000B1A 268A5504                       mov     dl, [es:di+4]  ; [es:di+BDS.drivenum]
00000B1E 8A26[2001]                     mov     ah, [rflag]    ; get read/write indicator
00000B22 80C440                         add     ah, 40h
00000B25 30C0                           xor     al, al
00000B27 1E                             push    ds
00000B28 0E                             push    cs
00000B29 1F                             pop     ds
00000B2A BE[FC09]                       mov     si, LBA_Packet
00000B2D CD13                           int     13h             ; LBA read/write
00000B2F 1F                             pop     ds
00000B30 731A                           jnc     short set_lbarw_7
00000B32 E8AC02                         call    again
                                set_lbarw_9:
00000B35 7503                           jnz     short set_lbarw_4
00000B37 E92B02                         jmp     harderr
                                ; ---------------------------------------------------------------------------

                                set_lbarw_4:
                                ;set_lbarw_9:      ; 22/12/2023
00000B3A 80FCCC                         cmp     ah, 0CCh        ; Write fault (hard disk)
00000B3D 7505                           jnz     short set_lbarw_5
00000B3F BD0100                         mov     bp, 1
                                        ;jmp    short set_lbarw_6
                                        ; 17/04/2024
00000B42 EBD6                           jmp     short set_lbarw_3
                                ; ---------------------------------------------------------------------------

                                set_lbarw_5:
                                set_lbarw_10:       ; 22/12/2023
00000B44 C706[A504]0500                 mov     word [soft_ecc_cnt], 5 ; soft ecc error retry count
                                set_lbarw_6:
                                set_lbarw_11:
00000B4A EBCE                           jmp     short set_lbarw_3
                                ; ---------------------------------------------------------------------------

                                set_lbarw_7:
00000B4C 813E[2001]0301                 cmp     word [rflag], 103h
00000B52 7523                           jnz     short set_lbarw_12
00000B54 B444                           mov     ah, 44h
00000B56 1E                             push    ds
00000B57 0E                             push    cs
00000B58 1F                             pop     ds
00000B59 CD13                           int     13h             ; DISK - IBM/MS Extension - VERIFY SECTORS
                                                                ;  (DL - drive, [SI - disk address packet)
00000B5B 1F                             pop     ds
00000B5C 7319                           jnc     short set_lbarw_12
00000B5E 80FC11                         cmp     ah, 11h         ; ECC corrected data error (soft error - retried OK )
00000B61 7506                           jnz     short set_lbarw_8
00000B63 FF0E[A504]                     dec     word [soft_ecc_cnt]
                                ;set_lbarw_8:
00000B67 740E                           jz      short set_lbarw_12
                                set_lbarw_8:
00000B69 E8CF07                         call    ResetDisk
00000B6C 80FC11                         cmp     ah, 11h
00000B6F 74D9                           jz      short set_lbarw_11
00000B71 FF0E[A304]                     dec     word [vretry_cnt]
                                        ;jnz    short set_lbarw_9
                                        ;jmp    harderr
                                        ; 22/12/2023
00000B75 EBBE                           jmp     short set_lbarw_9

                                ; ---------------------------------------------------------------------------
                                ;               ; 22/12/2023
                                ;set_lbarw_9:
                                ;               cmp     ah, 0CCh
                                ;               jnz     short set_lbarw_10
                                ;               mov     bp, 1
                                ;               jmp     short set_lbarw_11
                                ; ---------------------------------------------------------------------------
                                ;               ; 22/12/2023
                                ;set_lbarw_10:
                                ;               mov     word [soft_ecc_cnt], 5 ; soft ecc error retry count
                                ;set_lbarw_11:
                                ;               jmp     short set_lbarw_3
                                ; ---------------------------------------------------------------------------

                                set_lbarw_12:
00000B77 31C0                           xor     ax, ax
                                skip_dpt_setting:  ; 23/12/2023
00000B79 C3                             retn
                                        ;;;     ; 22/12/2023
                                %endif

                                ; ---------------------------------------------------------------------------
```

```
14268
14269                                           ; 22/12/2023
14270                                   ;baddrive_brdg:
14271                                           ;jmp    baddrive
14272
14273                                   ; ============== S U B   R O U TI N E =======================================
14274
14275                                   ;--------------------------------------------------------------
14276                                   ; set the drive-last-accessed flag for diskette only.
14277                                   ; we know that the hard disk will not be removed.
14278                                   ; es:di -> current bds.
14279                                   ; ds -> Bios_Data
14280                                   ; ax,cx,si are destroyed.
14281                                   ;--------------------------------------------------------------
14282
14283                                           ; 23/12/2023 - Retro DOS v5.0
14284
14285                                           ; 19/10/2022
14286                                   iosetup:
14287 00000B7A 268A4504                           mov     al, [es:di+4]  ; [es:di+BDS.drivenum]
14288 00000B7E A2[1E01]                           mov     [tim_drv], al  ; save drive letter
14289
14290                                   ; determine proper head settle values
14291
14292 00000B81 803E[A905]00                       cmp     byte [media_set_for_format], 0
14293 00000B86 75F1                               jnz     short skip_dpt_setting
14294 00000B88 A0[2C01]                           mov     al, [eot]      ; fetch up eot before changing ds
14295 00000B8B 1E                                 push    ds
14296 00000B8C C536[2D01]                         lds     si, [dpt]      ; get pointer to disk base table
14297 00000B90 884404                             mov     [si+4], al
14298
14299                                           ;; 23/12/2023
14300                                           ;mov    ah, al
14301                                           ;mov    al, [si+10]    ; [si+DISK_PARMS.DISK_MOTOR_STRT]
14302                                           ;;mov   ah, [si+4]     ; [si+DISK_PARMS.DISK_EOT]
14303                                           ;pop    ds
14304                                           ;mov    [motorstartup], al
14305                                           ;mov    [save_eot], ah
14306                                           ; 06/04/2024
14307 00000B93 8A640A                           mov     ah,[si+10]
14308 00000B96 1F                               pop     ds
14309 00000B97 8826[2601]                       mov     [motorstartup], ah
14310 00000B9B A2[2B01]                         mov     [save_eot], al
14311
14312                                   ; for 3.5" drives, both external as well as on the k09, we need to set the
14313                                   ; motor start time to 4. this checking for every i/o is going to affect
14314                                   ; performance across the board, but is necessary!!
14315
14316 00000B9E 1E                                 push    ds
14317 00000B9F C536[2D01]                         lds     si, [dpt]      ; get pointer to disk base table
14318                                             ; 23/12/2023  - Retro DOS v5.0
14319 00000BA3 26807D3E02                         cmp     byte [es:di+3Eh], 2 ; (PCDOS 7.1 IBMBIO.COM)
14320                                             ;cmp    byte [es:di+22h], 2 ; [es:di+BDS.formfactor]
14321                                                                         ; ffSmall
14322 00000BA8 7505                               jnz     short motor_start_ok
14323 00000BAA B004                               mov     al, 4
14324 00000BAC 86440A                             xchg    al, [si+10]    ; [si+DISK_PARMS.DISK_MOTOR_STRT]
14325                                   motor_start_ok:
14326
14327                                   ; ds:si now points to disk parameter table.
14328                                   ; get current settle and set fast settle
14329
14330                                           ;xor    al, al
14331                                           ;inc    al             ; ibm wants fast settle      to be 1
14332                                           ; 18/12/2022
14333 00000BAF 31C0                             xor     ax, ax
14334 00000BB1 40                               inc     ax
14335 00000BB2 864409                           xchg    al, [si+9]     ; [si+DISK_PARMS.DISK_HEAD_STTL]
14336                                                                   ; get settle and set up      for fast
14337 00000BB5 1F                               pop     ds
14338 00000BB6 A2[2701]                         mov     [settlecurrent], al
14339 00000BB9 B00F                             mov     al, 15         ; NORMSETTLE
14340                                                                   ; someone has diddled the settle
14341 00000BBB A2[2801]                         mov     [settleslow], al
14342                                           ; 23/12/2023
14343                                   ;skip_dpt_setting:
14344 00000BBE C3                               retn
14345
14346                                   ; ============== S U B   R O U TI N E =======================================
14347
14348                                   ;--------------------------------------------------------------
14349                                   ; set time of last access, and reset default values in the dpt.
14350                                   ;
14351                                   ;     note: trashes (at least) si
14352                                   ;--------------------------------------------------------------
14353
14354                                           ; 23/12/2023 - Retro DOS v5.0
14355
14356                                           ; 19/10/2022
14357                                   done:
14358                                           ;test   byte [es:di+23h], 1 ; [es:di+BDS.flags]
14359                                                                       ; fnon_removable
14360                                           ; 23/12/2023
14361 00000BBF 26F6453F01                       test    byte [es:di+3Fh], 1 ; (PCDOS 7.1 IBMBIO.COM)
14362 00000BC4 752F                             jnz     short ddbx     ; do not set for non-removable media
14363 00000BC6 E8F801                           call    set_tim
14364                                   ;diddleback:
14365                                   ; 09/12/2022
14366                                   diddle_back:
14367 00000BC9 9C                               pushf
14368 00000BCA 803E[A905]00                     cmp     byte [media_set_for_format], 0
14369 00000BCF 7523                             jnz     short nodiddleback
14370 00000BD1 50                               push    ax
14371 00000BD2 06                               push    es
14372 00000BD3 C436[2D01]                       les     si, [dpt]
14373 00000BD7 A0[2B01]                         mov     al, [save_eot]
14374 00000BDA 26884404                         mov     [es:si+4], al  ; [es:si+DISK_PARMS.DISK_EOT]
14375 00000BDE A0[2701]                         mov     al, [settlecurrent]
14376 00000BE1 8A26[2601]                       mov     ah, [motorstartup]
14377 00000BE5 26884409                         mov     [es:si+9], al  ; [es:si+DISK_PARMS.DISK_HEAD_STTL]
14378 00000BE9 26C6440302                       mov     byte [es:si+3], 2 ; [es:si+DISK_PARMS.DISK_SECTOR_SIZ]
14379 00000BEE 2688640A                         mov     [es:si+0Ah], ah       ; [es:si+DISK_PARMS.DISK_MOTOR_STRT]
14380 00000BF2 07                               pop     es
14381 00000BF3 58                               pop     ax
14382                                   nodiddleback:
14383 00000BF4 9D                               popf
14384                                   ddbx:
14385 00000BF5 C3                               retn
14386
14387                                   ; ============== S U B   R O U TI N E =======================================
14388
14389                                   ;--------------------------------------------------------------
14390                                   ;read the number of sectors specified in ax,
14391                                   ;handling track boundaries
```

```
                                ;es:di -> bds for this drive
                                ;--------------------------------------------------------------
                                
                                        ; 23/12/2023 - Retro DOS v5.0
                                
                                        ; 19/10/2022
                                block:
00000BF6 09C0                           or      ax, ax
00000BF8 74FB                           jz      short ddbx
                                        ; 23/12/2023
00000BFA 26F6453F01                     test    byte [es:di+3Fh], 1 ; (PCDOS 7.1 IBMBIO.COM)
                                        ;test   byte [es:di+23h], 1 ; [es:di+BDS.flags]
                                                                    ; fnon_removable
00000BFF 740D                           jz      short block_floppy
                                
                                ; check     to see if multi       track operation       is allowed. if not
                                ; we have to go    to the block_floppy below to break up the operation.
                                
00000C01 F606[A004]80                   test    byte [multrk_flag], 80h
                                        ;test   byte ptr ds:multrk_flag, 80h ; multrk_on
00000C06 7406                           jz      short block_floppy
00000C08 E82800                         call    Disk
00000C0B 31C0                           xor     ax, ax
00000C0D C3                             retn
                                ; ---------------------------------------------------------------------------
                                
                                block_floppy:
                                
                                ; read at most 1 track worth. perform minimization at sector / track
                                
00000C0E 268A4D13                       mov     cl, [es:di+19] ; [es:di+BDS.secpertrack]
                                        ;inc    cl
                                        ; 23/12/2023
00000C12 41                             inc     cx
00000C13 2A0E[3101]                     sub     cl, [cursec]
00000C17 30ED                           xor     ch, ch
00000C19 39C8                           cmp     ax, cx
00000C1B 7302                           jnb     short gotmin
00000C1D 89C1                           mov     cx, ax
                                gotmin:
                                
                                ; ax is the requested number of sectors to read
                                ; cx is the number that we can do on this track
                                
00000C1F 50                             push    ax
00000C20 51                             push    cx
00000C21 89C8                           mov     ax, cx
00000C23 E80D00                         call    Disk
00000C26 59                             pop     cx
00000C27 58                             pop     ax
                                
                                ; cx is the number of sectors just transferred
                                
00000C28 29C8                           sub     ax, cx          ; reduce sectors-remaining by last i/o
00000C2A D0E1                           shl     cl, 1
00000C2C 00CF                           add     bh, cl          ; adjust transfer address
00000C2E EBC6                           jmp     short block
                                dskerr_brdg:
00000C30 E9F400                         jmp     dskerr
                                
                                ; =============== S U B   R O U T I N E =======================================
                                
                                ; 15/10/2022
                                
                                ;--------------------------------------------------------------
                                ;perform disk i/o with retries
                                ; al = number of sectors (1-8, all on one track)
                                ; es:di point to drive parameters
                                ; xfer_seg:bx = transfer address
                                ;           (must not cross a 64k physical boundary)
                                ; [rflag] = 2 if read, 3 if write
                                ; [verify] = 0 for normal, 1 for verify after write
                                ;--------------------------------------------------------------
                                
                                        ; 18/04/2024
                                        ; 23/12/2023 - Retro DOS v5.0
                                        ; (PCDOS 7.1 IBMBIO.COM - BIOSCODE:0C74h)
                                
                                        ; 19/10/2022
                                Disk:
                                
                                ; Check for hard disk format and
                                ; if TRUE then set max error count to 2
                                
00000C33 BD0500                         mov     bp, 5           ; MAXERR
                                                                ; set up retry count
                                        ; 18/04/2024
                                        ;; 23/12/2023
                                        ;mov    cl, [es:di+3Fh]
                                        ;and    cx, 1
00000C36 26F6453F01                     test    byte [es:di+3Fh], 1
                                        ;test   byte [es:di+23h], 1
                                                                ; [es:di+BDS.flags], fnon_removable
00000C3B 7408                           jz      short GetRdWrInd
00000C3D 80FC04                         cmp     ah, 4           ; romverify ; Is this a       track verify?
00000C40 7403                           jz      short GetRdWrInd
00000C42 BD0200                         mov     bp, 2           ; This is not verify so only 1 retry
                                GetRdWrInd:
00000C45 892E[A304]                     mov     [vretry_cnt], bp ; verify op. retry cnt for write-verify
00000C49 892E[A504]                     mov     [soft_ecc_cnt], bp ; soft ecc error retry count.
00000C4D 8A26[2001]                     mov     ah, [rflag]     ; get read/write indicator
                                ;retry:
                                ; 09/12/2022
                                _retry:
00000C51 50                             push    ax
00000C52 8B16[3301]                     mov     dx, [curtrk]
                                        ;; 23/12/2023
                                        ;jcxz   disk_not_mini
                                        ; 18/04/2024
00000C56 26F6453F01                     test    byte [es:di+3Fh], 1
                                        ;test   byte [es:di+23h], 1
00000C5B 740B                           jz      short disk_not_mini
                                
                                        ; 23/12/2023
00000C5D 26837D7901                     cmp     word [es:di+79h], 1
                                        ;cmp    word [es:di+47h], 1 ; [es:di+BDS.bdsm_ismini]
                                                                    ; is this a mini disk? ((logical dos partition))
00000C62 7504                           jnz     short disk_not_mini ; no. continue to next.
                                        ; 23/12/2023
00000C64 2603557B                       add     dx, [es:di+7Bh]
                                        ;add    dx, [es:di+49h]       ; [es:di+BDS.bdsm_hidden_trks]
                                                                    ; add hidden trks.
                                disk_not_mini:
00000C68 D0CE                           ror     dh, 1
```

```
14516 00000C6A D0CE                             ror     dh, 1
14517 00000C6C 0A36[3101]                       or      dh, [cursec]
14518 00000C70 89D1                             mov     cx, dx
14519 00000C72 86E9                             xchg    ch, cl          ;  cl = sector, ch = cylinder
14520 00000C74 8A36[3201]                       mov     dh, [curhd]     ; load current head number and
14521 00000C78 268A5504                         mov     dl, [es:di+4]   ; physical drive number
14522                                                                   ; [es:di+BDS.drivenum]
14523                                           ; 23/12/2023
14524 00000C7C 26807D3E05                       cmp     byte [es:di+3Eh], 5
14525                                           ;cmp    byte [es:di+22h], 5 ; [es:di+BDS.formfactor], ffHardFile
14526 00000C81 7411                             jz      short do_fast  ; hard files use fast speed
14527
14528                                   ; if we have [step_drv] set to -1, we use the slow settle time.
14529                                   ; this helps when we have just done a reset disk operation and the head has
14530                                   ; been moved to another cylinder - the problem crops up with 3.5" drives.
14531
14532 00000C83 803E[7600]FF                        cmp     byte [step_drv], 0FFh ; -1
14533                                           ;jz     short do_writej
14534                                           ; 23/12/2023
14535 00000C88 746A                             jz      short do_write
14536 00000C8A 80FC02                           cmp     ah, 2           ; romread
14537 00000C8D 7405                             jz      short do_fast
14538 00000C8F 80FC04                           cmp     ah, 4           ; romverify
14539                                           ;jz     short do_fast
14540                                           ; 23/12/2023
14541 00000C92 7560                             jnz     short do_write
14542                                   ;do_writej:
14543
14544                                   ; reads always fast, unless we have just done a disk reset operation
14545
14546                                           ;jmp    short do_write ; reads always fast
14547                                   ; ---------------------------------------------------------------------------
14548
14549                                   do_fast:
14550 00000C94 E80501                           call    fastspeed       ; change settle       mode
14551                                   testerr:
14552 00000C97 7297                             jb      short dskerr_brdg
14553
14554                                           ; 23/12/2023 Retro DOS v5.0
14555                                           ; (PCDOS 7.1 IBMBIO.COM)
14556 00000C99 83FD05                           cmp     bp, 5           ; is there retry ?
14557 00000C9C 7505                             jnz     short testerror         ; yes
14558 00000C9E 80FCBB                           cmp     ah, 0BBh        ; Undefined error (hard disk)
14559 00000CA1 748D                             jz      short dskerr_brdg
14560                                   testerror:
14561
14562                                   ; set drive and track of last access
14563
14564 00000CA3 8816[7600]                       mov     [step_drv], dl
14565                                           ; 23/12/2023
14566 00000CA7 26886D78                         mov     [es:di+78h], ch
14567                                           ;mov    [es:di+46h], ch         ; [es:di+BDS.track]
14568                                   no_set:
14569                                           ;cmp    word [wrtverify], 103h
14570 00000CAB 813E[2001]0301                   cmp     word [rflag], 103h ; check for write and verify
14571 00000CB1 7452                             jz      short doverify
14572                                   noverify:
14573 00000CB3 58                               pop     ax
14574
14575                                   ; check the flags word in the bds to see if the drive is non removable
14576                                   ; if not we needn't do anything special
14577                                   ; if it is a hard disk then check to see if multi-track operation
14578                                   ; is specified. if specified we don't have to calculate for the next
14579                                   ; track since we are already done. so we can go to the exit of this routine.
14580
14581                                           ; 23/12/2023
14582 00000CB4 26F6453F01                       test    byte [es:di+3Fh], 1
14583                                           ;test   byte [es:di+23h], 1 ; [es:di+BDS.flags]
14584                                                                   ; fnon_removable
14585 00000CB9 7407                             jz      short its_removable
14586 00000CBB F606[A004]80                     test    byte [multrk_flag], 80h ; multrk_on
14587 00000CC0 7530                             jnz     short disk_ret
14588                                   its_removable:
14589 00000CC2 80E13F                           and     cl, 3Fh         ; eliminate cylinder bits from sector
14590 00000CC5 30E4                             xor     ah, ah
14591 00000CC7 2906[2201]                       sub     [seccnt], ax    ; reduce count of sectors to go      next sector
14592 00000CCB 00C1                             add     cl, al
14593 00000CCD 880E[3101]                       mov     [cursec], cl
14594 00000CD1 263A4D13                         cmp     cl, [es:di+13h]         ; [es:di+BDS.secpertrack]
14595                                                                   ; see if sector/track limit reached
14596 00000CD5 761B                             jbe     short disk_ret
14597 00000CD7 C606[3101]01                     mov     byte [cursec], 1 ; start with first sector of next track
14598 00000CDC 8A36[3201]                       mov     dh, [curhd]
14599 00000CE0 FEC6                             inc     dh
14600 00000CE2 263A7515                         cmp     dh, [es:di+15h]         ; [es:di+BDS.heads]
14601 00000CE6 7206                             jb      short noxor
14602 00000CE8 30F6                             xor     dh, dh
14603 00000CEA FF06[3301]                       inc     word [curtrk]
14604                                   noxor:
14605 00000CEE 8836[3201]                       mov     [curhd], dh
14606                                   disk_ret:
14607 00000CF2 F8                               clc
14608 00000CF3 C3                               retn
14609                                   ; ---------------------------------------------------------------------------
14610
14611                                   ; 15/10/2022
14612
14613                                   ; 24/12/2023 - Retro DOS v5.0
14614
14615                                   ;--------------------------------------------------------------
14616                                   ; the request is for write. determine if we are talking about
14617                                   ; the same track and drive. if so, use the fast speed.
14618                                   ;--------------------------------------------------------------
14619
14620                                   do_write:
14621 00000CF4 3A16[7600]                       cmp     dl, [step_drv]
14622 00000CF8 7506                             jnz     short do_norm  ; we have changed drives
14623                                           ; 24/12/2023
14624 00000CFA 263A6D78                         cmp     ch, [es:di+78h]
14625                                           ;cmp    ch, [es:di+46h]         ; [es:di+BDS.track]
14626 00000CFE 7494                             jz      short do_fast  ; we are still on the same track
14627                                   do_norm:
14628 00000D00 E87500                           call    normspeed
14629 00000D03 EB92                             jmp     short testerr
14630                                   ; ---------------------------------------------------------------------------
14631
14632                                   ;--------------------------------------------------------------
14633                                   ; we have a verify request also. get state info and go verify
14634                                   ;--------------------------------------------------------------
14635
14636                                   doverify:
14637 00000D05 58                               pop     ax
14638 00000D06 50                               push    ax
14639 00000D07 B404                             mov     ah, 4
```

```
14640 00000D09 E89000                           call    fastspeed
14641 00000D0C 73A5                             jnb     short noverify
14642
14643                                   ; check the error returned in ah to see if it is a soft ecc error.
14644                                   ; if it is not we needn't do anything special. if it is a soft
14645                                   ; ecc error then decrement the soft_ecc_cnt error retry count. if
14646                                   ; this retry count becomes 0 then we just ignore the error and go to
14647                                   ; no_verify but if we can still try then we call the routine to reset
14648                                   ; the disk and go to dskerr1 to retry the operation.
14649
14650 00000D0E 80FC11                           cmp     ah, 11h         ; soft ecc error ?
14651 00000D11 750B                             jnz     short not_softecc_err
14652 00000D13 FF0E[A504]                       dec     word [soft_ecc_cnt]
14653 00000D17 749A                             jz      short noverify ; no more retry
14654 00000D19 E81F06                           call    ResetDisk       ; reset disk
14655 00000D1C EB3E                             jmp     short dskerr1  ; retry
14656                                   ; ---------------------------------------------------------------------------
14657
14658                                   not_softecc_err:                ; other error.
14659 00000D1E E81A06                           call    ResetDisk
14660 00000D21 FF0E[A304]                       dec     word [vretry_cnt]
14661 00000D25 EB1C                             jmp     short dskerr0
14662                                   ; ---------------------------------------------------------------------------
14663
14664                                   ;--------------------------------------------------------------
14665                                   ; need to special case the change-line error ah=06h.
14666                                   ; if we get this, we need to return it.
14667                                   ;--------------------------------------------------------------
14668
14669                                   dskerr:
14670 00000D27 803E[7700]00                     cmp     byte [fhave96], 0     ; do we have changeline       support?
14671 00000D2C 7403                             jz      short dskerr_nochangeline ; brif not
14672 00000D2E E8BE0E                           call    checkio
14673                                   dskerr_nochangeline:
14674 00000D31 803E[A704]01                     cmp     byte [multitrk_format_flag], 1 ; multi trk format request?
14675 00000D36 7508                             jnz     short dochkagain ; no more retry.
14676 00000D38 BD0100                           mov     bp, 1
14677 00000D3B C606[A704]00                     mov     byte [multitrk_format_flag], 0 ; clear the flag.
14678                                   dochkagain:
14679 00000D40 E89E00                           call    again
14680                                   dskerr0:
14681 00000D43 7420                             jz      short harderr
14682                                           ; 24/12/2023
14683 00000D45 26F6453F01                       test    byte [es:di+3Fh], 1
14684                                           ;test   byte [es:di+23h], 1 ; [es:di+BDS.flags]
14685                                                                   ; fnon_removable
14686 00000D4A 7505                             jnz     short skip_timeout_chk
14687 00000D4C 80FC80                           cmp     ah, 80h         ; timeout?
14688 00000D4F 7414                             jz      short harderr
14689                                   skip_timeout_chk:
14690 00000D51 80FCCC                           cmp     ah, 0CCh        ; write fault error?
14691 00000D54 740A                             jz      short write_fault_err ;       then, don't retry.
14692 00000D56 C706[A504]0500                   mov     word [soft_ecc_cnt], 5 ; MAXERR
14693                                                                   ; set soft_ecc_cnt back       to maxerr
14694                                   dskerr1:
14695 00000D5C 58                               pop     ax              ; restore sector count
14696                                           ;jmp    retry
14697                                           ; 09/12/2022
14698 00000D5D E9F1FE                           jmp     _retry
14699                                   ; ---------------------------------------------------------------------------
14700
14701                                   write_fault_err:
14702 00000D60 BD0100                           mov     bp, 1           ; just retry only once
14703                                                                   ; for write fault error.
14704 00000D63 EBF7                             jmp     short dskerr1
14705
14706                                           ; fall into harderr
14707                                   ; ---------------------------------------------------------------------------
14708
14709                                   ; entry point for routines that call maperror themselves
14710
14711                                   harderr:
14712 00000D65 E84100                           call    maperror
14713                                   harderr2:
14714 00000D68 C606[1E01]FF                     mov     byte [tim_drv], 0FFh
14715                                                                   ; force a media check through rom
14716 00000D6D 8B0E[2201]                       mov     cx, [seccnt]    ; get count of sectors to go
14717 00000D71 8B26[3501]                       mov     sp, [spsav]     ; recover entry       stack pointer
14718
14719                                   ; since we are performing a non-local goto, restore the disk parameters
14720
14721                                           ;jmp    diddleback
14722                                           ; 09/12/2022
14723 00000D75 E951FE                           jmp     diddle_back
14724
14725                                   ; =============== S U B   R O U T I N E =======================================
14726
14727                                   ; change settle value from settlecurrent to whatever is appropriate
14728                                   ; note that this routine is never called for a fixed disk.
14729
14730                                           ; 19/10/2022
14731                                   normspeed:
14732 00000D78 803E[A905]00                     cmp     byte [media_set_for_format], 0
14733 00000D7D 751D                             jnz     short fastspeed
14734 00000D7F 06                               push    es
14735 00000D80 50                               push    ax
14736 00000D81 A0[2801]                         mov     al, [settleslow]
14737 00000D84 C436[2D01]                       les     si, [dpt]       ; current disk parm table
14738 00000D88 26884409                         mov     [es:si+9], al  ; [es:si+DISK_PARMS.DISK_HEAD_STTL]
14739 00000D8C 58                               pop     ax
14740 00000D8D 07                               pop     es
14741 00000D8E E80B00                           call    fastspeed
14742                                           ; 24/12/2023
14743                                           ;push   es
14744                                           ;les    si, [dpt]
14745                                           ;mov    byte [es:si+9], 1 ; [es:si+DISK_PARMS.DISK_HEAD_STTL]
14746                                           ;                       ; 1 is fast settle value
14747                                           ;pop    es
14748 00000D91 1E                               push    ds
14749 00000D92 C536[2D01]                       lds     si, [dpt]
14750 00000D96 C6440901                         mov     byte [si+9], 1
14751 00000D9A 1F                               pop     ds
14752
14753 00000D9B C3                               retn
14754
14755                                   ; =============== S U B   R O U T I N E =======================================
14756
14757                                   ; if the drive has been marked as too big (i.e. starting sector of the
14758                                   ; partition is > 16 bits, then always return drive not ready.
14759
14760                                           ; 24/12/2023 - Retro DOS v5.0
14761                                           ; (PCDOS 7.1 IBMBIO.COM - BIOSCODE:0DDDh)
14762                                   fastspeed:
14763                                           ;;test  byte [es:di+3Bh], 80h ; [es:di+BDS.fatsiz]
```

```
14764                                   ;test  byte [es:di+1Fh], 80h ; [es:di+BDS.fatsiz]
14765                                   ;                       ; ftoobig
14766                                   ;jnz   short notready
14767 00000D9C 06                       push   es
14768 00000D9D 8E06[A804]               mov    es, [xfer_seg]
14769 00000DA1 CD13                     int    13h             ; DISK -
14770 00000DA3 8C06[A804]               mov    [xfer_seg], es
14771 00000DA7 07                       pop    es
14772 00000DA8 C3                       retn
14773                                ; ---------------------------------------------------------------------------
14774                                ;          ; 24/12/2023
14775                                ;notready:
14776                                   ;stc
14777                                   ;mov   ah, 80h
14778                                   ;retn
14779
14780                                ; =============== S U B   R O U T I N E =======================================
14781
14782                                ; map error returned by rom in ah into corresponding code to be returned to
14783                                ; dos in al. trashes di. guaranteed to set carry.
14784
14785                                maperror:
14786 00000DA9 51                       push   cx
14787 00000DAA 06                       push   es
14788 00000DAB 1E                       push   ds              ; set es=Bios_Data
14789 00000DAC 07                       pop    es
14790 00000DAD 88E0                     mov    al, ah          ; put error code in al
14791 00000DAF A2[4601]                 mov    [lsterr], al    ; terminate list with error code
14792                                   ; 24/12/2023
14793 00000DB2 B90B00                   mov    cx, 11 ; PCDOS 7.1 ; 02/09/2023
14794                                   ;mov   cx, 9           ; numerr (= errout-errin)
14795                                                          ; number of possible error conditions
14796 00000DB5 BF[3C01]                 mov    di, errin       ; point to error conditions
14797 00000DB8 F2AE                     repne scasb
14798
14799                                   ; 24/12/2023
14800                                   ; 02/09/2023
14801 00000DBA 8A450A                   mov    al, [di+10] ; PCDOS 7.1 IBMBIO.COM
14802                                   ; 10/12/2022
14803                                   ;mov   al, [di+8]      ; [di+numerr-1]
14804                                                          ; get translation
14805                                   ; 19/10/2022 - Temporary !
14806                                   ;db    8Ah, 85h, 8, 0 ; mov al, [di+8]
14807 00000DBD 07                       pop    es
14808 00000DBE 59                       pop    cx
14809 00000DBF F9                       stc                    ; flag error condition
14810 00000DC0 C3                       retn
14811
14812                                ; =============== S U B   R O U T I N E =======================================
14813
14814                                ; set the time of last access for this drive.
14815                                ; this is done only for removable media. es:di -> bds
14816
14817                                set_tim:
14818 00000DC1 50                       push   ax
14819 00000DC2 E86CF7                   call   GetTickCnt      ; Does INT 1A ah=0 & updates daycnt
14820
14821                                ; we have the new time. if we see that the time has passed,
14822                                ; then we reset the threshold counter...
14823
14824                                   ; 24/12/2023 - Retro DOS v5.0
14825 00000DC5 263B5579                 cmp    dx, [es:di+79h]        ; PCDOS 7.1 IBMBIO.COM
14826                                   ;cmp   dx, [es:di+47h]        ; [es:di+BDS.tim_lo]
14827 00000DC9 7506                     jne    short setaccess
14828                                   ; 24/12/2023
14829 00000DCB 263B4D7B                 cmp    cx, [es:di+7Bh]        ; PCDOS 7.1 IBMBIO.COM
14830                                   ;cmp   cx, [es:di+49h]        ; [es:di+BDS.tim_hi]
14831                                   ;jz    short done_set
14832                                   ; 12/12/2022
14833 00000DCF 740E                     je     short done_set2
14834                                setaccess:
14835 00000DD1 C606[1D01]00             mov    byte [accesscount], 0
14836
14837                                   ; 24/12/2023 - Retro DOS v5.0 (PCDOS 7.1 IBMBIO.COM)
14838 00000DD6 26895579                 mov    [es:di+79h], dx
14839 00000DDA 26894D7B                 mov    [es:di+7Bh], cx
14840                                   ;mov   [es:di+47h], dx        ; [es:di+BDS.tim_lo]
14841                                   ;mov   [es:di+49h], cx        ; [es:di+BDS.tim_hi]
14842                                done_set:
14843 00000DDE F8                       clc
14844                                done_set2:         ; 12/12/2022
14845 00000DDF 58                       pop    ax
14846 00000DE0 C3                       retn
14847
14848                                ; =============== S U B   R O U T I N E =======================================
14849
14850                                ; this routine is called if an error occurs while formatting or verifying.
14851                                ; it resets the drive,and decrements the retry count.
14852                                ; on entry - ds:di - points to bds for the drive
14853                                ;        bp    - contains retry count
14854                                ; on exit    flags indicate result of decrementing retry count
14855
14856                                again:
14857 00000DE1 E85705                   call   ResetDisk
14858 00000DE4 80FC06                   cmp    ah, 6
14859 00000DE7 7402                     jz     short dont_dec_retry_count ; If      it is a media change error
14860                                                                       ; do not decrement retry count.
14861 00000DE9 4D                       dec    bp              ; decrement retry count
14862 00000DEA C3                       retn
14863                                ; ---------------------------------------------------------------------------
14864
14865                                dont_dec_retry_count:
14866 00000DEB 08E4                     or     ah, ah
14867 00000DED C3                       retn
14868
14869                                ;---------------------------------------------------------------------------
14870                                ; Retro DOS v5.0 - PCDOS 7.1 IBMBIO.COM - BIOSCODE:0E30h
14871                                ;---------------------------------------------------------------------------
14872                                ; 24/12/2023 - Retro DOS v5.0
14873                                ;;;;
14874
14875 00000DEE 00                  ioctl_drvnum:      db 0
14876
14877                                   ; 24/12/2023
14878
14879                                ; =============== S U B R O U T I N E =======================================
14880
14881                                get_phy_drv_num:
14882 00000DEF E8B2F7                   call   SetDrive        ; get physical drive number
14883                                                          ; INPUT: al = logical drive number (BDS.drivelet)
14884                                                          ; OUTPUT: physical drive number (BDS.drivenum)
14885 00000DF2 268A5504                 mov    dl, [es:di+4]  ; [es:di+BDS.drivenum]
14886 00000DF6 C3                       retn
14887
```

```
14888                                   ; ============== S U B R O U T I N E =======================================
14889
14890                                           ; 24/12/2023
14891                                   ioctl_output:
14892 00000DF7 E8F5FF                             call    get_phy_drv_num
14893 00000DFA 2E8816[EE0D]                       mov     [cs:ioctl_drvnum], dl
14894 00000DFF B441                               mov     ah, 41h
14895 00000E01 BBAA55                             mov     bx, 55AAh
14896 00000E04 CD13                               int     13h             ; DISK - Check for INT 13h Extensions
14897                                                                     ; BX = 55AAh, DL = drive number
14898                                                                     ; Return: CF set if not supported
14899                                                                     ; AH = extensions version
14900                                                                     ; BX = AA55h
14901                                                                     ; CX = Interface support bit map
14902 00000E06 7235                               jc      short int13h_exts_err
14903                                   ioctl_input_1:
14904 00000E08 C43E[1200]                         les     di, [ptrsav]
14905 00000E0C 26C47D0E                           les     di, [es:di+14] ; [es:di+IOCTL_REQ.MINORFUNCTION]
14906 00000E10 723E                               jc      short ioctl_input_2
14907 00000E12 B80046                             mov     ax, 4600h       ; Eject removable media
14908 00000E15 263805                             cmp     [es:di], al     ; al = 0 ; disk ioctl function = 0
14909 00000E18 7417                               je      short ioctl_output_1
14910 00000E1A 26803D01                           cmp     byte [es:di], 1 ; al = 1 ; disk ioctl function = 1
14911 00000E1E 751B                               jne     short ioctl_output_2
14912 00000E20 B80145                             mov     ax, 4501h       ; Lock/unlock media
14913                                                                     ; (al, 0 = lock, 1 = unlock)
14914 00000E23 26807D0100                         cmp     byte [es:di+1], 0 ; unlock (reverse of INT 13h ah=45h)
14915 00000E28 7407                               jz      short ioctl_output_1
14916 00000E2A 26384501                           cmp     [es:di+1], al   ; lock (reverse of INT 13h ah=45h)
14917 00000E2E 750B                               jne     short ioctl_output_2
14918 00000E30 48                                 dec     ax
14919                                   ioctl_output_1:
14920 00000E31 2E8A16[EE0D]                       mov     dl, [cs:ioctl_drvnum]
14921 00000E36 CD13                               int     13h             ; DISK - IBM/MS Extension - LOCK/UNLOCK DRIVE (DL - drive, [SI - disk address
packet)
14922 00000E38 7203                               jc      short int13h_exts_err
14923                                   ioctl_lock_err:
14924                                             ; cf=0
14925                                   ioctl_output_ret:
14926                                             ;clc
14927 00000E3A C3                                 retn
14928                                   ; ---------------------------------------------------------------------------
14929
14930                                   ioctl_output_2:
14931 00000E3B B401                               mov     ah, 1
14932                                   int13h_exts_err:
14933 00000E3D 80FCB0                             cmp     ah, 0B0h        ; volume not locked in drive
14934 00000E40 74F8                               je      short ioctl_lock_err
14935 00000E42 80FCB4                             cmp     ah, 0B4h        ; lock count exceeded
14936 00000E45 74F3                               je      short ioctl_lock_err
14937 00000E47 E9DAF7                             jmp     err_exitj
14938
14939                                   ; ============== S U B R O U T I N E =======================================
14940
14941                                           ; 24/12/2023
14942                                   ioctl_input:
14943 00000E4A E8A2FF                             call    get_phy_drv_num
14944 00000E4D F9                                 stc
14945 00000E4E EBB8                               jmp     short ioctl_input_1
14946                                   ioctl_input_2:
14947 00000E50 26803D06                           cmp     byte [es:di], 6         ; disk ioctl function = 6
14948 00000E54 75E5                               jne     short ioctl_output_2
14949 00000E56 B80245                             mov     ax, 4502h       ; get lock status
14950 00000E59 CD13                               int     13h             ; DISK - IBM/MS Extension - LOCK/UNLOCK DRIVE (DL - drive, [SI - disk address
packet)
14951 00000E5B 72E0                               jc      short int13h_exts_err
14952 00000E5D BB0C00                             mov     bx, 0Ch         ; bit 1 lock bit
14953 00000E60 3C00                               cmp     al, 0           ; not locked
14954 00000E62 7402                               jz      short ioctl_input_3
14955 00000E64 B30E                               mov     bl, 0Eh
14956                                   ioctl_input_3:
14957 00000E66 53                                 push    bx
14958 00000E67 B404                               mov     ah, 4
14959 00000E69 B90101                             mov     cx, 101h
14960 00000E6C B601                               mov     dh, 1
14961 00000E6E CD13                               int     13h             ; DISK - VERIFY SECTORS
14962                                                                     ; AL = number of sectors to verify, CH = track, CL = sector
14963                                                                     ; DH = head, DL = drive
14964                                                                     ; Return: CF set on error, AH = status
14965                                                                     ; AL = number of sectors verified
14966 00000E70 5B                                 pop     bx
14967 00000E71 80FC31                             cmp     ah, 31h         ; no media in drive (IBM/MS INT 13 extensions)
14968 00000E74 740B                               je      short ioctl_input_5
14969 00000E76 80FC80                             cmp     ah, 80h         ; timeout (not ready)
14970 00000E79 7406                               je      short ioctl_input_5
14971                                   ioctl_input_4:
14972 00000E7B 26895D01                           mov     [es:di+1], bx
14973 00000E7F EBB9                               jmp     short ioctl_lock_err
14974                                   ioctl_input_5:
14975 00000E81 81CB0108                           or      bx, 801h        ; bit 0 error bit (1 = error, 31h or 80h)
14976                                                                     ; bit 11 (not ready -removable media error- bit)
14977                                                                     ; if bit 11 = 0, another error (except 31h and 80h)
14978 00000E85 EBF4                               jmp     short ioctl_input_4
14979
14980                                   ; ---------------------------------------------------------------------------
14981                                   ;;;;
14982
14983                                   ; 16/10/2022 - Retro DOS 4.0 (Modified MSDOS 5.0 IO.SYS)
14984
14985                                   ;--------------------------------------------------------------------------
14986                                   ; MSDIOCTL.ASM - MSDOS 6.0 - 1991
14987                                   ;--------------------------------------------------------------------------
14988                                   ; 11/03/2019 - Retro DOS v4.0
14989
14990                                   ; 18/03/2019
14991
14992                                   ; ============================================================================
14993                                   ;
14994                                   ; NOTE: GetAccessFlag/SetAccessFlag is unpublished function.
14995                                   ;
14996                                   ;      This function is intended to give the user to control the
14997                                   ;      bds table flags of unformatted_media bit.
14998                                   ;      GetAccessFlag will show the status -
14999                                   ;    a_DiskAccess_Control.dac_access_flag = 0 disk i/o not allowed
15000                                   ;                                           1 disk i/o allowed
15001                                   ;      SetAccessFlag will set/reset the unformatted_media bit in flags -
15002                                   ;    a_DiskAccess_Control.dac_access_flag = 0 allow disk i/o
15003                                   ;                                           1 disallow disk i/o
15004                                   ; ============================================================================
15005
15006                                             ; generic ioctl dispatch tables
15007
15008                                   ; BIOSCODE:0C3Ch (MSDOS 6.21, IO.SYS)
15009
```

```
                                ; 24/12/2023
                                ; BIOSCODE:0ECAh (PCDOS 7.1, IBMBIO.COM)

                                ; ----------------------------------------------------------------------------
                                        ; 24/12/2023
                                        ;db 0
                                ; 09/12/2022
                                %if 0

                                IoReadJumpTable: db 8     ; ((IoWriteJumpTable-IoReadJumpTable)-1)/2
                                        dw 0CA7h        ; 60h   ; GetDeviceParameters
                                        dw 0EE8h        ; 61h   ; ReadTrack
                                        dw 0E86h        ; 62h   ; VerifyTrack
                                        dw 0CA3h                ; Cmd_Error_Proc
                                        dw 0CA3h                ; Cmd_Error_Proc
                                        dw 0CA3h                ; Cmd_Error_Proc
                                        dw 119Ah        ; 66h   ; GetMediaId
                                        dw 1269h        ; 67h   ; GetAccessFlag ; unpublished function
                                        dw 12C1h        ; 68h   ; SenseMediaType

                                IoWriteJumpTable: db 7    ; ((IOC_DC_Table-IoWriteJumpTable)-1)/2
                                        dw 0CF3h        ; 40h   ; SetDeviceParameters
                                        dw 0EEFh        ; 41h   ; WriteTrack
                                        dw 0DC1h        ; 42h   ; FormatTrack
                                        dw 0CA3h                ; Cmd_Error_Proc
                                        dw 0CA3h                ; Cmd_Error_Proc
                                        dw 0CA3h                ; Cmd_Error_Proc
                                        dw 11D2h        ; 46h   ; SetMediaId
                                        dw 1280h        ; 47h   ; SetAccessFlag ; unpublished function

                                %endif
                                        ; 24/12/2023 - Retro DOS v5.0
                                        ; (PCDOS 7.1 IBMBIO.COM - BIOSCODE:0ECAh)

                                        ; 09/12/2022
                                IoReadJumpTable:
00000E87 10                             db ((IoWriteJumpTable-IoReadJumpTable)-1)/2 ; 15
00000E88 [1A0F]                         dw GetDeviceParameters; 60h
00000E8A [9311]                         dw ReadTrack          ; 61h
00000E8C [3311]                         dw VerifyTrack        ; 62h
00000E8E [160F]                         dw Cmd_Error_Proc
00000E90 [160F]                         dw Cmd_Error_Proc
00000E92 [160F]                         dw Cmd_Error_Proc
00000E94 [1214]                         dw GetMediaId         ; 66h
00000E96 [EF14]                         dw GetAccessFlag      ; 67h ; unpublished function
00000E98 [4415]                         dw SenseMediaType     ; 68h
                                        ; 24/12/2023
                                        ; Retro DOS v5.0 (PCDOS 7.1 IBMBIO.COM)
00000E9A [160F]                         dw Cmd_Error_Proc     ; 69h
00000E9C [160F]                         dw Cmd_Error_Proc     ; 6Ah
00000E9E [160F]                         dw Cmd_Error_Proc
00000EA0 [160F]                         dw Cmd_Error_Proc
00000EA2 [160F]                         dw Cmd_Error_Proc
00000EA4 [160F]                         dw Cmd_Error_Proc     ; 6Eh
00000EA6 [C815]                         dw GetDrvMapInfo      ; 6Fh

                                IoWriteJumpTable:
00000EA8 0A                             db ((IOC_DC_Table-IoWriteJumpTable)-1)/2 ; 9
00000EA9 [7A0F]                         dw SetDeviceParameters; 40h
00000EAB [9A11]                         dw WriteTrack         ; 41h
00000EAD [6D10]                         dw FormatTrack        ; 42h
00000EAF [160F]                         dw Cmd_Error_Proc
00000EB1 [160F]                         dw Cmd_Error_Proc
00000EB3 [160F]                         dw Cmd_Error_Proc
00000EB5 [5214]                         dw SetMediaId         ; 46h
00000EB7 [0415]                         dw SetAccessFlag      ; 47h ; unpublished function
                                        ; 24/12/2023
                                        ; Retro DOS v5.0 (PCDOS 7.1 IBMBIO.COM)
00000EB9 [8115]                         dw SetLockState             ; 48h
00000EBB [9815]                         dw EjectMedia         ; 49h

                                ; ============================================================================
                                ; IOC_DC_Table
                                ;
                                ; This table contains all of the valid generic IOCtl Minor codes for
                                ; major function 08 to be used by the Ioctl_Support_Query function.
                                ; Added for 5.00
                                ; ============================================================================

                                        ; 24/12/2023 - Retro DOS v5.0
                                        ; (PCDOS 7.1 IBMBIO.COM - BIOSCODE:0F00h)

00000EBD 60                     IOC_DC_Table:      db 60h               ; GET_DEVICE_PARAMETERS
00000EBE 40                             db 40h                  ; SET_DEVICE_PARAMETERS
00000EBF 61                             db 61h                  ; READ_TRACK
00000EC0 41                             db 41h                  ; WRITE_TRACK
00000EC1 62                             db 62h                  ; VERIFY_TRACK
00000EC2 42                             db 42h                  ; FORMAT_TRACK
00000EC3 66                             db 66h                  ; GET_MEDIA_ID
00000EC4 46                             db 46h                  ; SET_MEDIA_ID
00000EC5 67                             db 67h                  ; GET_ACCESS_FLAG
00000EC6 47                             db 47h                  ; SET_ACCESS_FLAG
00000EC7 68                             db 68h                  ; SENSE_MEDIA_TYPE
                                        ; 24/12/2023
                                        ; Retro DOS v5.0 (PCDOS 7.1 IBMBIO.COM)
00000EC8 48                             db 48h                  ; SET_LOCK_STATE
00000EC9 49                             db 49h                  ; EJECT_MEDIA
00000ECA 6F                             db 6Fh                  ; GET_DRV_MAP_INFO

                                ;IOC_DC_TABLE_LEN EQU $ - IOC_DC_Table

                                        ; 24/12/2023 - Retro DOS v5.0
                                        ; (PCDOS 7.1 IBMBIO.COM - BIOSCODE:0F0Eh)

00000ECB 00                     new_genioctl:      db 0

                                ; ----------------------------------------------------------------------------

                                ; 16/10/2022

                                ; ============================================================================
                                ; Do_Generic_IOCtl: perform generic ioctl request
                                ;
                                ;    input: AL contains logical drive
                                ;
                                ;   functions are dispatched through a call. On return, carry indicates
                                ;   error code in al. Note::bES:b& ds undefined on return from
                                ;   subfunctions.
                                ;
                                ; ============================================================================

                                ; 11/03/2019
                                        ; 24/12/2023 - Retro DOS v5.0
                                        ; (PCDOS 7.1 IBMBIO.COM - BIOSCODE:0F0Fh)
```

```
15134
15135                                           ; 19/10/2022
15136                                   do_generic_ioctl:               ; 2C7h:0C6Bh = 70h:31DBh
15137 00000ECC E8D5F6                             call    SetDrive      ; ES:DI Points to bds for drive
15138
15139                                           ; 24/12/2023
15140                                           ;;;
15141 00000ECF 2EC606[CB0E]00                     mov     byte [cs:new_genioctl], 0
15142                                                                   ; 0, old generic ioctl function
15143 00000ED5 06                                 push    es
15144 00000ED6 C41E[1200]                         les     bx, [ptrsav]  ; ES:BX Points to request header
15145 00000EDA 26807F0D08                         cmp     byte [es:bx+0Dh], 8 ; [es:bx+IOCTL_REQ.MAJORFUNCTION]
15146                                                                   ; RAWIO
15147                                           ; 24/12/2023
15148                                           ;mov    al, [es:bx+0Eh]       ; [es:bx+IOCTL_REQ.MINORFUNCTION]
15149                                           ;pop    es
15150                                           ;jnz    short IoctlFuncErr
15151 00000EDF 740A                             jz      short chk_genioctl_minor
15152 00000EE1 2EFE06[CB0E]                     inc     byte [cs:new_genioctl]
15153                                                                   ; 1, new generic ioctl function (FAT32)
15154 00000EE6 26807F0D48                       cmp     byte [es:bx+0Dh], 48h ; Generic IOCtl Request support
15155                                                                   ; (called only if bit 6 of attribute is set to 1)
15156                                   chk_genioctl_minor:
15157 00000EEB 268A470E                         mov     al, [es:bx+0Eh]       ; [es:bx+IOCTL_REQ.MINORFUNCTION]
15158 00000EEF 07                               pop     es
15159 00000EF0 7525                             jnz     short IoctlFuncErr
15160                                           ;;;
15161
15162                                           ; cas note: Could do the above two blocks in reverse order.
15163                                           ; Would have to preserve al for SetDrive
15164
15165                                           ; 10/12/2022
15166 00000EF2 BE[870E]                         mov     si, IoReadJumpTable
15167                                           ;mov    si, 0C3Ch     ; IoReadJumpTable
15168                                                                   ; at 2C7h:0C3Ch       = 70h:31ACh
15169 00000EF5 A820                             test    al, 20h       ; GEN_IOCTL_FN_TST ; test of req. function
15170 00000EF7 7503                             jnz     short NotGenericWrite ; function is a read.
15171                                           ; 10/12/2022
15172 00000EF9 BE[A80E]                         mov     si, IoWriteJumpTable
15173                                           ;mov    si, 0C4Fh     ; IoWriteJumpTable
15174                                                                   ; at 2C7h:0C4Fh       = 70h:31BFh
15175                                   NotGenericWrite:
15176 00000EFC 24DF                             and     al, 0DFh      ; ~GEN_IOCTL_FN_TST ; get rid of read/write bit
15177 00000EFE 2C40                             sub     al, 40h       ; offset for base function
15178 00000F00 2E3A04                           cmp     al, [cs:si]
15179 00000F03 7712                             ja      short IoctlFuncErr
15180 00000F05 98                               cbw
15181                                           ; 24/12/2023
15182                                           ;shl    ax, 1
15183 00000F06 01C0                             add     ax, ax
15184 00000F08 46                               inc     si
15185 00000F09 01C6                             add     si, ax
15186 00000F0B 2EFF14                           call    near [cs:si]
15187                                           ;call   word ptr cs:[si]
15188 00000F0E 2E8E1E[3000]                     mov     ds, [cs:BIOSDATAWORD]
15189                                           ;mov    ds, word ptr cs:BIOSDATAWORD ; [cs:Bios_Data_Word]
15190                                                                   ; 2C7h:30h = 70h:25A0h
15191 00000F13 B481                             mov     ah, 81h       ; Return this status in       case of carry
15192 00000F15 C3                               retn                  ; Pass carry flag through to exit code
15193                                   ; ---------------------------------------------------------------------------
15194
15195                                           ; Cmd_Error_Proc is called as a procedure and also use
15196                                           ; as a fall through from above
15197                                   Cmd_Error_Proc:                         ; 2C7h:0CA3h = 70h:3213h
15198 00000F16 5A                               pop     dx
15199                                   IoctlFuncErr:
15200 00000F17 E9BBF1                           jmp     bc_cmderr
15201                                   ; ---------------------------------------------------------------------------
15202
15203                                   ; 16/10/2022
15204
15205                                   ; ============================================================================
15206                                   ;** GetDeviceParameters:
15207                                   ;
15208                                   ;   GetDeviceParameters implements the generic ioctl function:
15209                                   ;   majorcode=RAWIO, minorcode=GetDeviceParameters (60h)
15210                                   ;
15211                                   ;   ENTRY   (ES:di) = BDS for drive
15212                                   ;           PtrSav = long pointer to request header
15213                                   ;   EXIT    ??? BUGBUG
15214                                   ;   USES    ??? BUGBUG
15215                                   ; ============================================================================
15216
15217                                           ; 24/12/2023 - Retro DOS v5.0
15218                                           ; (PCDOS 7.1 IBMBIO.COM - BIOSCODE:0F5Dh)
15219
15220                                           ; 19/10/2022
15221                                   GetDeviceParameters:
15222                                           ; Copy info from bds to the device parameters packet
15223
15224 00000F1A C51E[1200]                       lds     bx, [ptrsav]  ; DS:BX points to request header
15225 00000F1E C55F13                           lds     bx, [bx+19]   ; [bx+IOCTL_REQ.GENERICIOCTL_PACKET]
15226                                                                   ; (DS:BX) = return buffer
15227                                           ; 24/12/2023
15228 00000F21 268A453E                         mov     al, [es:di+3Eh]
15229                                           ;mov    al, [es:di+34] ; [es:di+BDS.formfactor]
15230 00000F25 884701                           mov     [bx+1], al    ; [bx+A_DEVICEPARAMETERS.DP_DEVICETYPE]
15231                                           ; 24/12/2023
15232 00000F28 268B453F                         mov     ax, [es:di+3Fh]
15233                                           ;mov    ax, [es:di+35] ; [es:di+BDS.flags]
15234 00000F2C 83E003                           and     ax, 3         ; fnon_removable+fchangeline
15235                                                                 ; Mask off other bits
15236 00000F2F 894702                           mov     [bx+2], ax    ; [bx+A_DEVICEPARAMETERS.DP_DEVICEATTRIBUTES]
15237                                           ; 24/12/2023
15238 00000F32 268B4541                         mov     ax, [es:di+41h]
15239                                           ;mov    ax, [es:di+37] ; [es:di+BDS.cylinders]
15240 00000F36 894704                           mov     [bx+4], ax    ; [bx+A_DEVICEPARAMETERS.DP_CYLINDERS]
15241 00000F39 30C0                             xor     al, al        ; Set media type to default
15242 00000F3B 884706                           mov     [bx+6], al    ; [bx+A_DEVICEPARAMETERS.DP_MEDIATYPE]
15243
15244                                           ; copy recommended bpb
15245
15246                                           ; 24/12/2023
15247 00000F3E 8D7543                           lea     si, [di+43h]
15248                                           ;lea    si, [di+39]   ; [di+BDS.rbytespersec]       = [di+BDS.R_BPB]
15249 00000F41 F60701                           test    byte [bx], 1  ; [bx+A_DEVICEPARAMETERS.DP_SPECIALFUNCTIONS]
15250                                                                 ; BUILD_DEVICE_BPB
15251 00000F44 7412                             jz      short UseBpbPresent
15252 00000F46 1E                               push    ds            ; Save request packet segment
15253 00000F47 2E8E1E[3000]                     mov     ds, [cs:BIOSDATAWORD]
15254                                           ;mov    ds, word ptr cs:BIOSDATAWORD ; [cs:Bios_Data_Word]
15255                                                                 ; 2C7h:30h = 70h:25A0h
15256                                                                 ; Point back to Bios_Data
15257 00000F4C E828FA                           call    checksingle
```

```
15258 00000F4F E884F7                           call    GetBp           ; Build the bpb from scratch
15259 00000F52 1F                               pop     ds              ; Restore request packet segment
15260 00000F53 7224                             jb      short GetParmRet
15261 00000F55 8D7506                           lea     si, [di+6]      ; [di+BDS.bytespersec] = [di+BSD.DP_BPB]
15262                                                                   ; Use this subfield of bds instead
15263                                   UseBpbPresent:
15264 00000F58 8D7F07                           lea     di, [bx+7]      ; [bx+A_DEVICEPARAMETERS.DP_BPB]
15265                                                                   ; This is where       the result goes
15266                                           ; 24/12/2023
15267 00000F5B 31D2                             xor     dx, dx ; 0
15268
15269                                           ; 24/12/2023
15270 00000F5D B91F00                           mov     cx, 31          ; A_BPB.size = 31
15271                                           ;mov    cx, 25          ; A_BPB.size - 6
15272                                                                   ; For now use 'small' bpb
15273                                           ; 24/12/2023
15274                                           ;;;
15275 00000F60 2E3816[CB0E]                     cmp     [cs:new_genioctl], dl ; 0 ? ; *
15276 00000F65 7404                             jz      short gdp_1     ; old type (FAT12 & FAT16) structure
15277                                           ;mov    cx, 53          ; FAT32 BPB size
15278                                           ;mov    dx, 32          ; 53+32 = 85 bytes (A_BPB_FAT32.size)
15279 00000F67 B135                             mov     cl, 53
15280 00000F69 B220                             mov     dl, 32
15281                                   gdp_1:
15282                                           ;;;
15283 00000F6B 1E                               push    ds              ; reverse segments for copy
15284 00000F6C 06                               push    es
15285 00000F6D 1F                               pop     ds
15286 00000F6E 07                               pop     es
15287 00000F6F F3A4                             rep movsb
15288
15289                                           ; 24/12/2023
15290                                           ;;;
15291 00000F71 89D1                             mov     cx, dx          ; 0 or 32
15292 00000F73 E304                             jcxz    gdp_2
15293 00000F75 30C0                             xor     al, al          ; 32 zeros
15294 00000F77 F3AA                             rep stosb
15295                                   gdp_2:
15296                                           ;clc    ; cf is already 0 ; * ; 24/12/2023
15297                                           ;;;
15298
15299                                           ; 12/12/2022
15300                                           ; cf=0 (cmp instruction -above- resets cf)
15301                                           ;clc
15302                                   GetParmRet:
15303 00000F79 C3                               retn
15304                                   ; ---------------------------------------------------------------------------
15305
15306                                   ; 17/10/2022
15307                                   ; 16/10/2022
15308
15309                                   ; ============================================================================
15310                                   ; SetDeviceParameters:
15311                                   ;
15312                                   ; input: ES:di points to bds for drive
15313                                   ; ============================================================================
15314
15315                                           ; 24/12/2023 - Retro DOS v5.0
15316                                           ; (PCDOS 7.1 IBMBIO.COM - BIOSCODE:0FC0h)
15317
15318                                           ; 19/10/2022
15319                                   SetDeviceParameters:                    ; 2C7h:0CF3h = 70h:3263h
15320 00000F7A C51E[1200]                       lds     bx, [ptrsav]    ; DS:BX points to request header
15321 00000F7E C55F13                           lds     bx, [bx+19]     ; [bx+IOCTL_REQ.GENERICIOCTL_PACKET]
15322                                           ; 24/12/2023
15323 00000F81 26814D3F4001                     or      word [es:di+3Fh], 140h
15324                                           ;or     word [es:di+23h], 140h ; [es:di+BDS.flags]
15325                                                                   ; fchanged_by_format|fchanged
15326 00000F87 F60702                           test    byte [bx], 2    ; [bx+A_DEVICEPARAMETERS.DP_SPECIALFUNCTIONS]
15327                                                                   ; ONLY_SET_TRACKLAYOUT
15328                                           ;jnz    short setTrackTable
15329                                           ; 24/12/2023
15330 00000F8A 7403                             jz      short sdp_1
15331 00000F8C E98000                           jmp     setTrackTable
15332                                   sdp_1:
15333 00000F8F 8A4701                           mov     al, [bx+1]      ; [bx+A_DEVICEPARAMETERS.DP_DEVICETYPE]
15334                                           ; 24/12/2023
15335 00000F92 2688453E                         mov     [es:di+3Eh], al
15336                                           ;mov    [es:di+34], al ; [es:di+BDS.formfactor]
15337 00000F96 8B4704                           mov     ax, [bx+4]      ; [bx+A_DEVICEPARAMETERS.DP_CYLINDERS]
15338                                           ; 24/12/2023
15339 00000F99 26894541                         mov     [es:di+41h], ax
15340                                           ;mov    [es:di+37], ax ; [es:di+BDS.cylinders]
15341 00000F9D 8B4702                           mov     ax, [bx+2]      ; [bx+A_DEVICEPARAMETERS.DP_DEVICEATTRIBUTES]
15342 00000FA0 1E                               push    ds
15343                                           ; 17/10/2022
15344 00000FA1 2E8E1E[3000]                     mov     ds, [cs:BIOSDATAWORD]
15345                                           ;mov    ds, word ptr cs:BIOSDATAWORD ; [cs:Bios_Data_Word]
15346                                                                   ; 2C7h:30h = 70h:25A0h
15347                                           ;cmp    byte [fhave96], 0
15348 00000FA6 803E[7700]00                     cmp     byte [fhave96], 0
15349 00000FAB 1F                               pop     ds
15350 00000FAC 7502                             jnz     short HaveChange ; we have changeline support
15351                                           ; 10/12/2022
15352 00000FAE 24FD                             and     al, 0FDh
15353                                           ;and    ax, 0FFFDh      ; ~fchangeline
15354
15355                                           ; Ignore all bits except non_removable and changeline
15356                                   HaveChange:
15357 00000FB0 83E003                           and     ax, 3           ; fnon_removable|fchangeline
15358                                           ; 24/12/2023
15359 00000FB3 268B4D3F                         mov     cx, [es:di+3Fh]
15360                                           ;mov    cx, [es:di+35] ; [es:di+BDS.flags]
15361 00000FB7 81E1F4FD                         and     cx, 0FDF4h      ; ~(fnon_removable|fchangeline|good_tracklayout|unformatted_media)
15362 00000FBB 09C8                             or      ax, cx
15363                                           ; 24/12/2023
15364 00000FBD 2689453F                         mov     [es:di+3Fh], ax
15365                                           ;mov    [es:di+35], ax ; [es:di+BDS.flags]
15366 00000FC1 8A4706                           mov     al, [bx+6]      ; [bx+A_DEVICEPARAMETERS.DP_MEDIATYPE]
15367                                                                   ; Set media type
15368 00000FC4 1E                               push    ds
15369 00000FC5 2E8E1E[3000]                     mov     ds, [cs:BIOSDATAWORD]
15370                                           ;mov    ds, word ptr cs:BIOSDATAWORD ; [cs:Bios_Data_Word]
15371 00000FCA A2[A805]                         mov     [mediatype], al
15372                                           ;mov    ds:mediatype, al
15373
15374                                           ; 24/12/2023
15375                                           ;;;
15376 00000FCD B93500                           mov     cx, 53          ; FAT32 BPB size
15377 00000FD0 2E803E[CB0E]00                   cmp     byte [cs:new_genioctl], 0
15378 00000FD6 7502                             jnz     short sdp_2     ; new type (FAT32) structure
15379                                           ;mov    cx, 31          ; A_BPB.size = 31
15380 00000FD8 B11F                             mov     cl, 31
15381                                   sdp_2:
```

```
                                        ;;;
00000FDA 1F                             pop     ds

                                        ; The media changed (maybe) so we will have to do a set dasd
                                        ; the next time we format a track

                                        ; 24/12/2023
00000FDB 26804D3F80                     or      byte [es:di+3Fh], 80h
                                        ; 10/12/2022
                                        ;or     byte [es:di+35], 80h
                                        ;;or    word [es:di+35], 80h ; [es:di+BDS.flags]
                                                                     ; set_dasd_true
00000FE0 57                             push    di              ; Save bds pointer

                                        ; Figure out what we are supposed to do with the bpb
                                        ; were we asked to install a fake bpb?

00000FE1 F60701                         test    byte [bx], 1    ; [bx+A_DEVICEPARAMETERS.DP_SPECIALFUNCTIONS]
                                                                ; INSTALL_FAKE_BPB
00000FE4 7511                           jnz     short InstallFakeBpb

                                        ; were we returning a fake bpb when asked to build a bpb?

                                        ; 24/12/2023
00000FE6 26F6453F04                     test    byte [es:di+3Fh], 4
                                        ; 10/12/2022
                                        ;test   byte [es:di+35], 4
                                        ;;test  word [es:di+35], 4 ; [es:di+BDS.flags]
                                                                   ; return_fake_bpb
00000FEB 7405                           jz      short InstallRecommendedBpb

                                        ; we were returning a fake bpb but we can stop now

                                        ; 24/12/2023
00000FED 2680653FFB                     and     byte [es:di+3Fh], 0FBh
                                        ; 10/12/2022
                                        ;and    byte [es:di+35], 0FBh
                                        ;;and   word [es:di+35], 0FFFBh ; [es:di+BDS.flags]
                                                                        ; ~return_fake_bpb
                                InstallRecommendedBpb:
                                        ; 24/12/2023
                                        ;mov    cx, 31          ; A_BPB.size
                                        ;lea    di, [di+27h]    ; [di+BDS.R_BPB] = [di+BDS.rbytespersec]
                                        ; cx = 53 or 31
00000FF2 8D7D43                         lea     di, [di+43h]    ; (PCDOS 7.1 IBMBIO.COM)
00000FF5 EB08                           jmp     short CopyTheBpb
                                ; ---------------------------------------------------------------------------

                                InstallFakeBpb:
                                        ; 24/12/2023
00000FF7 26804D3F04                     or      byte [es:di+3Fh], 4
                                        ; 10/12/2022
                                        ;or     byte [es:di+35], 4
                                        ;;or    word [es:di+35], 4 ; byte [es:di+BDS.flags]
                                                                   ; return_fake_bpb
                                        ; 24/12/2023
                                        ; cx = 53 or 31
                                        ;mov    cx, 25          ; A_BPB.size - 6
                                                                ; move 'smaller' bpb
00000FFC 8D7D06                         lea     di, [di+6]      ; [es:di+BDS.BPB] = [es:di+BDS.bytespersec]
                                CopyTheBpb:
00000FFF 8D7707                         lea     si, [bx+7]      ; [bx+A_DEVICEPARAMETERS.DP_BPB]
00001002 F3A4                           rep movsb
00001004 1E                             push    ds              ; Save packet segment
                                        ; 17/10/2022
00001005 2E8E1E[3000]                   mov     ds, [cs:BIOSDATAWORD]
                                        ;mov    ds, word ptr cs:BIOSDATAWORD ; [cs:Bios_Data_Word]
                                                                ; Setup for ds -> Bios_Data
0000100A E8DD03                         call    RestoreOldDpt   ; Restore the old Dpt from TempDpt
0000100D 1F                             pop     ds              ; Restore packet segment
0000100E 5F                             pop     di              ; Restore bds pointer
                                setTrackTable:
                                        ; 24/12/2023
                                        ;mov    cx, [bx+38]     ; [bx+26h]
                                        ;;;
0000100F 8B4F5C                         mov     cx, [bx+5Ch]    ; [bx+A_DEVICEPARAMETERS.DP_TRACKTABLEENTRIES]
                                                                ; offset 85+7 (A_BPB.size+7) (FAT32)
00001012 2E803E[CB0E]00                 cmp     byte [cs:new_genioctl], 0
00001018 7503                           jnz     short sdp_3     ; new type (FAT32) structure
0000101A 8B4F26                         mov     cx, [bx+26h]    ; [bx+A_DEVICEPARAMETERS.DP_TRACKTABLEENTRIES]
                                                                ; offset 31+7 (A_BPB.size+7)
                                sdp_3:
                                        ;;;

0000101D 1E                             push    ds
0000101E 2E8E1E[3000]                   mov     ds, [cs:BIOSDATAWORD]
00001023 890E[AA04]                     mov     [sectorspertrack], cx
00001027 1F                             pop     ds

                                        ; 24/12/2023
00001028 2680653FF7                     and     byte [es:di+3Fh], 0F7h
                                        ; 10/12/2022
                                        ;and    byte [es:di+35], 0F7h
                                        ;;and   word [es:di+35], 0FFF7h ; [es:di+BDS.flags]
                                                                        ; ~good_tracklayout
0000102D F60704                         test    byte [bx], 4    ; [bx+A_DEVICEPARAMETERS.DP_SPECIALFUNCTIONS]
                                                                ; TRACKLAYOUT_IS_GOOD
00001030 7405                           jz      short UglyTrackLayOut
                                        ; 24/12/2023
00001032 26804D3F08                     or      byte [es:di+3Fh], 8
                                        ; 10/12/2022
                                        ;or     byte [es:di+35], 8
                                        ;;or    word [es:di+35], 8 ; [es:di+BDS.flags]
                                                                   ; good_tracklayout
                                UglyTrackLayOut:
00001037 83F93F                         cmp     cx, 63          ; MAX_SECTORS_IN_TRACK
0000103A 772D                           ja      short TooManyPerTrack
                                        ;jcxz   short SectorInfoSaved
0000103C E329                           jcxz    SectorInfoSaved       ; 19/10/2022

0000103E BF[AC04]                       mov     di, tracktable

                                        ; 24/12/2023
                                        ;lea    si, [bx+40]     ; [bx+A_DEVICEPARAMETERS.DP_SECTORTABLE]
                                        ;;;
00001041 8D775E                         lea     si, [bx+5Eh]    ; [bx+A_DEVICEPARAMETERS.DP_SECTORTABLE]
                                                                ; offset 85+9 (A_BPB.size+9) (FAT32)
00001044 2E803E[CB0E]00                 cmp     byte [cs:new_genioctl], 0
0000104A 7503                           jnz     short sdp_4     ; new type (FAT32) structure
0000104C 8D7728                         lea     si, [bx+28h]    ; [bx+A_DEVICEPARAMETERS.DP_SECTORTABLE]
                                                                ; offset 31+9 (A_BPB.size+9)
                                sdp_4:
                                        ;;;

```

```
                                        ; 17/10/2022
0000104F 2E8E06[3000]                   mov     es, [cs:BIOSDATAWORD]
                                        ;mov    es, word ptr cs:BIOSDATAWORD ; [cs:Bios_Data_Word]
                                                ; Trash our bds pointer
                                StoreSectorInfo:
00001054 47                             inc     di
00001055 47                             inc     di              ; Skip over cylinder and head
00001056 AD                             lodsw                   ; Get sector id
00001057 AA                             stosb                   ; Copy it
00001058 AD                             lodsw                   ; Get sector size

                                        ; 24/12/2023
                                        ; 02/09/2023 (PCDOS 7.1)
                                        ;call   SectSizeToSectIndex
                                        ; 18/04/2024
                                        ;cmp    ah, 3 ; 02/09/2023
00001059 80FC02                         cmp     ah, 2           ; (0=>128,1=>256,2=>512,3=>1024)
                                                                ; examine upper       byte only
0000105C 7704                           ja      short OneK
0000105E 88E0                           mov     al, ah          ; value in AH is the index!
00001060 EB02                           jmp     short sdp_s
                                OneK:
00001062 B003                           mov     al, 3           ; 1024 bytes per sector
                                sdp_s:
00001064 AA                             stosb                   ; Store sector SIZE index
00001065 E2ED                           loop    StoreSectorInfo
                                SectorInfoSaved:
00001067 F8                             clc
00001068 C3                             retn
                                ; ---------------------------------------------------------------------------

                                TooManyPerTrack:
00001069 B00C                           mov     al, 0Ch
0000106B F9                             stc
0000106C C3                             retn
                                ; ---------------------------------------------------------------------------

                                ; 16/10/2022
                                ; ===========================================================================
                                ; FormatTrack:
                                ; if specialfunction byte is 1,then this is a status call to see if there is
                                ; rom support for the combination of sec/trk and # of cyln,and if the
                                ; combination is legal. if specialfunction byte is 0,then format the track.
                                ;
                                ; input: ES:di points to bds for drive
                                ;
                                ; output:
                                ;   for status call:
                                ;   specialfunction byte set to:
                                ;           0 - rom support + legal combination
                                ;           1 - no rom support
                                ;           2 - illegal combination
                                ;           3 - no media present
                                ;   carry cleared.
                                ;
                                ;   for format track:
                                ;           carry set if error
                                ;
                                ; ===========================================================================

                                ; 16/03/2019
                                        ; 24/12/2023 - Retro DOS 5.0
                                        ; (PCDOS 7.1 IBMBIO.COM - BIOSCODE:10B7h)

                                        ; 19/10/2022
                                FormatTrack:
0000106D C51E[1200]                     lds     bx, [ptrsav]
00001071 C55F13                         lds     bx, [bx+19]     ; [bx+IOCTL_REQ.GENERICIOCTL_PACKET
00001074 F60701                         test    byte [bx], 1    ; [bx+A_DEVICEPARAMETERS.DP_SPECIALFUNCTIONS]
                                                                ; STATUS_FOR_FORMAT
00001077 740E                           jz      short DoFormatTrack
00001079 1E                             push    ds
                                        ; 17/10/2022
0000107A 2E8E1E[3000]                   mov     ds, [cs:BIOSDATAWORD]
                                        ;mov    ds, word ptr cs:BIOSDATAWORD ; [cs:Bios_Data_Word]
0000107F E82502                         call    SetMediaForFormat ; Also moves current Dpt to TempDpt
00001082 1F                             pop     ds
00001083 8807                           mov     [bx], al        ; [bx+A_DEVICEPARAMETERS.DP_SPECIALFUNCTIONS]
00001085 F8                             clc
00001086 C3                             retn
                                ; ---------------------------------------------------------------------------

                                DoFormatTrack:
                                        ; 24/12/2023 - Retro DOS 5.0
00001087 26807D3E05                     cmp     byte [es:di+3Eh], 5
                                        ;cmp    byte [es:di+34], 5 ; [es:di+BDS.formfactor]
                                                                ; DEV_HARDDISK
0000108C 7508                           jnz     short DoFormatDiskette
                                        ; 17/10/2022
0000108E 2E8E1E[3000]                   mov     ds, [cs:BIOSDATAWORD]
                                        ;mov    ds, word ptr cs:BIOSDATAWORD ; [cs:Bios_Data_Word]
                                                ; Point to Bios_Data (at 2C7h:30h or 70h:25A0h)
00001093 E99D00                         jmp     VerifyTrack
                                ; ---------------------------------------------------------------------------

                                DoFormatDiskette:
00001096 8B4F01                         mov     cx, [bx+1]
00001099 8B5703                         mov     dx, [bx+3]
0000109C F60702                         test    byte [bx], 2
                                        ; 17/10/2022
0000109F 2E8E1E[3000]                   mov     ds, [cs:BIOSDATAWORD]
                                        ;mov    ds, word ptr cs:BIOSDATAWORD ; [cs:Bios_Data_Word]
                                                ; Setup ds-> Bios_Data for verify
000010A4 7403                           jz      short DoFormatDiskette_1
000010A6 E9E500                         jmp     VerifyTrack_Err
                                ; ---------------------------------------------------------------------------

                                DoFormatDiskette_1:
000010A9 E8FB01                         call    SetMediaForFormat ; Also moves current Dpt to TempDpt
000010AC 3C01                           cmp     al, 1           ;  ROM support for sec/trk,# trks comb?
000010AE 7406                           jz      short NeedToSetDasd ; Old rom
000010B0 3C03                           cmp     al, 3           ; Time out error?
000010B2 7507                           jnz     short NoSetDasd         ; No,fine. (at this point, don't care
                                                                ; about the illegal combination)
000010B4 EB68                           jmp     short FormatFailed
                                ; ---------------------------------------------------------------------------

                                NeedToSetDasd:
000010B6 52                             push    dx
000010B7 E89001                         call    SetDasd         ; INT 13h, AH=17h
000010BA 5A                             pop     dx
                                NoSetDasd:
000010BB E8B9F8                         call    checksingle     ; Do any needed       diskette swapping
```

```
15630 000010BE 89D0                             mov     ax, dx          ; Get track from packet
15631 000010C0 A3[3901]                         mov     [trknum], ax
15632 000010C3 880E[3801]                       mov     [hdnum], cl     ; Store head from packet
15633 000010C7 88CC                             mov     ah, cl
15634 000010C9 BB[AC04]                         mov     bx, tracktable
15635 000010CC 8B0E[AA04]                       mov     cx, [sectorspertrack]
15636                                           ; 24/12/2023 - Retro DOS 5.0
15637 000010D0 E307                             jcxz    set_fmt_retry_count
15638                                   StoreCylinderHead:
15639 000010D2 8907                             mov     [bx], ax        ; Store into TrackTable
15640 000010D4 83C304                           add     bx, 4           ; Skip to next sector field
15641 000010D7 E2F9                             loop    StoreCylinderHead
15642                                   set_fmt_retry_count:      ; 24/12/2023
15643                                           ;mov    cx, 5           ; MAXERR - Set up retry       count
15644                                           ; 02/09/2023
15645 000010D9 B105                             mov     cl, 5
15646                                   FormatRetry:
15647 000010DB 51                               push    cx
15648 000010DC BB[AC04]                         mov     bx, tracktable
15649 000010DF A0[AA04]                         mov     al, [sectorspertrack]
15650 000010E2 B405                             mov     ah, 5           ; romformat
15651 000010E4 8C1E[A804]                       mov     [xfer_seg], ds
15652 000010E8 E86602                           call    ToRom
15653 000010EB 59                               pop     cx
15654 000010EC 7216                             jb      short FormatError
15655 000010EE 51                               push    cx              ; Now verify the sectors just formatted.
15656                                                                   ; NOTE: because of bug in some BIOSes we have to
15657                                                                   ;       set ES:BX to 00:00
15658 000010EF 53                               push    bx
15659 000010F0 31DB                             xor     bx, bx
15660 000010F2 891E[A804]                       mov     [xfer_seg], bx
15661 000010F6 A0[AA04]                         mov     al, [sectorspertrack]
15662 000010F9 B404                             mov     ah, 4           ; romverify
15663 000010FB B101                             mov     cl, 1
15664 000010FD E85102                           call    ToRom
15665 00001100 5B                               pop     bx
15666 00001101 59                               pop     cx
15667 00001102 7329                             jnb     short FormatOk
15668                                   FormatError:
15669 00001104 E83402                           call    ResetDisk
15670 00001107 C606[AA05]01                     mov     byte [had_format_error], 1
15671 0000110C 50                               push    ax
15672 0000110D 51                               push    cx
15673 0000110E 52                               push    dx
15674 0000110F E89501                           call    SetMediaForFormat
15675 00001112 3C01                             cmp     al, 1
15676 00001114 7503                             jnz     short WhileErr
15677 00001116 E83101                           call    SetDasd
15678                                   WhileErr:
15679 00001119 5A                               pop     dx
15680 0000111A 59                               pop     cx
15681 0000111B 58                               pop     ax
15682 0000111C E2BD                             loop    FormatRetry
15683                                   FormatFailed:
15684 0000111E C606[AA05]01                     mov     byte [had_format_error], 1
15685                                                                   ; Set the format error flag
15686 00001123 80FC06                           cmp     ah, 6           ; DSK_CHANGELINE_ERR - convert change line
15687 00001126 7502                             jnz     short DoMapIt   ; Error to time out error
15688 00001128 B480                             mov     ah, 80h         ; DSK_TIMEOUT_ERR
15689                                   DoMapIt:
15690 0000112A E97CFC                           jmp     maperror
15691                                   ; ---------------------------------------------------------------------------
15692
15693                                   FormatOk:
15694 0000112D C606[AA05]00                     mov     byte [had_format_error], 0 ; reset the format error flag
15695 00001132 C3                               retn
15696                                   ; ---------------------------------------------------------------------------
15697
15698                                   ; 16/10/2022
15699
15700                                   ; =============================================================================
15701                                   ;
15702                                   ; VerifyTrack:
15703                                   ;
15704                                   ; input: ES:di points to bds for drive
15705                                   ; =============================================================================
15706
15707                                           ; 24/12/2023 - Retro DOS 5.0
15708                                   VerifyTrack:
15709 00001133 1E                               push    ds
15710 00001134 C51E[1200]                       lds     bx, [ptrsav]    ; DS:BX points to request header.
15711 00001138 C55F13                           lds     bx, [bx+19]     ; [bx+IOCTL_REQ.GENERICIOCTL_PACKET]
15712
15713                                           ; Come here with DS:[BX] -> packet, ES:[DI] -> bds
15714
15715 0000113B 8B4F03                           mov     cx, [bx+3]      ; [bx+A_VERIFYPACKET.VP_CYLINDER]
15716 0000113E 8B4701                           mov     ax, [bx+1]      ; [bx+A_VERIFYPACKET.VP_HEAD]
15717 00001141 8B5705                           mov     dx, [bx+5]      ; [bx+A_FORMATPACKET.FP_TRACKCOUNT]
15718 00001144 8A1F                             mov     bl, [bx]        ; [bx+A_FORMATPACKET.FP_SPECIALFUNCTIONS]
15719                                                                   ; Get option flag word
15720 00001146 1F                               pop     ds
15721 00001147 C606[2001]04                     mov     byte [rflag], 4         ; romverify
15722 0000114C 890E[3301]                       mov     [curtrk], cx
15723 00001150 A2[3201]                         mov     [curhd], al     ; ASSUME heads < 256
15724 00001153 8B0E[AA04]                       mov     cx, [sectorspertrack]
15725
15726                                   ; Check specialfunctions to see if DO_FAST_FORMAT has been
15727                                   ; specified if not we should go to the normal track verification
15728                                   ; routine. If fast format has been specified we should get the
15729                                   ; number of tracks to be verified and check it to see if it is
15730                                   ; > 255. If it is then it is an error and we should go to
15731                                   ; VerifyTrack_Err. If not multiply the number of tracks by the
15732                                   ; sectors per track to get the total number of sectors to be
15733                                   ; verified. This should also be less than equal to 255
15734                                   ; otherwise we go to same error exit. If everything is okay
15735                                   ; we initialise cx to the total sectors. use ax as a temporary
15736                                   ; register.
15737
15738                                                           ; Special function requested?
15739 00001157 F6C302                           test    bl, 2           ; DO_FAST_FORMAT
15740 0000115A 7421                             jz      short NormVerifyTrack
15741 0000115C 89D0                             mov     ax, dx          ; Get ax = number of trks to verify
15742 0000115E 08E4                             or      ah, ah
15743 00001160 752C                             jnz     short VerifyTrack_Err ; #tracks > 255
15744 00001162 F6E1                             mul     cl
15745 00001164 08E4                             or      ah, ah
15746 00001166 7526                             jnz     short VerifyTrack_Err ; #sectors > 255
15747 00001168 89C1                             mov     cx, ax
15748                                           ; 24/12/2023
15749 0000116A 26F6453F01                       test    byte [es:di+3Fh], 1 ; PCDOS 7.1 IBMBIO.COM
15750                                           ; 10/12/2022
15751                                           ;test   byte [es:di+35], 1
15752                                           ;;test  word [es:di+35], 1 ; [es:di+BDS.flags]
15753                                                           ; fnon_removable
```

```
15754 0000116F 740C                             jz      short NormVerifyTrack
15755                                                   ; Multitrack operation = on?
15756                                           ; 10/12/2022
15757                                           ; 19/10/2022
15758 00001171 F606[A004]80                     test    byte [multrk_flag], 80h
15759                                           ;test   word [multrk_flag], 80h ; MULTI_TRK_ON
15760                                           ;;test  ds:multrk_flag,       80h ; MULTI_TRK_ON
15761 00001176 7405                             jz      short NormVerifyTrack
15762 00001178 C606[A704]01                     mov     byte [multitrk_format_flag], 1
15763                                   NormVerifyTrack:
15764 0000117D 31C0                             xor     ax, ax          ; 1st sector
15765 0000117F 31DB                             xor     bx, bx
15766 00001181 891E[A804]                       mov     [xfer_seg], bx ; Use 0:0 as the transfer address for verify
15767 00001185 E83F00                           call    TrackIo
15768 00001188 C606[A704]00                     mov     byte [multitrk_format_flag], 0
15769 0000118D C3                               retn
15770                                   ; ---------------------------------------------------------------------------
15771
15772                                   VerifyTrack_Err:
15773 0000118E B401                             mov     ah, 1
15774 00001190 E916FC                           jmp     maperror
15775                                   ; ---------------------------------------------------------------------------
15776
15777                                   ; 16/10/2022
15778
15779                                   ; ============================================================================
15780                                   ;
15781                                   ; ReadTrack:
15782                                   ;
15783                                   ; input: ES:di points to bds for drive
15784                                   ;
15785                                   ; ============================================================================
15786
15787                                   ReadTrack:
15788 00001193 C606[2001]02                     mov     byte [rflag], 2       ; romread
15789 00001198 EB05                             jmp     short ReadWriteTrack
15790                                   ; ---------------------------------------------------------------------------
15791
15792                                   WriteTrack:
15793
15794                                   ; ============================================================================
15795                                   ;
15796                                   ; WriteTrack:
15797                                   ;
15798                                   ; input: ES:di points to bds for drive
15799                                   ;
15800                                   ; ============================================================================
15801
15802 0000119A C606[2001]03                       mov     byte [rflag], 3       ; romwrite
15803
15804                                             ; Fall into ReadWriteTrack
15805
15806                                   ; ============================================================================
15807                                   ;
15808                                   ; readWriteTrack:
15809                                   ;
15810                                   ; input:
15811                                   ;    ES:di points to bds for drive
15812                                   ;    rFlag - 2 for read,3 for write
15813                                   ;
15814                                   ; ============================================================================
15815
15816                                   ReadWriteTrack:
15817                                             ; save bds pointer segment so we can use it to access
15818                                             ; our packet. Notice that this is not the standard register
15819                                             ; assignment for accessing packets
15820
15821                                             ; 19/10/2022
15822 0000119F 06                               push    es
15823 000011A0 C41E[1200]                       les     bx, [ptrsav]   ; ES:BX -> to request header
15824 000011A4 26C45F13                         les     bx, [es:bx+19] ; [es:bx+IOCTL_REQ.GENERICIOCTL_PACKET]
15825 000011A8 268B4703                         mov     ax, [es:bx+3]  ; [es:bx+A_TRACKREADWRITEPACKET.TRWP_CYLINDER]
15826 000011AC A3[3301]                         mov     [curtrk], ax
15827 000011AF 268B4701                         mov     ax, [es:bx+1]  ; [es:bx+A_TRACKREADWRITEPACKET.TRWP_HEAD]
15828 000011B3 A2[3201]                         mov     [curhd], al    ; Assume heads < 256!!!
15829 000011B6 268B4705                         mov     ax, [es:bx+5]  ; [es:bx+A_TRACKREADWRITEPACKET.TRWP_FIRSTSECTOR]
15830 000011BA 268B4F07                         mov     cx, [es:bx+7]  ; [es:bx+A_TRACKREADWRITEPACKET.TRWP_SECTORSTOREADWRITE]
15831 000011BE 26C45F09                         les     bx, [es:bx+9]  ; [es:bx+A_TRACKREADWRITEPACKET.TRWP_TRANSFERADDRESS]
15832                                                                  ; Get transfer address
15833
15834                                             ; we just trashed our packet address, but we no longer care
15835
15836 000011C2 8C06[A804]                       mov     [xfer_seg], es ; Pass transfer       segment
15837 000011C6 07                               pop     es
15838
15839                                             ; Fall into TrackIo
15840
15841                                   ; =============== S U B   R O U T I N E =======================================
15842
15843                                   ; 16/10/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 IO.SYS)
15844
15845                                   ; ============================================================================
15846                                   ;
15847                                   ; TrackIo:
15848                                   ;    performs track read/write/verify
15849                                   ;
15850                                   ;  input:
15851                                   ;     rFlag       - 2 = read
15852                                   ;                   3 = write
15853                                   ;                   4 = verify
15854                                   ;     AX    - Index into track table of first sector to io
15855                                   ;     CX    - Number of sectors to io
15856                                   ;     Xfer_Seg:BX - Transfer address
15857                                   ;     ES:DI       - Pointer to bds
15858                                   ;     CurTrk      - Current cylinder
15859                                   ;     CurHd       - Current head
15860                                   ;
15861                                   ; ============================================================================
15862
15863                                   ; 16/03/2019 - Retro DOS v4.0
15864
15865                                             ; 24/12/2023 - Retro DOS 5.0
15866
15867                                             ; 19/10/2022
15868                                   TrackIo:
15869                                                                  ; Procedure `disk' will pop stack to
15870 000011C7 8926[3501]                       mov     [spsav], sp    ; SpSav and return if error
15871 000011CB E8A9F7                           call    checksingle    ; Ensure correct disk is in drv
15872 000011CE 803E[A905]01                     cmp     byte [media_set_for_format], 1
15873                                                                  ; See if we have already set  disk
15874 000011D3 7407                             jz      short Dptalreadyset ; base table
15875 000011D5 50                               push    ax             ; set up tables       and variables for i/o
15876 000011D6 51                               push    cx
15877 000011D7 E8A0F9                           call    iosetup
```

```
15878 000011DA 59                              pop     cx
15879 000011DB 58                              pop     ax
15880                                  Dptalreadyset:                          ; Point si at the table entry of the
15881 000011DC BE[AC04]                        mov     si, tracktable ; first sector to be io'd
15882                                          ; 24/12/2023
15883                                          ;add    ax, ax         ; PCDOS 7.1 IBMBIO.COM
15884                                          ;add    ax, ax
15885 000011DF D1E0                            shl     ax, 1
15886 000011E1 D1E0                            shl     ax, 1
15887 000011E3 01C6                            add     si, ax
15888
15889                                          ; WE WANT:
15890                                          ; CX to be the number of times we have to loop
15891                                          ; DX to be the number of sectors we read on each iteration
15892
15893 000011E5 BA0100                          mov     dx, 1
15894
15895                                          ; 24/12/2023
15896 000011E8 26F6453F08                      test    byte [es:di+3Fh], 8 ; PCDOS 7.1 IBMBIO.COM
15897                                          ; 12/12/2022
15898                                          ;test   byte [es:di+23h], 8
15899                                          ;;test  word [es:di+35], 8 ; [es:di+BDS.flags]
15900                                                                  ; good_tracklayout
15901 000011ED 7402                            jz      short ionextsector
15902
15903 000011EF 87D1                            xchg    dx, cx         ; HEY! We can read all secs in one blow
15904                                  ionextsector:
15905 000011F1 51                              push    cx
15906 000011F2 52                              push    dx
15907 000011F3 46                              inc     si
15908 000011F4 46                              inc     si             ; Skip over the       cylinder and head in
15909                                                                  ; the track table
15910 000011F5 AC                              lodsb                  ; Get sector ID       from track table
15911 000011F6 A2[3101]                        mov     [cursec], al
15912
15913                                          ; assumptions for a fixed disk multi-track disk     i/o
15914                                          ; 1). In the input CX (# of sectors to go) to TrackIo,
15915                                          ;     only CL is valid.
15916                                          ; 2). Sector size should be set      to 512 bytes.
15917                                          ; 3). Good track layout
15918
15919                                          ; 24/12/2023
15920 000011F9 26F6453F01                      test    byte [es:di+3Fh], 1 ; PCDOS 7.1 IBMBIO.COM
15921                                          ; 12/12/2022
15922                                          ;test   byte [es:di+23h], 1
15923                                          ;;test  word [es:di+35], 1 ; [es:di+BDS.flags]
15924                                                                  ; fnon_removable ; Fixed disk?
15925 000011FE 7414                            jz      short IoRemovable ; No
15926
15927                                          ; 12/12/2022
15928 00001200 F606[A004]80                    test    byte [multrk_flag], 80h
15929                                          ;test   word [multrk_flag], 80h ; MULTI_TRK_ON
15930                                                                          ; Allow multi-track operation?
15931 00001205 740D                            jz      short IoRemovable ; No,don't do that.
15932 00001207 8916[2201]                      mov     [seccnt], dx
15933 0000120B 89D0                            mov     ax, dx
15934 0000120D E823FA                          call    Disk
15935 00001210 5A                              pop     dx
15936 00001211 59                              pop     cx
15937 00001212 F8                              clc
15938 00001213 C3                              retn
15939                                  ; ---------------------------------------------------------------------------
15940
15941                                  IoRemovable:
15942 00001214 AC                              lodsb                  ; Get sector size index       from track
15943                                                                  ; table and save it
15944 00001215 50                              push    ax
15945 00001216 56                              push    si
15946 00001217 1E                              push    ds             ; Save Bios_Data
15947 00001218 50                              push    ax
15948 00001219 8A26[2C01]                      mov     ah, [eot]      ; Preserve whatever might be in      ah
15949                                                                  ; Fetch EOT while ds-> Bios_Data
15950 0000121D C536[2D01]                      lds     si, [dpt]
15951 00001221 884403                          mov     [si+3], al     ; [si+DISK_PARMS.DISK_SECTOR_SIZ]
15952 00001224 886404                          mov     [si+4], ah     ; [si+DISK_PARMS.DISK_EOT]
15953 00001227 58                              pop     ax
15954 00001228 1F                              pop     ds
15955 00001229 88D0                            mov     al, dl
15956 0000122B A3[2201]                        mov     [seccnt], ax
15957 0000122E E802FA                          call    Disk
15958 00001231 5E                              pop     si             ; Advance buffer pointer by adding
15959                                                                  ; sector size
15960                                          ;pop    ax
15961                                          ; 24/12/2023
15962 00001232 59                              pop     cx
15963
15964                                          ; 02/09/2023 (PCDOS 7.1)
15965                                          ;call   SectorSizeIndexToSectorSize
15966                                          ;mov    cl, al ; 24/12/2023
15967 00001233 B88000                          mov     ax, 128
15968 00001236 D3E0                            shl     ax, cl
15969
15970 00001238 01C3                            add     bx, ax
15971 0000123A 5A                              pop     dx
15972 0000123B 59                              pop     cx
15973 0000123C E2B3                            loop    ionextsector
15974 0000123E 803E[A905]01                    cmp     byte [media_set_for_format], 1
15975                                          ;jz     short NoNeedDone
15976                                          ; 12/12/2022
15977 00001243 7404                            je      short NoNeedDone2
15978 00001245 E877F9                          call    done           ; set time of last access, and reset
15979                                                                  ; entries in Dpt.
15980                                  NoNeedDone:
15981 00001248 F8                              clc     ; not necessary ('done' clears cf) ; 24/12/2023
15982                                  NoNeedDone2:
15983 00001249 C3                              retn
15984
15985                                  ; =============== S U B   R O U T I N E =======================================
15986
15987                                  ; ---------------------------------------------------------------------------
15988                                  ;
15989                                  ; The sector size in bytes needs to be converted to an index value for the ibm
15990                                  ; rom. (0=>128,1=>256,2=>512,3=>1024). It is assumed that only these values
15991                                  ; are permissible.
15992                                  ;
15993                                  ; On Input   AX contains sector size in bytes
15994                                  ; On Output  AL Contains index
15995                                  ; All other registers preserved
15996                                  ;
15997                                  ; ---------------------------------------------------------------------------
15998
15999                                  ; 02/09/2023
16000                                  ;SectSizeToSectIndex:
16001                                  ;       cmp     ah, 2          ; (0=>128,1=>256,2=>512,3=>1024)
```

```
16002                                   ;                                       ; examine upper       byte only
16003                                   ;               ja      short OneK
16004                                   ;               mov     al, ah          ; value in AH is the index!
16005                                   ;               retn
16006                                   ;
16007                                   ; ---------------------------------------------------------------------------
16008                                   ;
16009                                   ;OneK:
16010                                   ;               mov     al, 3
16011                                   ;               retn
16012                                   ;
16013                                   ; =============== S U B   R O U TI N E =======================================
16014                                   ;
16015                                   ; 02/09/2023
16016                                   ;SectorSizeIndexToSectorSize:
16017                                   ;               mov     cl, al
16018                                   ;               mov     ax, 128
16019                                   ;               shl     ax, cl
16020                                   ;               retn
16021                                   ;
16022                                   ; =============== S U B   R O U TI N E =======================================
16023                                   ;
16024                                   ; 16/10/2022
16025                                   ;
16026                                   ; ---------------------------------------------------------------------------
16027                                   ;
16028                                   ; SetDASD
16029                                   ;
16030                                   ; Set up the rom for formatting.
16031                                   ; we have to tell the rom bios what type of disk is in the drive.
16032                                   ;
16033                                   ; On Input   - ES:di - Points to bds
16034                                   ;
16035                                   ; ---------------------------------------------------------------------------
16036                                   ;
16037                                               ; 24/12/2023 - Retro DOS 5.0
16038                                               ; (PCDOS 7.1 IBMBIO.COM - BIOSCODE:129Bh)
16039                                               ;
16040                                               ; 19/10/2022
16041                                   SetDasd:
16042 0000124A 803E[AA05]01                         cmp     byte [had_format_error], 1 ;
16043                                                                               ; See if we've previously set dasd type
16044 0000124F 740C                                 jz      short DoSetDasd
16045                                               ; 24/12/2023
16046 00001251 26F6453F80                           test    byte [es:di+3Fh], 80h
16047                                               ; 10/12/2022
16048                                               ;test   byte [es:di+23h], 80h
16049                                               ;;test  word [es:di+23h], 80h ; [es:di+BDS.flags]
16050                                                                               ; set_dasd_true
16051 00001256 7446                                 jz      short DasdHasBeenSet
16052                                               ; 24/12/2023
16053 00001258 2680653F7F                           and     byte [es:di+3Fh], 7Fh
16054                                               ; 10/12/2022
16055                                               ;and    byte [es:di+23h], 7Fh
16056                                               ;;and   word [es:di+23h], 0FF7Fh ; [es:di+BDS.flags]
16057                                                                               ; ~set_dasd_true
16058                                   DoSetDasd:
16059 0000125D C606[AA05]00                         mov     byte [had_format_error], 0 ; Reset it
16060 00001262 C606[3B01]50                         mov     byte [gap_patch], 50h ; Format gap for 48tpi disks
16061 00001267 B004                                 mov     al, 4
16062                                               ; 24/12/2023
16063 00001269 268A653E                             mov     ah, [es:di+3Eh]
16064                                               ; 02/09/2023
16065                                               ;mov    ah, [es:di+22h] ; [es:di+BDS.formfactor]
16066 0000126D 80FC02                               cmp     ah, 2
16067                                               ;cmp    byte [es:di+22h], 2 ; [es:di+BDS.formfactor]
16068                                                                               ; DEV_3INCH720KB
16069 00001270 7414                                 jz      short DoSet
16070                                               ; 24/12/2023
16071 00001272 B001                                 mov     al, 1
16072                                               ;cmp    ah, 1
16073 00001274 38C4                                 cmp     ah, al  ; 1
16074                                               ;cmp    byte [es:di+22h], 1 ; [es:di+BDS.formfactor]
16075                                                                               ; DEV_5INCH96TPI
16076                                               ;jz     short GotBig
16077                                               ; 24/12/2023
16078                                               ;mov    al, 1
16079                                               ;jmp    short DoSet
16080                                               ; 02/09/2023
16081 00001276 750E                                 jnz     short DoSet
16082                                   GotBig:
16083                                               ;mov    al, 2           ; 160/320k in a       1.2 meg drive
16084                                               ; 02/09/2023
16085 00001278 40                                   inc     ax  ; mov al, 2
16086 00001279 803E[A805]00                         cmp     byte [mediatype], 0
16087 0000127E 7506                                 jnz     short DoSet
16088                                               ;mov    al, 3           ; 1.2meg in a 1.2meg drive
16089                                               ; 10/12/2022
16090                                               ;inc    al  ; al = 3
16091                                               ; 18/12/2022
16092 00001280 40                                   inc     ax  ; al = 3
16093 00001281 C606[3B01]54                         mov     byte [gap_patch], 54h
16094                                   DoSet:
16095 00001286 1E                                   push    ds
16096 00001287 56                                   push    si
16097                                               ;
16098                                               ;mov    ds, [zeroseg]  ; Point to interrupt vectors
16099                                               ; 02/09/2023
16100 00001288 31F6                                 xor     si, si
16101 0000128A 8EDE                                 mov     ds, si  ; 0
16102                                               ;
16103 0000128C C5367800                             lds     si, [DSKADR]
16104                                               ;lds    si, [78h]       ; [DSKADR]  (Int 1Eh)
16105                                               ;;lds   si, ds:78h
16106                                               ;
16107 00001290 C644090F                             mov     byte [si+9], 0Fh ;
16108                                                                               ; [si+DISK_PARMS.DISK_HEAD_STTL]
16109 00001294 5E                                   pop     si
16110 00001295 1F                                   pop     ds
16111 00001296 B417                                 mov     ah, 17h
16112 00001298 268A5504                             mov     dl, [es:di+4]
16113 0000129C CD13                                 int     13h             ; DISK - DISK -       SET TYPE (AT,XT2,XT286,CONV,PS
16114                                                                               ; AL = disk type AL = 03h - high-capacity disk in high-capacity   drive
16115                                   DasdHasBeenSet:
16116 0000129E 268A6513                             mov     ah, [es:di+13h]       ; [es:di+BDS.secpertrack]
16117 000012A2 8826[3701]                           mov     [formt_eot], ah
16118 000012A6 C3                                   retn
16119                                   ;
16120                                   ; =============== S U B   R O U TI N E =======================================
16121                                   ;
16122                                   ; 16/10/2022
16123                                   ;
16124                                   ; ---------------------------------------------------------------------------
16125                                   ;
```

```
                                ; Set Media Type for Format
                                ; Performs the int 13 with ah = 18h to see if the medium described in the
                                ; BPB area in the BDS can be handled by the rom.
                                ; On Input, ES:DI -> current BDS.
                                ; The status of the operation is returned in AL
                                ;
                                ;   - 0 - if the support is available,and the combination is valid.
                                ;   - 1 - no rom support
                                ;   - 2 - illegal combination
                                ;   - 3 - no media present (rom support exists but cannot determine now)
                                ;
                                ; Flags also may be altered. All other registers preserved.
                                ; If the call to rom returns no error,then the current Dpt is "replaced" by
                                ; the one returned by the rom. This is Done by changing the pointer in [Dpt]
                                ; to the one returned. the original pointer to the disk base table is stored
                                ; in TempDpt, until it is restored.
                                ;
                                ; --------------------------------------------------------------------------

                                        ; 24/12/2023 - Retro DOS 5.0

                                        ; 19/10/2022
                                SetMediaForFormat:
000012A7 51                             push    cx
000012A8 52                             push    dx

                                        ; If we have a format error, then do not change Dpt, TempDpt.
                                        ; but we need to call int 13h, ah=18h again.

000012A9 803E[AA05]01                   cmp     byte [had_format_error], 1
000012AE 7425                           jz      short SkipSaveDskAdr
000012B0 30C0                           xor     al, al          ; If already done return 0
000012B2 803E[A905]01                   cmp     byte [media_set_for_format], 1
000012B7 7502                           jnz     short DoSetMediaForFormat
000012B9 EB7D                           jmp     SetMediaRet     ; Media already set
                                ; --------------------------------------------------------------------------

                                DoSetMediaForFormat:
000012BB 06                             push    es
000012BC 56                             push    si

                                        ; 02/09/2023
                                        ;mov    es, [zeroseg]  ; Point to interrupt vectors
000012BD 31F6                           xor     si, si ; 0
000012BF 8EC6                           mov     es, si

000012C1 26C4367800                     les     si, [es:DSKADR]
                                        ;les    si, es:78h     ; [es:DSKADR]
                                                                ; Get pointer to disk base table
000012C6 8936[2D01]                     mov     [dpt], si
000012CA 8C06[2F01]                     mov     [dpt+2], es    ; Save pointer to table

                                        ; Initialize the head settle time to 0Fh. See the offsets
                                        ; given in dskprm.inc.

000012CE 26C644090F                     mov     byte [es:si+9], 0Fh ; [es:si+DISK_PARMS.DISK_HEAD_STTL]
000012D3 5E                             pop     si
000012D4 07                             pop     es
                                SkipSaveDskAdr:
                                        ; 24/12/2023
000012D5 268B4D41                       mov     cx, [es:di+41h]       ; (PCDOS 7.1 IBMBIO.COM)
                                        ;mov    cx, [es:di+25h]       ; [es:di+BDS.cylinders]
000012D9 49                             dec     cx
000012DA 80E503                         and     ch, 3
000012DD D0CD                           ror     ch, 1
000012DF D0CD                           ror     ch, 1
000012E1 86E9                           xchg    ch, cl
000012E3 260A4D13                       or      cl, [es:di+13h]       ; [es:di+BDS.secpertrack]
000012E7 268A5504                       mov     dl, [es:di+4]  ; [es:di+BDS.drivenum]
000012EB 06                             push    es
000012EC 1E                             push    ds
000012ED 56                             push    si
000012EE 57                             push    di
000012EF B418                           mov     ah, 18h
000012F1 CD13                           int     13h             ; DISK - SET MEDIA TYPE       FOR FORMAT (AT model 3x9,XT2,XT286,PS)
                                                                ; DL = drive number, CH       = lower 8 bits of number of tracks, CL = sectors
per track
000012F3 7231                           jc      short FormaStatErr
000012F5 803E[AA05]01                   cmp     byte [had_format_error], 1
000012FA 7423                           jz      short skip_disk_base_setting
000012FC 06                             push    es              ; Save segment returned       by the rom

                                        ; 02/09/2023
                                        ;mov    es, [zeroseg]  ; Point to interrupt vector segment
000012FD 31F6                           xor     si, si
000012FF 8EC6                           mov     es, si ; 0
00001301 06                             push    es ; * ; 02/09/2023

00001302 26C4367800                     les     si, [es:DSKADR]
                                        ;les    si, es:78h     ; [es:DSKADR] (Int 1Eh)
                                                                ; Get current disk base       table
00001307 8936[AB05]                     mov     [tempdpt], si
0000130B 8C06[AD05]                     mov     [tempdpt+2], es ; Save it

                                        ; 02/09/2023
                                        ;;mov   es, [zeroseg]
                                        ;xor    si, si ; 0
                                        ;mov    es, si
0000130F 07                             pop     es ; * ; 02/09/2023

                                        ;mov    es:78h,di
00001310 26893E7800                     mov     [es:DSKADR], di
                                        ;pop    word ptr es:7Ah       ; replace with one returned by rom
00001315 268F067A00                     pop     word [es:DSKADR+2]
0000131A C606[A905]01                   mov     byte [media_set_for_format], 1
                                skip_disk_base_setting:
0000131F 30C0                           xor     al, al          ; Legal combination + rom support code
                                        ;mov    ds:had_format_error, al       ; Reset the flag
00001321 A2[AA05]                       mov     [had_format_error], al
00001324 EB0E                           jmp     short PopStatRet
                                ; --------------------------------------------------------------------------

                                FormaStatErr:
                                        ; 10/12/2022
00001326 B003                           mov     al, 3

00001328 80FC0C                         cmp     ah, 0Ch         ; DSK_ILLEGAL_COMBINATION
                                                                ; Illegal combination =       0Ch
0000132B 7406                           jz      short FormatStatIllegalComb
0000132D 80FC80                         cmp     ah, 80h         ; DSK_TIMEOUT_ERR
00001330 7402                           jz      short FormatStatTimeOut
                                        ; 10/12/2022
                                        ;dec    al
                                        ; 18/12/2022
```

```
16249 00001332 48                          dec     ax
16250                                      ; al = 2
16251                                      ;mov    al, 1           ; Function not supported.
16252                                      ;jmp    short PopStatRet
16253                                ; ---------------------------------------------------------------------------
16254
16255                                FormatStatIllegalComb:
16256                                      ; 10/12/2022
16257                                      ;dec    al      ; 3 -> 2 or 2 -> 1
16258                                      ; 18/12/2022
16259 00001333 48                          dec     ax
16260                                      ; al = 2
16261                                      ;mov    al, 2           ; Function supported, but
16262                                                              ; Illegal sect/trk,trk combination.
16263                                      ; 10/12/2022
16264                                      ;jmp    short PopStatRet
16265                                ; ---------------------------------------------------------------------------
16266
16267                                FormatStatTimeOut:
16268                                      ; 10/12/2022
16269                                      ; al = 3
16270                                      ;mov    al, 3           ; Function supported, but
16271                                                              ; Media not present.
16272                                PopStatRet:
16273 00001334 5F                          pop     di
16274 00001335 5E                          pop     si
16275 00001336 1F                          pop     ds
16276 00001337 07                          pop     es
16277                                SetMediaRet:
16278 00001338 5A                          pop     dx
16279 00001339 59                          pop     cx
16280 0000133A C3                          retn
16281
16282                                ; =============== S U B   R O U TI N E =======================================
16283
16284                                ; 16/10/2022
16285
16286                                ; ---------------------------------------------------------------------------
16287                                ;
16288                                ; RESET THE DRIVE
16289                                ;
16290                                ; we also set [Step_Drv] to -1 to force the main disk routine to use the
16291                                ; slow head settle time for the next operation. this is because the reset
16292                                ; operation moves the head to cylinder 0,so we need to do a seek the next
16293                                ; time around - there is a problem with 3.5" drives in that the head does
16294                                ; not settle down in time,even for read operations!!
16295                                ;
16296                                ; ---------------------------------------------------------------------------
16297
16298                                ResetDisk:
16299 0000133B 50                          push    ax
16300
16301                                      ; 02/09/2023
16302 0000133C B80100                      mov     ax, 1 ; PCDOS 7.1
16303 0000133F 3806[A905]                  cmp     [media_set_for_format], al ; 1
16304                                      ;cmp    byte [media_set_for_format], 1
16305                                                              ; Reset while formatting?
16306 00001343 7503                        jnz     short ResetDisk_cont
16307                                                              ; Then verify operation in "fmt & vrfy"
16308                                      ;mov    byte [had_format_error], 1 ; Might have failed.
16309 00001345 A2[AA05]                    mov     [had_format_error], al ; 1
16310                                ResetDisk_cont:
16311                                      ; 02/09/2023 (ah=0)
16312                                      ;xor    ah, ah          ; So signals that we had a format error
16313 00001348 CD13                        int     13h             ; DISK - RESET DISK SYSTEM
16314                                                              ; DL = drive (if bit 7 is set both hard     disks and floppy disks reset)
16315 0000134A C606[7600]FF                mov     byte [step_drv], 0FFh ; -1
16316                                                              ; Zap up the speed
16317 0000134F 58                          pop     ax
16318 00001350 C3                          retn
16319
16320                                ; =============== S U B   R O U TI N E =======================================
16321
16322                                ; 16/10/2022
16323
16324                                ; ---------------------------------------------------------------------------
16325                                ;
16326                                ; This routine sets up the drive parameter table with the values needed for
16327                                ; format,does an int 13. values in Dpt are restored after a verify is done.
16328                                ;
16329                                ; on entry  -      ES:DI - points to bds for the drive
16330                                ;          Xfer_Seg:BX - points to trkbuf
16331                                ;          AL     - number of sectors
16332                                ;          AH     - int 13 function code
16333                                ;          CL     - sector number for verify
16334                                ;          DS     - Bios_Data
16335                                ;
16336                                ; ON EXIT   -      DS,DI,ES,BX remain unchanged.
16337                                ;          AX and flags are the results of the int 13
16338                                ;
16339                                ; ---------------------------------------------------------------------------
16340
16341                                      ; 24/12/2023 - Retro DOS 5.0
16342
16343                                      ; 19/10/2022
16344                                ToRom:
16345 00001351 53                          push    bx
16346 00001352 56                          push    si
16347
16348                                      ; Compaq bug fix - check whether we are using new ROM
16349                                      ; functionality to set up format, not merely if it exists.
16350                                      ; This was formerly a check against [new_rom]
16351
16352 00001353 F606[A905]01                test    byte [media_set_for_format], 1
16353 00001358 7534                        jnz     short GotValidDpt
16354 0000135A 50                          push    ax
16355 0000135B 06                          push    es              ; Save bds segment
16356                                      ; 24/12/2023
16357 0000135C 26807D3E02                  cmp     byte [es:di+3Eh], 2
16358                                      ;cmp    byte [es:di+22h], 2 ; [es:di+BDS.formfactor]
16359                                                              ; ffSmall ; is it a 3.5" drive?
16360                                      ; 24/12/2023
16361                                      ;pushf  ; not necessary        ; (Save the cmp result)
16362 00001361 8E06[1A00]                  mov     es, [zeroseg]
16363                                      ;les    si, es:78h      ; Get pointer to disk base table
16364 00001365 26C4367800                  les     si, [es:DSKADR]
16365                                      ;mov    word ptr ds:dpt, si
16366                                      ;mov    word ptr ds:dpt+2, es ;        Save pointer to table
16367 0000136A 8936[2D01]                  mov     [dpt], si
16368 0000136E 8C06[2F01]                  mov     [dpt+2], es     ; Save pointer to table
16369
16370 00001372 A0[3701]                    mov     al, [formt_eot]
16371 00001375 26884404                    mov     [es:si+4], al   ; [es:si+DISK_PARMS.DISK_EOT]
16372 00001379 A0[3B01]                    mov     al, [gap_patch]
```

```
16373 0000137C 26884407                         mov     [es:si+7], al  ; [es:si+DISK_PARMS.DISK_FORMT_GAP]
16374                                                                  ; Important for       format
16375 00001380 26C644090F                       mov     byte [es:si+9], 0Fh ; [es:si+DISK_PARMS.DISK_HEAD_STTL]
16376                                                                  ; Assume we are       doing a seek operation
16377                                                                  ; Setup motor start correctly for 3.5" drives
16378                                           ; 24/12/2023
16379                                           ;popf                  ; Get result of       earlier cmp
16380 00001385 7505                             jnz     short MotorStrtOK
16381 00001387 26C6440A04                       mov     byte [es:si+0Ah], 4 ; [es:si+DISK_PARMS.DISK_MOTOR_STRT]
16382                                   MotorStrtOK:
16383 0000138C 07                               pop     es              ; Restore bds segment
16384 0000138D 58                               pop     ax
16385                                   GotValidDpt:
16386 0000138E 8B16[3901]                       mov     dx, [trknum]    ; Set track number
16387 00001392 88D5                             mov     ch, dl          ; Set low 8 bits in ch
16388 00001394 268A5504                         mov     dl, [es:di+4]   ; Set drive number
16389 00001398 8A36[3801]                       mov     dh, [hdnum]     ; Set head number
16390 0000139C 06                               push    es              ; Save bds segment
16391 0000139D 8E06[A804]                       mov     es, [xfer_seg]
16392 000013A1 CD13                             int     13h             ; DISK -
16393 000013A3 07                               pop     es              ; Restore bds segment
16394 000013A4 5E                               pop     si
16395 000013A5 5B                               pop     bx
16396 000013A6 C3                               retn
16397
16398                                   ; ---------------------------------------------------------------------------
16399
16400                                   ; 16/10/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 IO.SYS)
16401                                   ; 24/12/2023 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMBIO.COM)
16402
16403                                   ; BIOSCODE:1124h (MSDOS 6.21, IO.SYS)
16404                                   ; PCDOS 7.1 IBMBIO.COM - BIOSCODE:1404h
16405
16406                                   ; ============================================================================
16407                                   ;
16408                                   ; get the owner of the physical drive represented by the logical drive in al.
16409                                   ; the assumption is that we **always** keep track of the owner of a drive!!
16410                                   ; if this is not the case, the system may hang, just following the linked list.
16411                                   ;
16412                                   ; ============================================================================
16413
16414                                           ; 24/12/2023 - Retro DOS 5.0
16415
16416                                           ; 19/10/2022
16417                                   ioctl_getown:
16418 000013A7 E8FAF1                           call    SetDrive
16419 000013AA 268A4504                         mov     al, [es:di+4]  ; [es:di+BDS.drivenum]
16420                                                                  ; Get physical drive number
16421 000013AE C43E[1901]                       les     di, [start_bds] ; Get start of bds chain
16422                                   ownloop:
16423 000013B2 26384504                         cmp     [es:di+4], al  ; [es:di+BDS.drivenum]
16424 000013B6 7507                             jnz     short getnextBDS
16425                                           ; 24/12/2023
16426 000013B8 26F6453F20                       test    byte [es:di+3Fh], 20h ; (PCDOS 7.1 IBMBIO.COM)
16427                                           ; 10/12/2022
16428                                           ;test   byte [es:di+23h], 20h
16429                                           ;;test  word [es:di+23h], 20h ; [es:di+BDS.flags]
16430                                                                  ; fi_own_physical
16431 000013BD 7514                             jnz     short exitown
16432                                   getnextBDS:
16433 000013BF 26C43D                           les     di, [es:di]    ; [es:di+BDS.link]
16434 000013C2 EBEE                             jmp     short ownloop
16435                                   ; ---------------------------------------------------------------------------
16436
16437                                   ; ============================================================================
16438                                   ;
16439                                   ; set the ownership of the physical drive represented by the logical drive
16440                                   ; in al to al.
16441                                   ;
16442                                   ; ============================================================================
16443
16444                                           ; 24/12/2023 - Retro DOS 5.0
16445
16446                                           ; 19/10/2022
16447                                   ioctl_setown:
16448 000013C4 E8DDF1                           call    SetDrive
16449 000013C7 C606[7A00]01                     mov     byte [fsetowner], 1
16450                                                                  ; set flag for CheckSingle to look at.
16451 000013CC E8A8F5                           call    checksingle
16452                                           ; 02/09/2023
16453 000013CF FE0E[7A00]                       dec     byte [fsetowner] ; 0
16454                                           ;mov    byte [fsetowner], 0
16455                                                                  ; set ownership       of drive reset flag
16456                                           ; Fall into ExitOwn
16457
16458                                   ; ============================================================================
16459                                   ;
16460                                   ; if there is only one logical drive assigned to this physical drive, return
16461                                   ; 0 to user to indicate this. Enter with ES:di -> the owner's bds.
16462                                   ;
16463                                   ; ============================================================================
16464
16465                                           ; 24/12/2023 - Retro DOS 5.0
16466                                   exitown:
16467 000013D3 30C9                             xor     cl, cl
16468                                           ; 24/12/2023
16469 000013D5 26F6453F10                       test    byte [es:di+3Fh], 10h ; (PCDOS 7.1 IBMBIO.COM)
16470                                           ; 12/12/2022
16471                                           ;test   byte [es:di+23h], 10h
16472                                           ;;test  word [es:di+23h], 10h ; [es:di+BDS.flags]
16473                                                                  ; fi_am_mult
16474 000013DA 7406                             jz      short exitnomult
16475 000013DC 268A4D05                         mov     cl, [es:di+5]  ; [es:di+BDS.drivelet]
16476                                                                  ; Get logical drive number
16477                                                                  ; Get it 1-based
16478 000013E0 FEC1                             inc     cl
16479                                   exitnomult:
16480 000013E2 C51E[1200]                       lds     bx, [ptrsav]
16481 000013E6 884F01                           mov     [bx+1],cl      ; [bx+unit]
16482                                                                  ; Exit normal termination
16483                                           ; 12/12/2022
16484                                           ; cf=0
16485                                           ;clc
16486 000013E9 C3                               retn
16487
16488                                   ; =============== S U B   R O U TI N E =======================================
16489
16490                                   ; 16/10/2022
16491
16492                                   ; ---------------------------------------------------------------------------
16493                                   ;
16494                                   ; moves the old Dpt that had been saved in TempDpt back to Dpt. this is done
16495                                   ; only if the first byte of TempDpt is not -1.
16496                                   ; all registers (including flags) are preserved.
```

```
16497                                   ;
16498                                   ; ---------------------------------------------------------------------------
16499
16500                                             ; 24/12/2023
16501                                             ; 19/10/2022
16502                                   RestoreOldDpt:
16503                                             ; if we have already restored the disk base table earlier,
16504                                             ; do not do it again.
16505
16506 000013EA 50                                  push    ax
16507 000013EB 30C0                                xor     al, al
16508 000013ED A2[AA05]                            mov     [had_format_error], al; Reset flag and
16509 000013F0 8606[A905]                          xchg    al, [media_set_for_format] ; get current flag setting
16510 000013F4 08C0                                or      al, al
16511 000013F6 7418                                jz      short DontRestore
16512 000013F8 56                                  push    si
16513 000013F9 1E                                  push    ds
16514 000013FA 06                                  push    es
16515 000013FB C536[AB05]                          lds     si, [tempdpt]
16516
16517                                             ; 17/10/2022
16518                                             ;mov    es, [cs:BIOSDATAWORD]
16519                                             ;;mov   es, word ptr cs:BIOSDATAWORD ; [cs:Bios_Data_Word]
16520                                             ;mov    es, [es:zeroseg]
16521                                             ;;mov   es, es:zeroseg ; CAS -- bleeeech!
16522
16523                                             ; 24/12/2023
16524 000013FF 31C0                                xor     ax, ax
16525 00001401 8EC0                                mov     es, ax ; 0
16526
16527                                             ;mov    es:78h, si      ; [es:DSKADR]  (Int 1Eh)
16528 00001403 2689367800                          mov     [es:DSKADR], si
16529                                             ;mov    word ptr es:7Ah, ds ; [es:DSKADR+2]
16530 00001408 268C1E7A00                          mov     [es:DSKADR+2], ds
16531 0000140D 07                                  pop     es
16532 0000140E 1F                                  pop     ds
16533 0000140F 5E                                  pop     si
16534                                   DontRestore:
16535 00001410 58                                  pop     ax
16536                                             ; 12/12/2022
16537                                             ; cf=0
16538                                             ;clc                    ;  Clear carry
16539 00001411 C3                                  retn
16540
16541                                   ; ---------------------------------------------------------------------------
16542
16543                                   ; 16/10/2022
16544
16545                                   ; ===========================================================================
16546                                   ;   get media id
16547                                   ; ===========================================================================
16548                                   ;
16549                                   ; FUNCTION: get the volume label,the system id and the serial number from
16550                                   ;       the media that has the extended boot record.
16551                                   ;       for the conventional media,this routine will return "unknown
16552                                   ;       media type" error to dos.
16553                                   ;
16554                                   ; INPUT :   ES:di -> bds table for this drive.
16555                                   ;
16556                                   ; OUTPUT:   the request packet filled with the information,if not carry.
16557                                   ;       if carry set,then al contains the device driver error number
16558                                   ;       that will be returned to dos.
16559                                   ;       register DS,DX,AX,CX,DI,SI destroyed.
16560                                   ;
16561                                   ; SUBROUTINES TO BE CALLED:
16562                                   ;   BootIo:NEAR
16563                                   ;
16564                                   ; LOGIC:
16565                                   ;   to recognize the extended boot record,this logic will actually
16566                                   ;   access the boot sector even if it is a hard disk.
16567                                   ;   note:the valid extended bpb is recognized by looking at the mediabyte
16568                                   ;   field of bpb and the extended boot signature.
16569                                   ;
16570                                   ; {
16571                                   ;   get logical drive number from bds table;
16572                                   ;   rFlag = read operation;
16573                                   ;   BootIo;         /*get the media boot record into the buffer
16574                                   ;   if (no error) then
16575                                   ;        if (extended boot record) then
16576                                   ;          { set volume label,volume serial number and system id
16577                                   ;            of the request packet to those of the boot record;
16578                                   ;          };
16579                                   ;        else               /*not an extended bpb */
16580                                   ;          { set register al to "unknown media.." error code;
16581                                   ;            set carry bit;
16582                                   ;          };
16583                                   ;   else
16584                                   ;        ret;      /*already error code is set in the register al
16585                                   ;
16586                                   ; ===========================================================================
16587
16588                                   ;size_of_EXT_BOOT_SERIAL equ 4
16589                                   ;;size_of_EXT_BOOT_VOL_LABEL equ 11
16590                                   ;;size_of_EXT_SYSTEM_ID equ 8
16591
16592                                             ; 24/12/2023 - Retro DOS 5.0
16593                                             ; (PCDOS 7.1 IBMBIO.COM - BIOSCODE:1478h)
16594
16595                                             ; 19/10/2022
16596                                   GetMediaId:
16597 00001412 E8B000                              call    ChangeLineChk
16598 00001415 268A4505                            mov     al, [es:di+5]  ; [es:di+BDS.drivelet] ; Logical drive number
16599 00001419 C606[2001]02                        mov     byte [rflag], 2       ; Read operation
16600 0000141E E88C00                              call    BootIo         ; Read boot sector into      DiskSector
16601 00001421 722E                                jb      short IOCtl_If1
16602                                                                     ; Valid? (0F0h-0FFh?)
16603 00001423 803E[6701]F0                        cmp     byte [disksector+15h], 0F0h
16604                                                                     ; [disksector+EXT_BOOT.BPB+EBPB.MEDIADESCRIPTOR]
16605                                             ;jb     short IOCtl_If2       ; brif not valid (0F0h - 0FFh)
16606                                             ; 24/12/2023
16607 00001428 7225                                jb      short IOCtl_If7
16608
16609                                             ; 24/12/2023
16610                                             ; 10/12/2022
16611                                             ;mov    si, disksector+26h
16612                                             ;;;
16613                                             ; 24/12/2023
16614                                             ;mov    si, disksector+43h ; BS_FAT32_VolID
16615 0000142A BE[9401]                            mov     si, disksector+42h ; BS_FAT32_BootSig ; 24/12/2023
16616 0000142D 833E[6801]00                        cmp     word [disksector+16h], 0 ; BPB.FATSz16
16617 00001432 7403                                jz      short IOCtl_If3 ; FAT32 fs
16618 00001434 83EE1C                              sub     si, 1Ch         ; FAT (12-16) fs ; 43h-1Ch = 27h ; BS_VolID
16619                                             ; si = disksector+26h = BS_BootSig ; 24/12/2023
16620                                   IOCtl_If3:
```

```
16621                                   ;cmp    byte [si-1], 29h ; BS_BootSig or BS_FAT32_BootSig
16622                                   ;;;
16623 00001437 803C29                  cmp     byte [si], 29h
16624                                   ;cmp    byte [disksector+26h], 29h ; [disksector+EXT_BOOT.SIG]
16625                                                   ; EXT_BOOT_SIGNATURE
16626 0000143A 7512                    jne     short IOCtl_If2         ; not extended boot record
16627 0000143C C43E[1200]              les     di, [ptrsav]    ; es:di points to request header
16628 00001440 26C47F13                les     di, [es:bx+19] ; [es:bx+IOCTL_REQ.GENERICIOCTL_PACKET]
16629                                   ; 10/12/2022
16630                                   ;mov    si, disksector+27h ; disksector+EXT_BOOT.SERIAL
16631 00001444 46                      inc     si
16632                                   ; 24/12/2023
16633                                   ; si = disksector+27h (BS_VolID)
16634                                   ;      or disksector+43h (BS_FAT32_VolID)
16635
16636 00001445 83C702                  add     di, 2           ; A_MEDIA_ID_INFO.MI_SERIAL
16637                         IOCtl_If4:        ; 24/12/2023
16638 00001448 B91700                  mov     cx, 23          ; size_of_EXT_BOOT_SERIAL
16639                                                   ; L+size_of_EXT_BOOT_VOL_LABEL
16640                                                   ; +size_of_EXT_SYSTEM_ID
16641 0000144B F3A4                    rep movsb               ; Move from Bios_Data into request packet
16642
16643                                   ; 10/12/2022
16644                                   ; cf = 0
16645                                   ;clc
16646
16647 0000144D C3                      retn
16648                         ; ---------------------------------------------------------------------------
16649
16650                                   ; 24/12/2023
16651                         IOCtl_If2:
16652 0000144E F9                      stc
16653                         IOCtl_If7:
16654 0000144F B007                    mov     al, 7           ; error_unknown_media
16655                                   ;stc
16656                         IOCtl_If6:
16657                         IOCtl_If1:
16658 00001451 C3                      retn
16659                         ; ---------------------------------------------------------------------------
16660
16661                         ; 16/10/2022
16662
16663                         ; ============================================================================
16664                         ;  set media id
16665                         ; ============================================================================
16666
16667                         ; function: set the volume label, the system id and the serial number of
16668                         ;       the media that has the extended boot record.
16669                         ;       for the conventional media, this routine will return "unknown
16670                         ;       media.." error to dos.
16671                         ;       this routine will also set the corresponding informations in
16672                         ;       the bds table.
16673                         ;
16674                         ; input :   ES:di -> bds table for this drive.
16675                         ;
16676                         ; output:   the extended boot record in the media will be set according to
16677                         ;       the request packet.
16678                         ;       if carry set, then al contains the device driver error number
16679                         ;       that will be returned to dos.
16680                         ;
16681                         ; subroutines to be called:
16682                         ;   BootIo:NEAR
16683                         ;
16684                         ; logic:
16685                         ;
16686                         ; {
16687                         ;   get drive_number from bds;
16688                         ;   rFlag = "read operation";
16689                         ;   BootIo;
16690                         ;   if (no error) then
16691                         ;        if (extended boot record) then
16692                         ;          { set volume label,volume serial number and system id
16693                         ;            of the boot record to those of the request packet;
16694                         ;            rFlag = "write operation";
16695                         ;            get drive number from bds;
16696                         ;            BootIo;        /*write it back*/
16697                         ;          };
16698                         ;        else               /*not an extended bpb */
16699                         ;          { set register al to "unknown media.." error code;
16700                         ;            set carry bit;
16701                         ;            ret;  /*return back to caller */
16702                         ;          };
16703                         ;   else
16704                         ;        ret;              /*already error code is set */
16705                         ;
16706                         ; ============================================================================
16707
16708                                   ; 24/12/2023 - Retro DOS 5.0
16709
16710                                   ; 19/10/2022
16711                         SetMediaId:
16712 00001452 E87000                  call    ChangeLineChk
16713 00001455 268A4505                mov     al, [es:di+5]  ; [es:di+BDS.drivelet]
16714                                                   ; Logical drive       number
16715 00001459 88C2                    mov     dl, al
16716 0000145B C606[2001]02            mov     byte [rflag], 2       ; romread
16717 00001460 52                      push    dx
16718 00001461 E84900                  call    BootIo          ; Read boot sec       to Bios_Data:DiskSector
16719 00001464 5A                      pop     dx
16720 00001465 72EA                    jb      short IOCtl_If6
16721                                                   ; Valid? (0F0h-0FFh?)
16722 00001467 803E[6701]F0            cmp     byte [disksector+15h], 0F0h
16723                                                   ; [disksector+EXT_BOOT.BPB+EBPB.MEDIADESCRIPTOR]
16724 0000146C 72E1                    jb      short IOCtl_If7       ; Brif not
16725
16726                                   ; 24/12/2023
16727                                   ;cmp    byte [disksector+26h], 29h ; [disksector+EXT_BOOT.SIG]
16728                                   ;                       ; EXT_BOOT_SIGNATURE
16729                                   ;jnz    short IOCtl_If7       ; not extended boot record
16730
16731 0000146E 06                      push    es              ; Save BDS pointer
16732 0000146F 57                      push    di
16733 00001470 1E                      push    ds              ; Point ES To boot record
16734 00001471 07                      pop     es
16735
16736                                   ; 24/12/2023
16737                                   ;;;
16738                                   ;mov    di, disksector+43h ; disksector+EXT_BOOT.SERIAL
16739 00001472 BF[9401]                mov     di, disksector+42h ; BS_FAT32_BootSig ; 24/12/2023
16740 00001475 833E[6801]00            cmp     word [disksector+16h], 0 ; BPB.FATSz16
16741 0000147A 7403                    jz       short IOCtl_If5       ; FAT32 fs
16742 0000147C 83EF1C                  sub     di, 1Ch         ; 67-28 ; offset disksektor+27h
16743                                   ; di = disksector+26h = BS_BootSig ; 24/12/2023
16744                         IOCtl_If5:
```

```
16745                                   ;cmp    byte [di-1], 29h ; BS_BootSig or BS_FAT32_BootSig
16746 0000147F 803D29                   cmp     byte [di], 29h
16747 00001482 7404                     je      short IOCtl_If8
16748 00001484 5F                       pop     di              ; not extended boot record
16749 00001485 07                       pop     es
16750                                   ;jmp    short IOCtl_If7
16751                                   ; 24/12/2023
16752 00001486 EBC6                     jmp     short IOCtl_If2
16753                                 IOCtl_If8:
16754                                   ;;;
16755                                   ; 24/12/2023
16756                                   ;mov    di, disksector+27h ; disksector+EXT_BOOT.SERIAL
16757 00001488 47                       inc     di
16758                                   ; di = disksector+27h (BS_VolID)
16759                                   ;      or disksector+43h (BS_FAT32_VolID)
16760
16761 00001489 C536[1200]               lds     si, [ptrsav]    ; ds:si points to request header.
16762 0000148D C57413                   lds     si, [si+19]     ; [si+IOCTL_REQ.GENERICIOCTL_PACKET]
16763 00001490 83C602                   add     si, 2           ; A_MEDIA_ID_INFO.MI_SERIAL
16764
16765                                   ; 24/12/2023
16766                                   ;mov    cx, 23          ; size_of_EXT_BOOT_SERIAL
16767                                   ;                       ; +size_of_EXT_BOOT_VOL_LABEL
16768                                   ;                       ; +size_of_EXT_SYSTEM_ID
16769                                   ;rep movsb
16770 00001493 E8B2FF                   call    IOCtl_If4       ; copy volume serial, label and system id
16771
16772 00001496 06                       push    es              ; point ds back to Bios_Data
16773 00001497 1F                       pop     ds
16774 00001498 5F                       pop     di              ; restore bds pointer
16775 00001499 07                       pop     es
16776 0000149A E8B2F3                   call    mov_media_ids   ; update the bds media id info.
16777 0000149D 88D0                     mov     al, dl
16778 0000149F C606[2001]03             mov     byte [rflag], 3       ; romwrite
16779 000014A4 E80600                   call    BootIo          ; write it back.
16780 000014A7 C606[1E01]FF             mov     byte [tim_drv], 0FFh
16781                                                           ; make sure chk_media check the driver
16782                                                           ; return with error code from BootIo
16783 000014AC C3                               retn
16784                                 ; ---------------------------------------------------------------------------
16785
16786                                           ; 24/12/2023
16787                                 ;IOCtl_If7:
16788                                 ;         mov     al, 7           ; error_unknown_media
16789                                 ;         stc
16790                                 ;IOCtl_If6:
16791                                 ;         retn
16792
16793                                 ; =============== S U B   R O U T I N E =======================================
16794
16795                                 ; 16/10/2022
16796
16797                                 ; ---------------------------------------------------------------------------
16798                                 ;   BootIo
16799                                 ; ---------------------------------------------------------------------------
16800                                 ;
16801                                 ; function: read/write the boot record into boot sector.
16802                                 ;
16803                                 ; input :
16804                                 ;       al=logical drive number
16805                                 ;       rFlag = operation (read/write)
16806                                 ;
16807                                 ; output:   for read operation,the boot record of the drive specified in bds
16808                                 ;       be read into the DiskSector buffer.
16809                                 ;       for write operation,the DiskSector buffer image will be written
16810                                 ;       to the drive specified in bds.
16811                                 ;       if carry set,then al contains the device driver error number
16812                                 ;       that will be returned to dos.
16813                                 ;       AX,CX,DX register destroyed.
16814                                 ;       if carry set,then al will contain the error code from DiskIO.
16815                                 ;
16816                                 ; subroutines to be called:
16817                                 ;   DiskIO:NEAR
16818                                 ;
16819                                 ; logic:
16820                                 ;
16821                                 ; {
16822                                 ;   first_sector = 0;       /*logical sector 0 is the boot sector */
16823                                 ;   sectorcount = 1;        /*read 1 sector only */
16824                                 ;   buffer = DiskSector;    /*read it into the DiskSector buffer */
16825                                 ;   call DiskIO (rFlag,drive_number,first_sector,sectorcount,buffer);
16826                                 ; }
16827                                 ; ===========================================================================
16828
16829                                           ; 19/10/2022
16830                                 BootIo:
16831 000014AD 06                               push    es
16832 000014AE 57                               push    di
16833 000014AF 53                               push    bx
16834 000014B0 1E                               push    ds
16835 000014B1 07                               pop     es              ; Point ES: to Bios_Data
16836
16837                                           ; Call DiskIO to read/write the boot sec. The parameters which
16838                                           ; need to be initialized for this subroutine out here are
16839                                           ; - Transfer address to Bios_Data:DiskSector
16840                                           ; - Low sector needs to be initalized to 0. this is a reg. param
16841                                           ; - Hi sector in [Start_Sec_H] needs to be initialised to 0.
16842                                           ; - Number of sectors <-- 1
16843
16844 000014B2 BF[5201]                         mov     di, disksector ; es:di -> transfer address
16845 000014B5 31D2                             xor     dx, dx          ; First sector (h) -> 0
16846 000014B7 8916[9C04]                       mov     [start_sec_h], dx ; Start sector (h) -> 0
16847 000014BB B90100                           mov     cx, 1
16848 000014BE E850F5                           call    diskio
16849 000014C1 5B                               pop     bx
16850 000014C2 5F                               pop     di
16851 000014C3 07                               pop     es
16852 000014C4 C3                               retn
16853
16854                                 ; =============== S U B   R O U T I N E =======================================
16855
16856                                 ; 16/10/2022
16857
16858                                 ; ---------------------------------------------------------------------------
16859                                 ;   ChangeLineChk
16860                                 ; ---------------------------------------------------------------------------
16861                                 ;
16862                                 ; when the user calls get/set media id call before dos establishes the media
16863                                 ; by calling "media_chk",the change line activity of the drive is going to be
16864                                 ; lost.    this routine will check the change line activity and will save the
16865                                 ; history in the flags.
16866                                 ;
16867                                 ; FUNCTION: check the change line error activity
16868                                 ;
```

```
                                ; INPUT :  ES:di -> bds table.
                                ;
                                ; OUTPUT:   flag in bds table will be updated if change line occurs.
                                ;
                                ; SUBROUTINES TO BE CALLED:
                                ;   Set_Changed_DL
                                ;
                                ; ----------------------------------------------------------------------------

                                        ; 24/12/2023 - Retro DOS 5.0
                               ChangeLineChk:
000014C5 268A5504                       mov     dl, [es:di+4]  ; [es:di+BDS.drivenum]
000014C9 08D2                           or      dl, dl         ; Fixed disk?
000014CB 7821                           js      short ChangeLnChkRet ; Yes, skip it.
                                        ; 24/12/2023
000014CD 26F6453F04                     test    byte [es:di+3Fh], 4 ; [es:di+BDS.flags] ; PCDOS 7.1
                                        ; 12/12/2022
                                        ;test   byte [es:di+23h], 4
                                        ;;test  word [es:di+23h], 4 ; [es:di+BDS.flags]
                                                                    ; return_fake_bpb
000014D2 751A                           jnz     short ChangeLnChkRet
000014D4 803E[7700]01                   cmp     byte [fhave96], 1   ; This rom support change line?
000014D9 7513                           jnz     short ChangeLnChkRet
000014DB E8AD07                         call    haschange       ; This drive support change line?
000014DE 740E                           jz      short ChangeLnChkRet ; Do nothing

                                        ; Execute the rom disk interrupt to check changeline activity.

000014E0 B416                           mov     ah, 16h
000014E2 CD13                           int     13h     ; DISK - FLOPPY       DISK - CHANGE OF DISK STATUS (AT,XT2,XT286,CONV,PS)
                                                        ; DL = drive to       check
                                                        ; Return: AH = disk change status
000014E4 7308                           jnb     short ChangeLnChkRet
000014E6 53                             push    bx
000014E7 BB4000                         mov     bx, 40h         ; fchanged
                                                                ; Update flag in BDS for this
                                                                ; physical drive
000014EA E87707                         call    set_changed_dl
000014ED 5B                             pop     bx
                               ChangeLnChkRet:
000014EE C3                             retn

                                ; ----------------------------------------------------------------------------

                                ; 16/10/2022

                                ; ============================================================================
                                ;   GetAccessFlag
                                ; ============================================================================
                                ;
                                ; FUNCTION: get the status of UNFORMATTED_MEDIA bit of flags in bds table
                                ;
                                ; INPUT :
                                ;       ES:di -> bds table
                                ;
                                ; OUTPUT:   a_DiskAccess_Control.dac_access_flag = 0 if disk i/o not allowed.
                                ;                                                  = 1 if disk i/o allowed.
                                ; ============================================================================

                                        ; 24/12/2023 - Retro DOS 5.0

                                        ; 19/10/2022
                               GetAccessFlag:
000014EF C51E[1200]                     lds     bx, [ptrsav]   ; DS:BX points to request header
000014F3 C55F13                         lds     bx, [bx+19]    ; [bx+IOCTL_REQ.GENERICIOCTL_PACKET]
                                        ;mov    al, 0          ; Assume result       is unformatted
                                        ; 10/12/2022
000014F6 28C0                           sub     al, al
                                        ; 24/12/2023
000014F8 26F6454002                     test    byte [es:di+40h], 02h ; (PCDOS 7.1 IBMBIO.COM)
                                        ;test   word ptr es:[di+3Fh], 200h
                                        ; 10/12/2022
                                        ;test   byte [es:di+36], 02h
                                        ;;test  word [es:di+35], 200h ; [es:di+BDS.flags]
                                                                      ; unformatted_media
000014FD 7501                           jnz     short GafDone  ; Done if unformatted
                                        ;inc    al             ; Return true for formatted
                                        ; 24/12/2023
000014FF 40                             inc     ax
                               GafDone:
00001500 884701                         mov     [bx+1], al     ; [bx+A_DISKACCESS_CONTROL.DAC_ACCESS_FLAG]
00001503 C3                             retn
                                ; ----------------------------------------------------------------------------

                                ; 16/10/2022

                                ; ============================================================================
                                ;   SetAccessFlag
                                ; ============================================================================
                                ;
                                ; function: set/reset the UNFORMATTED_MEDIA bit of flags in bds table
                                ;
                                ; input :
                                ;       ES:di -> bds table
                                ;
                                ; output:   unformtted_media bit modified according to the user request
                                ; ============================================================================

                                        ; 24/12/2023 - Retro DOS 5.0

                                        ; 19/10/2022
                               SetAccessFlag:
00001504 C51E[1200]                     lds     bx, [ptrsav]   ; ES:BX points to request header
00001508 C55F13                         lds     bx, [bx+19]    ; [bx+IOCTL_REQ.GENERICIOCTL_PACKET]
                                        ; 24/12/2023
0000150B 26806540FD                     and     byte [es:di+40h], 0FDh ; (PCDOS 7.1 IBMBIO.COM)
                                        ;and    word ptr es:[di+3Fh], 0FDFFh
                                        ; 10/12/2022
                                        ;and    byte [es:di+36], 0FDh
                                        ;;and   word [es:di+35], 0FDFFh ; [es:di+BDS.flags]
                                                                        ; ~unformatted_media
00001510 807F0100                       cmp     byte [bx+1], 0 ; [bx+A_DISKACCESS_CONTROL.DAC_ACCESS_FLAG]
00001514 7505                           jnz     short saf_Done
                                        ; 24/12/2023
00001516 26804D4002                     or      byte [es:di+40h], 02h ; (PCDOS 7.1 IBMBIO.COM)
                                        ;or     word ptr es:[di+3Fh], 200h
                                        ; 15/04/2024
                                        ; 10/12/2022
                                        ;or     byte [es:di+36], 02h
                                        ;;or    word [es:di+35], 200h ; [es:di+BDS.flags]
                                                                      ; unformatted_media
                               saf_Done:
0000151B C3                             retn
                                ; ----------------------------------------------------------------------------
```

```
16993
16994                                   ; 16/10/2022
16995
16996                                   ; ============================================================================
16997                                   ; Ioctl_Support_Query
16998                                   ; ============================================================================
16999                                   ;
17000                                   ; New device command which was added in DOS 5.00 to allow a query of a
17001                                   ; specific GENERIC IOCtl to see if it is supported. Bit 7 in the
17002                                   ; device attributes specifies if this function is supported.
17003                                   ;
17004                                   ; ============================================================================
17005
17006                                           ; 24/12/2023 - Retro DOS 5.0
17007
17008                                           ; 19/10/2022
17009                                   ioctl_support_query:
17010 0000151C 06                                 push    es
17011 0000151D C41E[1200]                         les     bx, [ptrsav]   ; ES:BX Points to request header.
17012 00001521 268B470D                           mov     ax, [es:bx+13] ; [es:bx+IOCTL_REQ.MAJORFUNCTION]
17013                                                                    ; AL == Major, AH == Minor
17014                                           ; 24/12/2023
17015                                           ; 02/09/2023 (PCDOS 7.1)
17016 00001525 3C48                             cmp     al, 48h        ; IOC_NEW_DC (PCDOS 7.1)
17017                                                                    ; new generic ioctl function (FAT32)
17018 00001527 7404                             je      short ioctl_support
17019
17020 00001529 3C08                             cmp     al, 8          ; IOC_DC
17021                                                                    ; See if major code is 8
17022 0000152B 7513                             jne     short nosupport
17023                                   ioctl_support:
17024 0000152D 0E                               push    cs
17025 0000152E 07                               pop     es
17026                                           ; 24/12/2023
17027                                           ; 02/09/2023
17028 0000152F B90E00                           mov     cx, 14          ; (PCDOS 7.1) IOC_DC_TABLE_LEN
17029                                           ;mov    cx, 11         ; IOC_DC_TABLE_LEN
17030                                           ; 10/12/2022
17031 00001532 BF[BD0E]                         mov     di, IOC_DC_Table
17032                                           ;mov    di, 0C60h      ; IOC_DC_Table
17033                                                                  ; at 2C7h:0C60h       = 70h:31D0h
17034 00001535 86C4                             xchg    al, ah         ; Put minor code in AL
17035 00001537 F2AE                             repne scasb            ; Scan for minor code in AL
17036 00001539 7505                             jnz     short nosupport       ; it was not found
17037 0000153B B80001                           mov     ax, 100h
17038                                           ; 10/12/2022
17039                                           ; (jump to ioctlsupexit is not required)
17040                                           ;jmp    short $+2      ; ioctlsupexit
17041                                                                  ; Signal ioctl is supported
17042                                           ;;jmp   short ioctlsupexit
17043                                   ; ---------------------------------------------------------------------------
17044                                   ioctlsupexit:
17045 0000153E 07                               pop     es
17046                                           ; 10/12/2022
17047                                           ; cf = 0
17048                                           ;clc
17049 0000153F C3                               retn
17050                                   ; ---------------------------------------------------------------------------
17051                                   nosupport:
17052 00001540 07                               pop     es
17053 00001541 E991EB                           jmp     bc_cmderr
17054                                   ; ---------------------------------------------------------------------------
17055
17056                                   ; 16/10/2022
17057
17058                                   ; ============================================================================
17059                                   ;   GetMediaSenseStatus
17060                                   ; ============================================================================
17061                                   ;
17062                                   ; FUNCTION: Will return the type of diskette media in the specified DOS
17063                                   ;       diskette drive and whether the media is the default type
17064                                   ;       for that drive. (default type means the max size for that
17065                                   ;       drive)
17066                                   ;
17067                                   ; INPUT :   ES:DI -> BDS table
17068                                   ; OUTPUT:   If carry clear
17069                                   ;       DS:BX -> Updated IOCtlPacket
17070                                   ;
17071                                   ;                   Special Function at offset 0:
17072                                   ;                         0       - Media detected is not default type
17073                                   ;                         1       - Media detected is default type
17074                                   ;
17075                                   ;                   Device Type at offset 1:
17076                                   ;                         2        - 720K 3.5" 80 tracks
17077                                   ;                         7       - 1.44M 3.5" 80 tracks
17078                                   ;                         9       - 2.88M 3.5" 80 tracks
17079                                   ;
17080                                   ; Error Codes returned in AX if carry set:
17081                                   ;
17082                                   ; 8102 - Drive not ready  - No disk is in the drive.
17083                                   ; 8107 - Unknown media type       - Drive doesn't support this function or
17084                                   ;                           the media is really unkown, any error
17085                                   ;                           other than "media not present"
17086                                   ;
17087                                   ; ============================================================================
17088
17089                                           ; 19/10/2022
17090                                   SenseMediaType:
17091 00001544 C51E[1200]                       lds     bx, [ptrsav]   ; DS:BX points to request header.
17092 00001548 C55F13                           lds     bx, [bx+19]    ; bx+IOCTL_REQ.GENERICIOCTL_PACKET]
17093                                           ; 10/10/2022
17094                                           ;mov    word [bx], 0   ; Initialize the 2 packet bytes
17095 0000154B 31D2                             xor     dx, dx
17096 0000154D 8917                             mov     [bx], dx ; 0
17097                                           ;
17098 0000154F 268A5504                         mov     dl, [es:di+4]  ; [es:di+BDS.drivenum]
17099                                                                  ; Get int 13h drive number from      BDS
17100                                           ; 10/12/2022
17101                                           ;xor    dh, dh         ; DX = physical       drive number
17102 00001553 B420                             mov     ah, 20h        ; Get Media Type function
17103                                                                  ; If no carry media type in AL
17104 00001555 CD13                             int     13h            ; DISK - QCACHE       - DISMOUNT
17105 00001557 7216                             jc      short MediaSenseEr ; error code      in AH
17106 00001559 FE07                             inc     byte [bx]      ; Signal media type is default (bit 1)
17107                                   DetermineMediaType:
17108 0000155B FEC8                             dec     al
17109 0000155D 3C02                             cmp     al, 2          ; Chk for 720K ie: (3-1) = 2
17110 0000155F 740A                             jz      short GotMediaType
17111 00001561 0404                             add     al, 4
17112 00001563 3C07                             cmp     al, 7          ; Chk for 1.44M ie: (4-1+4) = 7
17113 00001565 7404                             jz      short GotMediaType
17114 00001567 3C09                             cmp     al, 9          ; Chk for 2.88M       ie: (6-1+4) = 9
17115 00001569 7510                             jnz     short UnknownMediaType ; Just didn't recognize media type
17116                                   GotMediaType:
```

```
17117 0000156B 884701                   mov     [bx+1],al       ; Save the return value
17118                                   ; 10/12/2022
17119                                   ; cf = 0
17120                                   ;clc                    ; Signal success
17121 0000156E C3                       retn
17122                           ; ---------------------------------------------------------------------------
17123
17124                           MediaSenseEr:
17125 0000156F 80FC32                   cmp     ah, 32h         ; See if not default media error
17126 00001572 74E7                     jz      short DetermineMediaType ; Not really an error
17127 00001574 B002                     mov     al, 2           ; Now assume drive not ready
17128 00001576 80FC31                   cmp     ah, 31h         ; See if media was present
17129 00001579 7402                     jz      short SenseErrExit ; Return drive not ready
17130                           UnknownMediaType:
17131 0000157B B007                     mov     al, 7           ; Just don't know the media type
17132                           SenseErrExit:
17133 0000157D B481                     mov     ah, 81h         ; Signal error return
17134 0000157F F9                       stc
17135 00001580 C3                       retn
17136
17137                           ; ---------------------------------------------------------------------------
17138                                   ; 10/12/2022
17139                                   ;db    0
17140                           ; ---------------------------------------------------------------------------
17141
17142                           ;---------------------------------------------------------------------------
17143                           ; PCDOS 7.1 IBMBIO.COM - BIOSCODE:15F2h
17144                           ;---------------------------------------------------------------------------
17145                           ; 26/12/2023 - Retro DOS v5.0 (Modified PCDOS 7.1)
17146
17147                           ; =============== S U B R O U T I N E =======================================
17148
17149                           SetLockState:
17150 00001581 C51E[1200]               lds     bx, [ptrsav]    ; set media lock state
17151 00001585 C55F13                   lds     bx, [bx+13h]    ; [bx+IOCTL_REQ.GENERICIOCTL_PACKET]
17152                                   ;mov    dl, [es:di+4]   ; [es:di+BDS.drivenum]
17153                                   ;call   check_int13h_exts_present
17154                                   ; 26/12/2023
17155 00001588 E82100                   call    check_int13h_exts_p
17156                                   ;mov    al, 3           ; unknown command error
17157 0000158B 721C                     jc      short setlockst_ret
17158 0000158D 8A07                     mov     al, [bx]        ; [bx+A_LOCKSTATE_CONTROL.LOCKSTATE_FUNCTIONS]
17159 0000158F B445                     mov     ah, 45h
17160 00001591 CD13                     int     13h             ; DISK - IBM/MS Extension - LOCK/UNLOCK DRIVE
17161                                   ; (DL - drive, [SI - disk address packet)
17162 00001593 884701                   mov     [bx+1], al      ; 1 = locked, 0 = not locked
17163                                   ; [bx+A_LOCKSTATE_CONTROL.LOCKSTATE_FLAG]
17164
17165                                   ; 26/12/2023
17166 00001596 EB0A                     jmp     short sls_em
17167
17168                           ;       jnc     short setlockst_ret
17169                           ;       mov     al, ah
17170                           ;       call    maperror
17171                           ;setlockst_ret:
17172                           ;       mov     ah, 81h         ; Return this status in case of carry
17173                           ;       retn
17174
17175                           ; =============== S U B R O U T I N E =======================================
17176
17177                           EjectMedia:
17178                                   ;mov    dl, [es:di+4]   ; eject media in drive
17179                                   ;                       ; [es:di+BDS.drivenum]
17180                                   ;call   check_int13h_exts_present
17181                                   ; 26/12/2023
17182 00001598 E81100                   call    check_int13h_exts_p
17183                                   ;mov    al, 3           ; unknown command error
17184 0000159B 720C                     jc      short ejectm_ret
17185 0000159D B80046                   mov     ax, 4600h
17186 000015A0 CD13                     int     13h             ; DISK - IBM/MS Extension - EJECT MEDIA
17187                                   ; (DL - drive)
17188                           sls_em:                         ; 26/12/2023
17189 000015A2 7305                     jnc     short ejectm_ret
17190 000015A4 88E0                     mov     al, ah
17191 000015A6 E800F8                   call    maperror
17192                           setlockst_ret:                  ; 26/12/2023
17193                           ejectm_ret:
17194 000015A9 B481                     mov     ah, 81h         ; Return this status in case of carry
17195 000015AB C3                       retn
17196
17197                           ; =============== S U B R O U T I N E =======================================
17198
17199                                   ; 26/12/2023
17200                           check_int13h_exts_p:
17201 000015AC 268A5504                 mov     dl, [es:di+4]
17202
17203                           check_int13h_exts_present:
17204 000015B0 B441                     mov     ah, 41h
17205 000015B2 53                       push    bx
17206 000015B3 BBAA55                   mov     bx, 55AAh
17207 000015B6 CD13                     int     13h             ; DISK - Check for INT 13h Extensions
17208                                                           ; BX = 55AAh, DL = drive number
17209                                                           ; Return: CF set if not supported
17210                                                           ; AH = extensions version
17211                                                           ; BX = AA55h
17212                                                           ; CX = Interface support bit map
17213 000015B8 81FB55AA                 cmp     bx, 0AA55h
17214 000015BC 5B                       pop     bx
17215 000015BD 7505                     jnz     short exts_notsupported
17216 000015BF F6C102                   test    cl, 2           ; bit 1 - drive locking and ejecting subset
17217 000015C2 7503                     jnz     short exts_supported
17218                           exts_notsupported:
17219                                   ; 26/12/2023
17220 000015C4 B003                     mov     al, 3
17221                                   ;
17222 000015C6 F9                       stc
17223                           exts_supported:
17224 000015C7 C3                       retn
17225
17226                           ; =============== S U B R O U T I N E =======================================
17227
17228                           GetDrvMapInfo:
17229 000015C8 8CD9                     mov     cx, ds          ; get drive map information
17230                                                           ;
17231                                                           ; es:di points to BDS which belongs to
17232                                                           ;       the requested logical/dos drive number
17233                                                           ;
17234                                                           ; Format of parameter block:
17235                                                           ; Offset  Description (Table 01570)
17236                                                           ;  00h    (call) length of this buffer (in bytes)
17237                                                           ;  01h    (ret) number of bytes in parameter block
17238                                                           ;          actually used
17239                                                           ;  02h    (ret) drive flags
17240                                                           ;  03h    (ret) physical drive number
```

```
                                                        ;          00h-7Fh floppy
                                                        ;          80h-FEh hard
                                                        ;          FFh no physical drive
                                                        ;  04h    (ret) bitmap of logical drives associated with
                                                        ;          physical drive
                                                        ;          bit 0 = drive A:, etc.
                                                        ;  08h    (ret) relative block address of partition start
                                                        ;          qword
                                                        ;
                                                        ; Ref: Ralf Brown's Interrupt List, INTERRUP.G
000015CA C51E[1200]                     lds     bx, [ptrsav]
000015CE C55F13                         lds     bx, [bx+13h]   ; [bx+IOCTL_REQ.GENERICIOCTL_PACKET]
000015D1 B80381                         mov     ax, 8103h      ; ah = generic ioctl error code (81h)
                                                               ; al = unknown command error (03h)
000015D4 803F10                         cmp     byte [bx], 10h ; parameter buffer length = 16 bytes
000015D7 7251                           jb      short gdmi_4
000015D9 268A5504                       mov     dl, [es:di+4]  ; [es:di+BDS.drivenum]
000015DD 885703                         mov     [bx+3], dl     ; parameter block - offset 3 - physical drive number
000015E0 C6470110                       mov     byte [bx+1], 10h ; parameter block - actually used length
000015E4 268B4517                       mov     ax, [es:di+17h]        ; [es:di+BDS.hiddensectors]
000015E8 894708                         mov     [bx+8], ax     ; parameter block - offset 8 - partition start LBA
000015EB 268B4519                       mov     ax, [es:di+19h]        ; [es:di+BDS.hiddensectors+2]
000015EF 89470A                         mov     [bx+0Ah], ax   ; parameter block - offset 10
000015F2 31C0                           xor     ax, ax ; 0
000015F4 884702                         mov     [bx+2], al     ; drive flags = 0 (protected mode flags etc.)
000015F7 89470C                         mov     [bx+0Ch], ax   ; high dword of partition start address (LBA) is 0
000015FA 89470E                         mov     [bx+0Eh], ax
000015FD 894704                         mov     [bx+4], ax     ; logical drive bitmap of same physical drive
                                                               ; initialized as 0
00001600 894706                         mov     [bx+6], ax ; 0
00001603 8EC1                           mov     es, cx
                                        ;les    di, dword ptr es:start_bds ; 1st BDS
00001605 26C43E[1901]                   les     di, [es:start_bds]
0000160A B90100                         mov     cx, 1          ; bit 0 (drive A:)
                                        gdmi_1:
0000160D 83FFFF                                 cmp     di, 0FFFFh     ; last BDS ?
00001610 7415                                   jz      short gdmi_3   ; yes
00001612 26385504                               cmp     [es:di+4], dl  ; [es:di+BDS.drivenum], dl
                                                                       ; is it same physical drive ?
00001616 7506                                   jnz     short gdmi_2   ; no
00001618 094F04                                 or      [bx+4], cx     ; set bit for logical drive index of this BDS
                                                                       ; (previously) shifted bit (which is 1/ON) is in ax:cx
0000161B 094706                                 or      [bx+6], ax
                                        gdmi_2:
0000161E D1E1                                   shl     cx, 1          ; shift one left for setting the next drive's bit
00001620 D1D0                                   rcl     ax, 1          ; set high word of the bit select (set) value
00001622 26C43D                                 les     di, [es:di]    ; next BDS
00001625 EBE6                                   jmp     short gdmi_1   ; loop until di = -1 (last BDS sign)
                                        gdmi_3:
00001627 B80001                                 mov     ax, 100h       ; success
                                        gdmi_4:
0000162A C3                                     retn

                                        ;--------------------------------------------------------------------------

                                        ; 16/10/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 IO.SYS)
                                        ; 26/12/2023 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMBIO.COM)

                                        ;--------------------------------------------------------------------------
                                        ; MSINT13.ASM - MSDOS 6.0 - 1991
                                        ;--------------------------------------------------------------------------
                                        ; 16/03/2019 - Retro DOS v4.0

                                        ;   int 2f function 13h allows the user to change the orig13 int_13 vector
                                        ;   after booting. this allows testing and implementation of custom int_13
                                        ;   handlers, without giving up ms-dos error recovery
                                        ;   entry:  ds:dx   == addr. of new int_13 handler
                                        ;           es:bx   == addr. of new int_13 vector used by warm boot (int19)
                                        ;   exit:   orig13  == address of new int_13 handler
                                        ;           ds:dx   == old orig13 value
                                        ;           es:bx   == old old13  value
                                        ;
                                        ; int 2f handler for external block drivers to communicate with the internal
                                        ; block driver in msdisk. the multiplex number chosen is 8. the handler
                                        ; sets up the pointer to the request packet in [ptrsav] and then jumps to
                                        ; dsk_entry, the entry point for all disk requests.
                                        ;
                                        ; on exit from this driver, we will return to the external driver
                                        ; that issued this int 2f, and can then remove the flags from the stack.
                                        ; this scheme allows us to have a small external device driver, and makes
                                        ; the maintainance of the various drivers (driver and msbio) much easier,
                                        ; since we only need to make changes in one place (most of the time).
                                        ;
                                        ;   ax=800h - check for installed handler - reserved
                                        ;   ax=801h - install the bds into the linked list
                                        ;   ax=802h - dos request
                                        ;   ax=803h - return bds table starting pointer in ds:di
                                        ;      (ems device driver hooks int 13h to handle 16kb dma overrun
                                        ;       problem. bds table is going to be used to get head/sector
                                        ;       informations without calling generic ioctl get device parm call.)

                                        ;BIOSSEGMENT equ 70h
                                        DOSBIOSSEG equ 0070h ; 17/10/2022

                                        ;;BIOSCODE:1302h (MSDOS 6.21, IO.SYS)
                                        ;BIOSCODE:16AAh (PCDOS 7.1, IBMBIO.COM) ; 26/12/2023

                                        i2f_handler:                            ; here is 02C7h:1302h =      0070h:3872h
00001629 80FC13                                 cmp     ah, 13h
0000162C 7413                                   jz      short int2f_replace_int13
0000162E 80FC08                                 cmp     ah, 8
00001631 7432                                   jz      short mine

                                        ; Check for WIN386 startup and return the BIOS instance data

00001633 80FC16                                 cmp     ah, 16h        ; MultWin386
00001636 746D                                   jz      short win386call
00001638 80FC4A                                 cmp     ah, 4Ah        ; multMULT
0000163B 7503                                   jnz     short i2f_handler_iret
0000163D E99800                                 jmp     handle_multmult
                                        ; --------------------------------------------------------------------------

                                        i2f_handler_iret:
00001640 CF                                     iret
                                        ; --------------------------------------------------------------------------

                                        int2f_replace_int13:
00001641 FA                                     cli     ; 26/12/2023
00001642 50                                     push    ax      ; free up a register for caller's ds
00001643 8CD8                                   mov     ax, ds  ; then we can use ds: -> Bios_Data
                                                ;;mov   ds, word [cs:0030h] ; 15/10/2022
                                                ;mov    ds, word ptr cs:BIOSDATAWORD ; [cs:0030h]
                                                                ; = [02C7h:0030h] = [0070h:25A0h]
00001645 2E8E1E[3000]                           mov     ds, [cs:BIOSDATAWORD] ; 17/10/2022
```

```
                                        ; 19/10/2022
                                        ;push   word ptr ds:Orig13    ; save old value of old13 and
                                        ;push   word ptr ds:Orig13+2  ; orig13 so that we can
                                        ;push   word ptr ds:Old13     ; return them to caller
                                        ;push   word ptr ds:Old13+2

                                        ; 02/09/2023 (PCDOS 7.1)
                                        ;push   word [Orig13]
0000164C FF36[B600]                     push    word [Orig13+2]
                                        ;push   word [Old13]
00001650 FF36[0801]                     push    word [Old13+2]

                                        ;mov    word ptr ds:Orig13, dx; orig13 := addr. of new int_13
                                        ;mov    word ptr ds:Orig13+2, ax
                                        ;mov    word ptr ds:Old13, bx ; old13 := addr. of new boot_13
                                        ;mov    word ptr ds:Old13+2, es

                                        ;mov    [Orig13], dx
                                        ; 02/09/2023
00001654 8716[B400]                     xchg    dx, [Orig13]
00001658 A3[B600]                       mov     [Orig13+2], ax
                                        ;mov    [Old13], bx
                                        ; 02/09/2023
0000165B 871E[0601]                     xchg    bx, [Old13]
0000165F 8C06[0801]                     mov     [Old13+2], es

00001663 07                             pop     es                      ; es:bx := old old13 vector
                                        ; 02/09/2023
                                        ;pop    bx
00001664 1F                             pop     ds                      ; ds:dx := old orig13 vector
                                        ;pop    dx ; 02/09/2023
00001665 58                             pop     ax
                                i2f_iret:
00001666 CF                             iret
                                ; ---------------------------------------------------------------------------

                                mine:
00001667 3CF8                           cmp     al, 0F8h                ; iret on reserved functions
00001669 73FB                           jnb     short i2f_iret
0000166B 08C0                           or      al, al                  ; a get installed state request?
0000166D 7503                           jnz     short disp_func
0000166F B0FF                           mov     al, 0FFh
                                        ;jmp    short i2f_iret
                                        ; 02/09/2023
00001671 CF                             iret
                                ; ---------------------------------------------------------------------------

                                disp_func:
00001672 3C01                           cmp     al, 1                   ; request for installing bds?
00001674 7418                           jz      short do_subfun_01
00001676 3C03                           cmp     al, 3                   ; get bds vector?
00001678 7423                           jz      short do_get_bds_vector

                                ; set up pointer to request packet

0000167A 1E                             push    ds
0000167B 2E8E1E[3000]                   mov     ds, [cs:BIOSDATAWORD] ; 17/10/2022
                                        ;mov    ds, word ptr cs:BIOSDATAWORD ; [cs:Bios_Data_Word]
                                                                        ; = [0070h:25A0h] = [02C7h:0030h]
                                        ; 19/10/2022
                                        ;mov    word ptr ds:ptrsav, bx
                                        ;mov    word ptr ds:ptrsav+2, es
00001680 891E[1200]                     mov     [ptrsav], bx
00001684 8C06[1400]                     mov     [ptrsav+2], es
00001688 1F                             pop     ds
                                        ;jmp    far ptr i2f_dskentry
                                        ; 07/08/2023 - Retro DOS v4.2 IO.SYS (optimization)
                                        ; PCDOS 7.1 IBMBIO.COM - BIOSCODE:1708h
00001689 EA[5E06]7000                   jmp     DOSBIOSSEG:dsk_entry ; BIOSDATA:dsk_entry
                                        ;; 17/10/2022
                                        ;;jmp   far DOSBIOSSEG:dsk_entry
                                        ;jmp    DOSBIOSSEG:i2f_dskentry ; 70h:i2f_dskentry
                                                                        ; NOTE: jump to a FAR function, not an
                                                                        ;  IRET type function. Callers of
                                                                        ;  this int2f subfunction will have
                                                                        ;  to be careful to do a popf

                                ; ---------------------------------------------------------------------------

                                do_subfun_01:
0000168E 06                             push    es
0000168F 1E                             push    ds
00001690 1E                             push    ds
00001691 07                             pop     es
                                        ; 17/10/2022
00001692 2E8E1E[3000]                   mov     ds, [cs:BIOSDATAWORD]
                                        ;mov    ds, word ptr cs:BIOSDATAWORD ; [cs:Bios_Data_Word]
                                                                        ; point ds: -> Bios_Data
00001697 E8BC03                         call    install_bds
0000169A 1F                             pop     ds
0000169B 07                             pop     es
                                        ;jmp    short i2f_iret
                                        ; 02/09/2023
0000169C CF                             iret
                                ; ---------------------------------------------------------------------------

                                do_get_bds_vector:
                                        ; 17/10/2022
0000169D 2E8E1E[3000]                   mov     ds, [cs:BIOSDATAWORD]
                                        ;mov    ds, word ptr cs:BIOSDATAWORD ; [cs:Bios_Data_Word]
000016A2 C53E[1901]                     lds     di, [start_bds]
                                        ;lds    di, ds:start_bds
                                ;ii2f_iret: ; 10/12/2022
                                        ;jmp    short i2f_iret
                                        ; 02/09/2023
000016A6 CF                             iret
                                ; ---------------------------------------------------------------------------

                                ; 17/10/2022
                                ; 16/10/2022

                                ; WIN386 startup stuff is done here. If starting up we set our WIN386 present
                                ; flag and return instance data. If exiting, we reset the WIN386 present flag
                                ; NOTE: We assume that the BIOS int 2fh is at the bottom of the chain.

                                win386call:
000016A7 1E                             push    ds
000016A8 2E8E1E[3000]                   mov     ds, [cs:BIOSDATAWORD]
                                        ;mov    ds, word ptr cs:BIOSDATAWORD ; [cs:Bios_Data_Word]
                                                                        ; at 2C7h:30h =       70h:25A0h
000016AD 3C05                           cmp     al, 5                   ; Win386_Init
                                                                        ; is itwin386 initializing?
000016AF 7410                           jz      short Win386Init
000016B1 3C06                           cmp     al, 6                   ; Win386_Exit
```

```
                                                        ; is itwin386 exiting?
000016B3 7523                           jnz     short win_iret ; if not, continue int2f chain
                                        ; 12/12/2022
000016B5 F6C201                         test    dl, 1
                                        ;test   dx, 1          ; is itwin386 or win286 dos extender?
000016B8 751E                           jnz     short win_iret ; if not win386, then continue
                                        ;and    ds:IsWin386, 0 ; indicate that       win386 is not present
000016BA 8026[1208]00                   and     byte [IsWin386], 0
000016BF EB17                           jmp     short win_iret
                                ; ---------------------------------------------------------------------------

                                Win386Init:
                                        ; 12/12/2022
000016C1 F6C201                         test    dl, 1
                                        ;test   dx, 1          ; is it win386 or win286 dos extender?
000016C4 7512                           jnz     short win_iret ; if not win386, then continue
                                        ;or     ds:IsWin386, 1 ; Indicate WIN386 present
000016C6 800E[1208]01                   or      byte [IsWin386], 1
                                        ;mov    word ptr ds:SI_Next, bx       ; Hook our structure into chain
                                        ;mov    word ptr ds:SI_Next+2, es
000016CB 891E[E007]                     mov     [SI_Next], bx
000016CF 8C06[E207]                     mov     [SI_Next+2], es
                                        ;mov    bx, offset Win386_SI ; point ES:BX to Win386_SI
000016D3 BB[DE07]                       mov     bx, Win386_SI  ; 19/10/2022
000016D6 1E                             push    ds
000016D7 07                             pop     es
                                win_iret:
000016D8 1F                             pop     ds
                                ii2f_iret:  ; 10/12/2022
                                        ;jmp    short i2f_iret ; return back up the chain
                                        ; 02/09/2023
000016D9 CF                             iret
                                ; ---------------------------------------------------------------------------

                                handle_multmult:
000016DA 3C01                           cmp     al, 1
000016DC 7514                           jnz     short try_2
000016DE 1E                             push    ds
000016DF E84500                         call    HMAPtr         ; get offset of free HMA
                                        ; 10/12/2022
                                        ;xor    bx, bx
                                        ;dec    bx
000016E2 BBFFFF                         mov     bx, 0FFFFh
000016E5 8EC3                           mov     es, bx         ; seg of HMA
000016E7 89FB                           mov     bx, di
000016E9 F7D3                           not     bx
000016EB 09DB                           or      bx, bx
000016ED 7401                           jz      short try_1
000016EF 43                             inc     bx
                                try_1:
000016F0 1F                             pop     ds
                                        ;jmp    short ii2f_iret
                                        ; 02/09/2023
000016F1 CF                             iret
                                ; ---------------------------------------------------------------------------

                                try_2:
000016F2 3C02                           cmp     al, 2          ; multMULTALLOCHMA
000016F4 7530                           jnz     short try_3
000016F6 1E                             push    ds
                                        ; 10/12/2022
                                        ;xor    di, di
                                        ;dec    di
000016F7 BFFFFF                         mov     di, 0FFFFh     ; assume not enough space
000016FA 8EC7                           mov     es, di
000016FC E82800                         call    HMAPtr         ; get offset of free HMA
000016FF 83FFFF                         cmp     di, 0FFFFh
00001702 7421                           jz      short InsuffHMA
00001704 F7DF                           neg     di             ; free space in HMA
00001706 39FB                           cmp     bx, di
00001708 7605                           jbe     short try_4
                                        ; 10/12/2022
                                        ;sub    di, di
                                        ;dec    di
0000170A BFFFFF                         mov     di, 0FFFFh
                                        ;jmp    short InsuffHMA
                                        ; 02/09/2023
0000170D 1F                             pop     ds
0000170E CF                             iret
                                ; ---------------------------------------------------------------------------

                                try_4:
                                        ;mov    di, ds:FreeHMAPtr
0000170F 8B3E[D707]                     mov     di, [FreeHMAPtr]
00001713 83C30F                         add     bx, 15
                                        ;and    bx, 0FFF0h
                                        ; 10/12/2022
00001716 80E3F0                         and     bl, 0F0h
                                        ;add    ds:FreeHMAPtr, bx ; update the free pointer
00001719 011E[D707]                     add     [FreeHMAPtr], bx
0000171D 7506                           jnz     short InsuffHMA
0000171F C706[D707]FFFF                 mov     word [FreeHMAPtr], 0FFFFh ; -1
                                        ;mov    ds:FreeHMAPtr, 0FFFFh
                                                                ; no more HMA if we have wrapped
                                InsuffHMA:
00001725 1F                             pop     ds
                                        ; 10/12/2022
                                try_3:
                                        ;jmp    short ii2f_iret
                                        ; 02/09/2023
00001726 CF                             iret
                                ; ---------------------------------------------------------------------------

                                        ; 10/12/2022
                                ;try_3:
                                        ;jmp    ii2f_iret

                                ; =============== S U B   R O U T I N E =======================================

                                ; 16/10/2022

                                ;---------------------------------------------------------------------------
                                ;
                                ; procedure : HMAPtr
                                ;
                                ;           Gets the offset of the free HMA area ( with respect to
                                ;                                                  seg ffff )
                                ;           If DOS has not moved high, tries to move DOS high.
                                ;           In the course of doing this, it will allocate all the HMA
                                ;           and set the FreeHMAPtr to past the end of the BIOS and
                                ;           DOS code. The call to MoveDOSIntoHMA (which is a pointer)
                                ;           enters the routine in sysinit1 called FTryToMovDOSHi.
                                ;
                                ;   RETURNS : offset of free HMA in DI
```

```
17613                                   ;           BIOS_DATA, seg in DS
17614                                   ;
17615                                   ;--------------------------------------------------------------------------
17616
17617                                           ; 17/10/2022
17618                                   HMAPtr:
17619 00001727 2E8E1E[3000]                     mov     ds, [cs:BIOSDATAWORD]
17620                                           ;mov    ds, word ptr cs:BIOSDATAWORD ; [cs:Bios_Data_Word]
17621 0000172C 8B3E[D707]                       mov     di, [FreeHMAPtr]
17622                                           ;mov    di, ds:FreeHMAPtr
17623 00001730 83FFFF                           cmp     di, 0FFFFh
17624 00001733 750F                             jnz     short HMAPtr_retn
17625 00001735 803E[DD07]00                     cmp     byte [SysinitPresent], 0
17626                                           ;cmp    ds:SysinitPresent, 0
17627 0000173A 7408                             jz      short HMAPtr_retn
17628 0000173C FF1E[D907]                       call    far [MoveDOSIntoHMA]
17629                                           ;call   ds:MoveDOSIntoHMA ; call far [MoveDOSIntoHMA]
17630 00001740 8B3E[D707]                       mov     di, [FreeHMAPtr]
17631                                           ;mov    di, ds:FreeHMAPtr
17632                                   HMAPtr_retn:
17633 00001744 C3                               retn
17634
17635                                   ; =============== S U B   R O U T I N E =======================================
17636
17637                                   ; 16/10/2022
17638
17639                                   ; move a 512 byte sector from ds:si to es:di, do not trash cx
17640                                   ; but go ahead and update direction flag, si, & di
17641
17642                                   move_sector:
17643
17644                                   ; The 80386 microprocessor considers an access to WORD 0FFFFh in
17645                                   ; any segment to be a fault. Theoretically, this could be handled
17646                                   ; by the fault handler and the behavior of an 8086 could be emulated
17647                                   ; by wrapping the high byte to offset 0000h. This would be a lot
17648                                   ; of work and was, indeed, blown off by the Win386 guys. COMPAQ
17649                                   ; also handles the fault incorrectly in their ROM BIOS for real
17650                                   ; mode. Their fault handler was only designed to deal with one
17651                                   ; special case which occurred in a magazine benchmark, but didn't
17652                                   ; handle the general case worth beans.
17653                                   ;
17654                                   ; Simply changing this code to do a byte loop would work okay but
17655                                   ; would involve a general case performance hit. Therefore, we'll
17656                                   ; check for either source or destination offsets being within one
17657                                   ; sector of the end of their segments and only in that case fall
17658                                   ; back to a byte move.
17659
17660 00001745 FC                                   cld
17661 00001746 51                               push    cx
17662 00001747 B90001                           mov     cx, 256
17663 0000174A 81FE00FE                         cmp     si, 0FE00h
17664 0000174E 770A                             ja      short movsec_bytes
17665 00001750 81FF00FE                         cmp     di, 0FE00h
17666 00001754 7704                             ja      short movsec_bytes
17667 00001756 F3A5                             rep movsw
17668 00001758 59                               pop     cx
17669 00001759 C3                               retn
17670                                   ; ---------------------------------------------------------------------------
17671
17672                                   movsec_bytes:
17673 0000175A D1E1                             shl     cx, 1
17674 0000175C F3A4                             rep movsb
17675 0000175E 59                               pop     cx
17676 0000175F C3                               retn
17677
17678                                   ; =============== S U B   R O U T I N E =======================================
17679
17680                                   ; 16/10/2022
17681
17682                                   ; check_wrap is a routine that adjusts the starting sector, starting head
17683                                   ; and starting cylinder for an int 13 request that requests i/o of a lot
17684                                   ; of sectors. it only does this for fixed disks. it is used in the sections
17685                                   ; of code that handle ecc errors and dma errors. it is necessary, because
17686                                   ; ordinarily the rom would take care of wraps around heads and cylinders,
17687                                   ; but we break down a request when we get an ecc or dma error into several
17688                                   ; i/o of one or more sectors. in this case, we may already be beyond the
17689                                   ; number of sectors on a track on the medium, and the request would fail.
17690                                   ;
17691                                   ; input conditions:
17692                                   ;   all registers set up for an int 13 request.
17693                                   ;
17694                                   ; output:
17695                                   ;   dh - contains starting head number for request
17696                                   ;   cx - contains starting sector and cylinder numbers
17697                                   ;   (the above may or may not have been changed, and are 0-based)
17698                                   ;   all other registers preserved.
17699
17700                                           ; 26/12/2023 - Retro DOS 5.0
17701                                   check_wrap:
17702 00001760 50                               push    ax
17703 00001761 53                               push    bx
17704 00001762 06                               push    es
17705 00001763 57                               push    di
17706 00001764 E86C00                           call    find_bds        ; get pointer to bds for drive in dl
17707 00001767 725E                             jb      short no_wrap   ; finished if DOS doesn't use it
17708                                           ; 26/12/2023
17709 00001769 26F6453F01                       test    byte [es:di+3Fh], 1
17710                                           ; 12/12/2022
17711                                           ;test   byte [es:di+23h], 1
17712                                           ;;test  word [es:di+23h], 1 ; [es:di+BDS.flags],fnon_removable
17713 0000176E 7457                             jz      short no_wrap   ; no wrapping for removable media
17714 00001770 268B5D13                         mov     bx, [es:di+13h]         ; [es:di+BDS.secpertrack]
17715 00001774 89C8                             mov     ax, cx
17716 00001776 83E03F                           and     ax, 3Fh         ; extract sector number
17717 00001779 39D8                             cmp     ax, bx          ; are we going to wrap?
17718 0000177B 764A                             jbe     short no_wrap
17719 0000177D F6F3                             div     bl              ; ah=new sector       #, al=# of head wraps
17720
17721                                   ; we need to be careful here. if the new sector # is 0, then we are on the
17722                                   ; last sector on that track.
17723
17724 0000177F 08E4                             or      ah, ah
17725 00001781 7503                             jnz     short not_on_bound
17726                                           ; 18/12/2022
17727 00001783 48                               dec     ax ; *
17728 00001784 88DC                             mov     ah, bl          ; set sector=BDS_BPB.BPB_SECTORSPERTRACK
17729                                                                   ; if on boundary
17730                                           ;dec    al ; *          ; also decrement # of head wraps
17731                                   not_on_bound:
17732 00001786 80E1C0                           and     cl, 0C0h        ; zero out sector #
17733 00001789 08E1                             or      cl, ah          ; or in new sector #
17734 0000178B 30E4                             xor     ah, ah          ; ax = # of head wraps
17735 0000178D 40                               inc     ax
17736 0000178E 00F0                             add     al, dh          ; add in starting head #
```

```
17737 00001790 80D400                           adc     ah, 0           ; catch any carry
17738                                           ; 02/09/2023
17739 00001793 268B5D15                         mov     bx, [es:di+15h]         ; [es:di+BDS.heads]
17740 00001797 39D8                             cmp     ax, bx
17741                                           ;cmp    ax, [es:di+15h]         ; [es:di+BDS.heads]
17742                                                                   ; are we going to wrap around a      head?
17743 00001799 7632                             jbe     short no_wrap_head ; do       not lose new head number!!
17744 0000179B 52                               push    dx              ; preserve drive number and head number
17745 0000179C 31D2                             xor     dx, dx
17746                                           ;mov    bx, [es:di+15h]         ; [es:di+BDS.heads]
17747 0000179E F7F3                             div     bx              ; dx=new head #, ax=# of cylinder wraps
17748
17749                                   ; careful here! if new head # is 0, then we are on the last head.
17750
17751 000017A0 09D2                             or      dx, dx
17752 000017A2 7507                             jnz     short no_head_bound
17753 000017A4 89DA                             mov     dx, bx          ; on boundary. set to BDS_BPB.BPB_HEADS
17754
17755                                   ; if we had some cylinder wraps, we need to reduce them by one!!
17756
17757 000017A6 09C0                             or      ax, ax
17758 000017A8 7401                             jz      short no_head_bound
17759 000017AA 48                               dec     ax              ; reduce number       of cylinder wraps
17760                                   no_head_bound:
17761 000017AB 88D7                             mov     bh, dl          ; bh has new head number
17762 000017AD 5A                               pop     dx              ; restore drive number and head number
17763 000017AE FECF                             dec     bh              ; get it 0-based
17764 000017B0 88FE                             mov     dh, bh          ; set up new head number in dh
17765 000017B2 88CF                             mov     bh, cl
17766 000017B4 80E73F                           and     bh, 3Fh         ; preserve sector number
17767 000017B7 B306                             mov     bl, 6
17768 000017B9 86CB                             xchg    cl, bl
17769 000017BB D2EB                             shr     bl, cl          ; get ms cylinder bits to ls end
17770 000017BD 00C5                             add     ch, al          ; add in cylinder wrap
17771 000017BF 10E3                             adc     bl, ah          ; add in high byte
17772 000017C1 D2E3                             shl     bl, cl          ; move up to ms       end
17773 000017C3 86D9                             xchg    bl, cl          ; restore cylinder bits      into cl
17774 000017C5 08F9                             or      cl, bh          ; or in sector number
17775                                   no_wrap:
17776 000017C7 F8                               clc
17777 000017C8 5F                               pop     di
17778 000017C9 07                               pop     es
17779 000017CA 5B                               pop     bx
17780 000017CB 58                               pop     ax
17781 000017CC C3                               retn
17782                                   ; ---------------------------------------------------------------------------
17783
17784                                   no_wrap_head:
17785 000017CD 88C6                             mov     dh, al          ; do not lose new head number
17786 000017CF FECE                             dec     dh              ; get it 0-based
17787 000017D1 EBF4                             jmp     short no_wrap
17788
17789                                   ; =============== S U B   R O U T I N E =======================================
17790
17791                                   ; 16/10/2022
17792
17793                                   ; this is a special version of the bds lookup code which is
17794                                   ; based on physical drives rather than the usual logical drives
17795                                   ; carry is set if the physical drive in dl is found, es:di -> its bds
17796                                   ; otherwise carry is clear
17797                                   ;
17798                                   ; guaranteed to trash no registers except es:di
17799
17800                                           ; 19/10/2022
17801                                   find_bds:
17802 000017D3 C43E[1901]                       les     di, [start_bds]         ; point es:di to first bds
17803                                   fbds_1:
17804 000017D7 26385504                         cmp     [es:di+4], dl   ; [es:di+BDS.drivenum]
17805 000017DB 7409                             jz      short fdbs_2
17806 000017DD 26C43D                           les     di, [es:di]     ; [es:di+BDS.link]
17807                                                                   ; go to next bds
17808 000017E0 83FFFF                           cmp     di, 0FFFFh
17809 000017E3 75F2                             jnz     short fbds_1
17810 000017E5 F9                               stc
17811                                   fdbs_2:
17812 000017E6 C3                               retn
17813
17814                                   ; =============== S U B   R O U T I N E =======================================
17815
17816                                   ; 16/10/2022
17817                                           ; 17/10/2022
17818                                   doint:
17819                                           ; 10/12/2022
17820 000017E7 8A5608                           mov     dl, [bp+8]      ; [bp+INT13FRAME.olddx]
17821                                                                   ; get physical drive number
17822                                           ; 19/10/2022 - Temporary !
17823                                           ;db     8Ah, 96h, 8, 0 ; mov dl, [bp+8]
17824
17825 000017EA 30E4                             xor     ah, ah
17826 000017EC 08C0                             or      al, al
17827 000017EE 7410                             jz      short dointdone         ; if zero sectors, return ax=0
17828                                           ; 10/12/2022
17829 000017F0 8A6603                           mov     ah, [bp+3]      ; [bp+INT13FRAME.oldax+1]
17830                                                                   ; get request code
17831                                           ;db     8Ah, 0A6h, 3, 0         ; mov ah, [bp+3]
17832 000017F3 FF7610                           push    word [bp+10h]   ; [bp+INT13FRAME.oldf]
17833                                           ;db     0FFh, 0B6h, 10h, 0 ; push word [bp+10h]
17834 000017F6 9D                               popf
17835                                           ;call   far 70h:797h ; MSDOS 6.21 IO.SYS BIOSCODE:14EAh
17836                                           ; 17/10/2022
17837 000017F7 9A[0B07]7000                     call    DOSBIOSSEG:call_orig13
17838                                           ;;call  call_orig13     ; call far 70h:797h
17839                                                                   ; call far KERNEL_SEGMENT:call_orig13
17840 000017FC 9C                               pushf
17841                                           ; 10/12/2022
17842 000017FD 8F4610                           pop     word [bp+10h]   ; [bp+INT13FRAME.oldf]
17843                                           ;db     8Fh, 86h, 10h, 0 ; pop word [bp+10h]
17844                                   dointdone:
17845 00001800 C3                               retn
17846
17847                                   ;---------------------------------------------------------------------------
17848
17849                                   ; 16/10/2022
17850
17851                                   ; this is the true int 13 handler. we parse the request to see if there is
17852                                   ; a dma violation. if so, depending on the function, we:
17853                                   ;   read/write break the request into three pieces and move the middle one
17854                                   ;           into our internal buffer.
17855                                   ;
17856                                   ;   format      copy the format table into the buffer
17857                                   ;   verify      point the transfer address into the buffer
17858                                   ;
17859                                   ; this is the biggest bogosity of all. the ibm controller does not handle
17860                                   ; operations that cross physical 64k boundaries. in these cases, we copy
```

```
                        ; the offending sector into the buffer below and do the i/o from there.

                        ;struc INT13FRAME
                        ;.oldbp: resw
                        ;.oldax: resw
                        ;.oldbx: resw
                        ;.oldcx: resw
                        ;.olddx: resw
                        ;.oldds: resw      ; now we save caller's ds, too
                        ;.olddd: resd
                        ;.oldf:    resw
                        ;end struc

                        ;--------------------------------------------------------------------------

                        ;   entry conditions:
                        ;   ah = function
                        ;   al = number of sectors
                        ;   es:bx = dma address
                        ;   cx = packed track and sector
                        ;   dx = head and drive
                        ;   output conditions:
                        ;   no dma violation.

                        ;   use extreme caution when working with this code. In general,
                        ;     all registers are hot at all times.
                        ;
                        ;   question:  does this code handle cases where dma errors
                        ;     occur during ecc retries, and where ecc errors occur during
                        ;     dma breakdowns???? Hmmmmm.

                        ;--------------------------------------------------------------------------

                        ; ------------------------------------------------------------------------

                                    ; 26/12/2023 - Retro DOS v5.0
                                    ; PCDOS 7.1 IBMBIO.COM - BIOSCODE:1889h
                        dtype_array:
00001801 90004000                   dd 400090h          ; 40h:90h is drive type array addr

                        ; 17/10/2022
                        ;DTYPEARRAY equ dtype_array - DOSBIOSEG_2C7h ; (14F5h for MSDOS 5.0 IO.SYS)
                        ; 09/12/2022
                        DTYPEARRAY equ dtype_array

                        ; ------------------------------------------------------------------------

                        ; stick some special stuff out of mainline

                        ; we know we're doing a format command. if we have changeline
                        ; support, then flag some special changed stuff and set changed
                        ; by format bit for all logical drives using this physical drive

                        format_special_stuff:
00001805 803E[7700]00               cmp     byte [fhave96], 0     ; do we have changeline support?
0000180A 7459                       jz      short format_special_stuff_done ; brif not
0000180C 53                         push    bx
0000180D BB4001                     mov     bx, 140h        ; fchanged_by_format+fchanged
00001810 E85104                     call    set_changed_dl ; indicate that media changed by format
00001813 5B                         pop     bx
00001814 EB4F                       jmp     short format_special_stuff_done
                        ; ------------------------------------------------------------------------

                        ; 16/10/2022

                        ; we know we've got ec35's on the system. Now see if we're doing
                        ; a floppy. If so, create a mask and see if this particular
                        ; drive is an ec35. If so, set dtype_array[drive]=93h

                                    ; 19/10/2022
                        ec35_special_stuff:
00001816 84D2                       test    dl, dl          ; floppy or hard disk?
00001818 7852                       js      short ec35_special_stuff_done ;      if hard drive, we're done
0000181A 50                         push    ax              ; see if this PARTICULAR drive is ec35
0000181B 51                         push    cx
0000181C 88D1                       mov     cl, dl          ; turn drive number into bit map
0000181E B001                       mov     al, 1           ; assume drive 0
00001820 D2E0                       shl     al, cl          ; shift over correct number of times
00001822 8406[A204]                 test    [ec35flag], al ; electrically compatible 3.5 incher?
00001826 59                         pop     cx
00001827 58                         pop     ax
00001828 7442                       jz      short ec35_special_stuff_done
                                                            ; done if this floppy is not an      ec35
0000182A 53                         push    bx              ; free up a far       pointer (es:bx)
0000182B 06                         push    es
                                    ; 17/10/2022
0000182C 2EC41E[0118]               les     bx, [cs:DTYPEARRAY]
                                    ;les    bx, dword ptr cs:DTYPEARRAY ; [cs:dtype_array]
                                                            ; 0070h:3A65h =       2C7h:14F5h
00001831 00D3                       add     bl, dl
00001833 80D700                     adc     bh, 0           ; find entry for this drive
00001836 26C60793                   mov     byte [es:bx], 93h ; establish drive type as:
                                                            ; (360k disk in 360k drive,
                                                            ; no double-stepping, 250 kbs transfer rate)
0000183A 07                         pop     es
0000183B 5B                         pop     bx
0000183C EB2E                       jmp     short ec35_special_stuff_done
                        ; ------------------------------------------------------------------------

                        ; 16/10/2022

                        ; ps2_30 machine has some problem with ah=8h (read drive parm), int 13h.
                        ; this function does not reset the common buses after the execution.
                        ; to solve this problem, when we detect ah=8h, then we will save the result and
                        ; will issue ah=1 (read status) call to reset the buses.

                        ps2_special_stuff:
0000183E 803E[1E00]08               cmp     byte [prevoper], 8 ; (ps2_30)
                                                            ; read driver parm ?
00001843 7407                       jz      short ps2_30_problem
00001845 803E[1E00]15               cmp     byte [prevoper], 15h
                                                            ; apparently function 15h fails, too
0000184A 752D                       jnz     short ps2_special_stuff_done
                        ps2_30_problem:
0000184C 50                         push    ax
0000184D B401                       mov     ah, 1
                                    ; 26/12/2023
                                    ;call  70h:70Bh ; PCDOS 7.1 IBMBIO.COM BIOSCODE:18D7h
                                    ;                       ; call BIOSDATA:call_orig13
                                    ;;call far 70:797h ; MSDOS 6.21 IO.SYS BIOSCODE:1543h
                                    ; 17/10/2022
0000184F 9A[0B07]7000               call    DOSBIOSSEG:call_orig13
                                    ;call   call_orig13     ; call far 70:797h
                                                            ; call far KERNEL_SEGMENT:call_orig13
```

```
17985 00001854 58                              pop     ax
17986 00001855 EB22                            jmp     short ps2_special_stuff_done
17987                                  ; ---------------------------------------------------------------------------
17988
17989                                  ; 17/10/2022
17990                                  ; 16/10/2022
17991
17992                                  ; here is the actual int13 handler
17993
17994                                  i13z:                                   ; 0070h:3ABBh =       02C7h:154Bh
17995
17996                                  ; cas -- inefficient! could push ds and load ds-> Bios_Data before
17997                                  ; vectoring up here from Bios_Data
17998
17999                                          ; 19/10/2022
18000 00001857 1E                              push    ds              ; save caller's ds register first thing
18001                                          ;;mov   ds, word [cs:0030h]
18002                                                                  ; and set up our own ds -> Bios_Data
18003 00001858 2E8E1E[3000]                    mov     ds, [cs:BIOSDATAWORD]
18004                                          ;mov    ds, word ptr cs:BIOSDATAWORD ; [cs:0030h]
18005                                                                  ; = [02C7h:0030h] = [0070h:25A0h]
18006
18007                                  ; let the operation proceed. if there is a dma violation, then we do things
18008
18009 0000185D A3[1E00]                        mov     [prevoper], ax ; save request
18010 00001860 80FC05                          cmp     ah, 5           ; romformat
18011 00001863 74A0                            jz      short format_special_stuff
18012                                                                  ; go do special stuff for format
18013                                  format_special_stuff_done:
18014 00001865 803E[A204]00                    cmp     byte [ec35flag], 0 ; any electrically compat 3.5 inchers?
18015 0000186A 75AA                            jnz     short ec35_special_stuff
18016                                                                  ; go handle it out of line if so
18017                                  ec35_special_stuff_done:
18018                                          ; 26/12/2023
18019                                          ;call   70h:70Bh ; PCDOS 7.1 IBMBIO.COM BIOSCODE:18EDh
18020                                          ;;call  far 70:797h ; MSDOS 6.21 IO.SYS BIOSCODE:1560h
18021 0000186C 9A[0B07]7000                    call    DOSBIOSSEG:call_orig13
18022                                          ;call   call_orig13    ; call far KERNEL_SEGMENT:call_orig13
18023
18024 00001871 9C                              pushf                   ; save result flags
18025
18026 00001872 803E[AF05]FA                    cmp     byte [model_byte], 0FAh ; is this a ps2/30?
18027                                                                  ; mdl_ps2_30
18028 00001877 74C5                            jz      short ps2_special_stuff
18029                                                                  ; exit mainline to address special
18030                                  ps2_special_stuff_done:                 ; ps2/30 problem if so
18031 00001879 9D                              popf
18032 0000187A 7221                            jb      short goterr13 ; error on original orig13 call-thru?
18033                                  ret_from_i13:
18034 0000187C 1F                              pop     ds
18035 0000187D CA0200                          retf    2               ; restore ds & iret w/flags
18036                                  ; ---------------------------------------------------------------------------
18037
18038                                  ; most of our code exits through here. If carry isn't set, then
18039                                  ; just do a simple exit. Else doublecheck that we aren't getting
18040                                  ; a changeline error.
18041
18042                                  i13ret_ck_chglinerr:
18043 00001880 73FA                            jnb     short ret_from_i13 ; done if not an error termination
18044                                  i13_ret_error:
18045 00001882 80FC06                          cmp     ah, 6           ; did i see a change event?
18046 00001885 7513                            jnz     short int13b    ; skip if wrong       error
18047 00001887 08D2                            or      dl, dl          ; is this for the hard disk?
18048 00001889 780F                            js      short int13b    ; yes, ignore
18049 0000188B 803E[7700]00                    cmp     byte [fhave96], 0
18050 00001890 7408                            jz      short int13b    ; just in case ROM returned this
18051                                                                  ; error even though it told us it
18052                                                                  ; never would
18053 00001892 53                              push    bx
18054 00001893 BB4000                          mov     bx, 40h         ; fchanged
18055 00001896 E8CB03                          call    set_changed_dl
18056 00001899 5B                              pop     bx
18057                                  int13b:
18058 0000189A F9                              stc                     ; now return the error
18059 0000189B EBDF                            jmp     short ret_from_i13
18060                                  ; ---------------------------------------------------------------------------
18061
18062                                  ; some kind of error occurred. see if it is dma violation
18063
18064                                  goterr13:
18065 0000189D 80FC09                          cmp     ah, 9           ; dma error?
18066 000018A0 747C                            jz      short gotdmaerr
18067                                  goterr13_xxxx:
18068 000018A2 80FC11                          cmp     ah, 11h         ; ecc error?
18069 000018A5 75DB                            jnz     short i13_ret_error ; other error. just     return back.
18070 000018A7 803E[A905]01                    cmp     byte [media_set_for_format], 1 ; formatting?
18071 000018AC 74D4                            jz      short i13_ret_error
18072
18073 000018AE 803E[1F00]02                    cmp     byte [prevoper+1], 2
18074                                          ;cmp    byte ptr ds:prevoper+1,      2 ; ecc-corrected error
18075                                                                  ; (2 = romread)
18076                                                                  ; ECC correction only applies to reads
18077 000018B3 75CD                            jnz     short i13_ret_error
18078
18079 000018B5 30E4                            xor     ah, ah
18080                                          ;;call  far 70:797h ; MSDOS 6.21 IO.SYS BIOSCODE:15ABh
18081                                          ; 17/10/2022
18082 000018B7 9A[0B07]7000                    call    DOSBIOSSEG:call_orig13
18083                                          ;call   call_orig13    ; call far KERNEL_SEGMENT:call_orig13
18084                                                                  ; call far 70:797h
18085 000018BC A1[1E00]                        mov     ax, [prevoper]
18086 000018BF 30E4                            xor     ah, ah          ; return code =       no error
18087 000018C1 3C01                            cmp     al, 1           ; if request for one sector, assume ok
18088 000018C3 74B7                            jz      short ret_from_i13 ; return with carry clear
18089 000018C5 53                              push    bx
18090 000018C6 51                              push    cx
18091 000018C7 52                              push    dx
18092 000018C8 A2[2000]                        mov     [number_of_sec], al
18093                                  loop_ecc:
18094 000018CB B80102                          mov     ax, 201h        ; read one sector
18095
18096                                  ; we do reads one sector at a time. this ensures that we will eventually
18097                                  ; finish the request since ecc errors on one sector do read in that sector.
18098                                  ;
18099                                  ; we need to put in some "intelligence" into the ecc handler to handle reads
18100                                  ; that attempt to read more sectors than are available on a particular
18101                                  ; track.
18102                                  ;
18103                                  ; we call check_wrap to set up the sector #, head # and cylinder # for
18104                                  ; this request.
18105                                  ;
18106                                  ; at this point, all registers are set up for the call to orig13, except
18107                                  ; that there may be a starting sector number that is bigger than the number
18108                                  ; of sectors on a track.
```

```
                                ;
000018CE E88FFE                 call    check_wrap      ; get correct parameters for int 13
                                ;;call  far 70:797h ; MSDOS 6.21 IO.SYS BIOSCODE:15C5h
                                ; 17/10/2022
000018D1 9A[0B07]7000           call    DOSBIOSSEG:call_orig13
                                ;call   call_orig13    ; call far KERNEL_SEGMENT:call_orig13
000018D6 730C                   jnb     short ok11_op
000018D8 80FC09                 cmp     ah, 9           ; DMA error during ECC read?
000018DB 741B                   jz      short handle_dma_during_ecc
000018DD 80FC11                 cmp     ah, 11h         ; only allow ecc errors
000018E0 7510                   jnz     short ok11_exit_err
                                ; 10/12/2022
                                ; xor ax ax -> ah = 0
                                ;mov    ah, 0           ; ecc error. reset the system again.
000018E2 31C0                   xor     ax, ax          ; clear the error code so that if this
                                                        ; was the last sector, no error      code
                                                        ; will be returned for the corrected
                                                        ; read. (clear carry too.)
                                ok11_op:
000018E4 FE0E[2000]                     dec     byte [number_of_sec]
000018E8 7409                   jz      short ok11_exit       ; all done?
000018EA FEC1                   inc     cl              ; advance sector number
                                                        ; add 200h to address
000018EC FEC7                   inc     bh
000018EE FEC7                   inc     bh
000018F0 EBD9                   jmp     short loop_ecc
                                ; ---------------------------------------------------------------------------

                                ; locate error returns centrally

                                ok11_exit_err:
000018F2 F9                     stc                     ; set carry bit again.
                                ok11_exit:
000018F3 5A                     pop     dx
000018F4 59                     pop     cx
000018F5 5B                     pop     bx
000018F6 EB88                   jmp     short i13ret_ck_chglinerr
                                ; ---------------------------------------------------------------------------

                                ; do the single sector read again, this time into our temporary
                                ; buffer, which is guaranteed not to have a DMA error, then
                                ; move the data to its proper location and proceed

                                handle_dma_during_ecc:
000018F8 06                             push    es
000018F9 53                             push    bx
000018FA BB[5201]                       mov     bx, disksector
000018FD 1E                             push    ds
000018FE 07                             pop     es              ; point es:bx to buffer
000018FF B80102                         mov     ax, 201h        ; read one sector
                                        ;;call  far 70:797h ; MSDOS 6.21 IO.SYS BIOSCODE:15F8h
                                        ; 17/10/2022
00001902 9A[0B07]7000                   call    DOSBIOSSEG:call_orig13
                                        ;call   call_orig13    ; call far KERNEL_SEGMENT:call_orig13
00001907 5B                             pop     bx
00001908 07                             pop     es
00001909 7305                           jnb     short handle_dma_during_ecc_noerr
0000190B 80FC11                         cmp     ah, 11h
0000190E 75E2                           jnz     short ok11_exit_err ; if anything but ecc error, bomb out

                                ; now we're kosher. Copy the data to where it belongs and resume
                                ; the ECC looping code.

                                handle_dma_during_ecc_noerr:
00001910 56                             push    si
00001911 57                             push    di
00001912 89DF                           mov     di, bx
00001914 BE[5201]                       mov     si, disksector
00001917 E82BFE                         call    move_sector
0000191A 5F                             pop     di
0000191B 5E                             pop     si
0000191C EBC6                           jmp     short ok11_op
                                ; ---------------------------------------------------------------------------

                                ; we truly have a dma violation. restore register ax and retry the
                                ; operation as best we can.

                                gotdmaerr:
0000191E A1[1E00]                       mov     ax, [prevoper] ; 19/10/2022
00001921 FB                             sti
00001922 80FC02                         cmp     ah, 2           ; romread
00001925 723B                           jb      short i13_done_dmaerr
                                                        ; just pass dma error thru for
                                                        ; functions we don't handle
00001927 80FC04                         cmp     ah, 4           ; romverify
0000192A 743C                           jz      short intverify
0000192C 80FC05                         cmp     ah, 5           ; romformat
0000192F 7448                           jz      short intformat
00001931 772F                           ja      short i13_done_dmaerr

                                ; we are doing a read/write call. check for dma problems

                                ;   ******** set up stack frame here!!! ********

00001933 52                             push    dx
00001934 51                             push    cx
00001935 53                             push    bx
00001936 50                             push    ax
00001937 55                             push    bp
00001938 89E5                           mov     bp, sp
0000193A 8CC2                           mov     dx, es          ; check for 64k boundary error
                                        ; 26/12/2023
                                        ;add    dx, dx
                                        ;add    dx, dx
                                        ;add    dx, dx
                                        ;add    dx, dx          ; dx = dx*16
0000193C D1E2                           shl     dx, 1
0000193E D1E2                           shl     dx, 1
00001940 D1E2                           shl     dx, 1
00001942 D1E2                           shl     dx, 1           ; segment converted to absolute      address
00001944 01DA                           add     dx, bx          ; combine with offset
00001946 81C2FF01                       add     dx, 511         ; simulate a transfer

                                ; if carry is set, then we are within 512 bytes of the end of the segment.
                                ; we skip the first transfer and perform the remaining buffering and transfer

0000194A 7303                           jnb     short no_skip_first
0000194C E98300                         jmp     bufferx         ; restore dh=head & do buffer
                                ; ---------------------------------------------------------------------------

                                no_skip_first:
0000194F D0EE                           shr     dh, 1           ; dh = number of sectors before      address
00001951 B480                           mov     ah, 128         ; ah = max number of sectors in      segment
00001953 28F4                           sub     ah, dh
```

```
18233
18234                                   ; ah is now the number of sectors that we can successfully write in this
18235                                   ; segment. if this number is above or equal to the requested number, then we
18236                                   ; continue the operation as normal. otherwise, we break it into pieces.
18237                                   ;
18238                                   ; wait a sec. this is goofy. the whole reason we got here in the
18239                                   ; first place is because we got a dma error. so it's impossible
18240                                   ; for the whole block to fit, unless the dma error was returned
18241                                   ; in error.
18242
18243 00001955 38C4                              cmp     ah, al          ; can we fit it       in?
18244 00001957 7236                            jb      short doblock   ; no, perform blocking.
18245
18246                                   ; yes, the request fits. let it happen.
18247
18248 00001959 8A7609                          mov     dh, [bp+9]      ; [bp+INT13FRAME.olddx+1]
18249                                                                   ; set up head number
18250 0000195C E888FE                          call    doint
18251 0000195F E9D900                          jmp     bad13           ; and return from this place
18252                                   ; ---------------------------------------------------------------------------
18253
18254                                   i13_done_dmaerr:
18255 00001962 B409                            mov     ah, 9           ; pass dma error thru to caller
18256 00001964 F9                              stc
18257 00001965 E914FF                          jmp     ret_from_i13    ; return with error,
18258                                                                   ; we know it's not a changeline error
18259                                   ; ---------------------------------------------------------------------------
18260
18261                                   ; verify the given sectors. place the buffer pointer into our space.
18262
18263                                   intverify:
18264 00001968 06                              push    es              ; save caller's dma address
18265 00001969 53                              push    bx
18266 0000196A 1E                              push    ds              ; es:bx -> Bios_Data:disksector
18267 0000196B 07                              pop     es
18268                                   dosimple:
18269 0000196C BB[5201]                        mov     bx, disksector
18270                                                                   ; do the i/o from Bios_Data:disksector
18271                                           ;;call  far 70:797h ; MSDOS 6.21 IO.SYS BIOSCODE:1665h
18272                                           ; 17/10/2022
18273 0000196F 9A[0B07]7000                    call    DOSBIOSSEG:call_orig13
18274                                           ;call   call_orig13    ; call far KERNEL_SEGMENT:call_orig13
18275 00001974 5B                              pop     bx
18276 00001975 07                              pop     es
18277 00001976 E907FF                          jmp     i13ret_ck_chglinerr
18278                                   ; ---------------------------------------------------------------------------
18279
18280                                   ; format operation. copy the parameter table into Bios_Data:disksector
18281
18282                                   intformat:
18283 00001979 06                              push    es
18284 0000197A 53                              push    bx
18285 0000197B 56                              push    si
18286 0000197C 57                              push    di
18287 0000197D 1E                              push    ds
18288
18289                                   ; point ds to the caller's dma buffer, es to Bios_Data
18290                                   ; in other words, swap (ds, es)
18291
18292 0000197E 06                              push    es
18293 0000197F 1E                              push    ds
18294 00001980 07                              pop     es
18295 00001981 1F                              pop     ds
18296 00001982 89DE                            mov     si, bx
18297 00001984 BF[5201]                        mov     di, disksector
18298 00001987 E8BBFD                          call    move_sector     ; user's data into Bios_Data:disksector
18299 0000198A 1F                              pop     ds
18300 0000198B 5F                              pop     di
18301 0000198C 5E                              pop     si              ; do the i/o from
18302 0000198D EBDD                            jmp     short dosimple ; Bios_Data:disksector
18303                                   ; ---------------------------------------------------------------------------
18304
18305                                   ; we can't fit the request into the entire block. perform the operation on
18306                                   ; the first block.
18307                                   ;
18308                                   ; doblock is modified to correctly handle multi-sector disk i/o.
18309                                   ; old doblock had added the number of sectors i/oed (ah in old doblock) after
18310                                   ; the doint call to cl. observing only the lower 6 bits of cl(=max. 64) can
18311                                   ; represent a starting sector, if ah was big, then cl would be clobbered.
18312                                   ; by the way, we still are going to use cl for this purpose since checkwrap
18313                                   ; routine will use it as an input. to prevent cl from being clobbered, a
18314                                   ; safe number of sectors should be calculated like "63 - # of sectors/track".
18315                                   ; doblock will handle the first block of requested sectors within the
18316                                   ; boundary of this safe value.
18317
18318                                           ; 26/12/2023 - Retro DOS v5.0
18319                                   doblock:
18320
18321                                   ; try to get the # of sectors/track from bds via rom drive number.
18322                                   ; for any mini disks installed, here we have to pray that they have the
18323                                   ; same # of sector/track as the main dos partition disk drive.
18324
18325 0000198F 8B5608                          mov     dx, [bp+8]      ; [bp+INT13FRAME.olddx]
18326                                                                   ; get head #, drive #
18327 00001992 51                              push    cx
18328 00001993 06                              push    es
18329 00001994 57                              push    di              ; ah - # of sectors before dma boundary
18330                                                                   ; al - requested # of sectors for i/o.
18331 00001995 E83BFE                          call    find_bds
18332 00001998 268B4D13                        mov     cx, [es:di+13h]       ; [es:di+BDS.secpertrack]
18333                                          ; 26/12/2023
18334 0000199C 26F6453F01                      test    byte [es:di+3Fh], 1
18335                                          ; 12/12/2022
18336                                          ;test   byte [es:di+23h], 1
18337                                          ;;test  word [es:di+23h], 1 ; [es:di+BDS.flags],fnon_removable
18338 000019A1 5F                              pop     di
18339 000019A2 07                              pop     es
18340 000019A3 88E0                            mov     al, ah          ; set al=ah for       floppies
18341 000019A5 7404                            jz      short doblockflop ; they are track by track operation
18342 000019A7 B43F                            mov     ah, 63          ; ah = 63-secpt       (# safe sectors??)
18343 000019A9 28CC                            sub     ah, cl          ; al - # of sectors before dma boundary
18344                                   doblockflop:
18345 000019AB 59                              pop     cx
18346                                   doblockcontinue:
18347 000019AC 38C4                            cmp     ah, al          ; if safe_# >= #_of_sectors_to_go_before dma,
18348 000019AE 7305                            jnb     short doblocklast ; then #_of_sectors_to_go as it is for doint.
18349 000019B0 50                              push    ax
18350 000019B1 88E0                            mov     al, ah          ; otherwise, set al to ah to operate.
18351 000019B3 EB03                            jmp     short doblockdoint
18352                                   ; ---------------------------------------------------------------------------
18353
18354                                   doblocklast:
18355 000019B5 88C4                            mov     ah, al
18356 000019B7 50                              push    ax
```

```
18357                                   doblockdoint:                           ; let ah = al =       # of sectors for this shot
18358 000019B8 E82CFE                             call    doint
18359 000019BB 727E                               jb      short bad13     ; something happened, bye!
18360 000019BD 58                                 pop     ax
18361 000019BE 286602                             sub     [bp+2],ah       ; sub [bp+INT13FRAME.oldax], ah
18362                                                                     ; decrement by the successful operation
18363 000019C1 00E1                               add     cl, ah          ; advance sector #. safety gauranteed.
18364 000019C3 00E7                               add     bh, ah          ; advance dma addres
18365 000019C5 00E7                               add     bh, ah          ; twice for 512 byte sectors
18366 000019C7 38C4                               cmp     ah, al          ; check the previous value
18367 000019C9 740A                               jz      short buffer    ; if #_of_sectors_to_go       < safe_#,
18368                                                                     ; then we are done already.
18369 000019CB 28E0                               sub     al, ah          ; otherwise,
18370                                                                     ; #_sector_to_go = #_of_sector_to_go - safe_#
18371 000019CD E890FD                             call    check_wrap      ; get new cx, dh for the next operation.
18372 000019D0 EBDA                               jmp     short doblockcontinue ;       handles next sectors left.
18373                                   ; ---------------------------------------------------------------------------
18374
18375                                   bufferx:
18376 000019D2 8A7609                             mov     dh, [bp+9]      ; [bp+INT13FRAME.olddx+1]
18377                                                                     ; set up head number
18378                                   buffer:
18379 000019D5 53                                 push    bx
18380 000019D6 8A6603                             mov     ah, [bp+3]      ; [bp+INT13FRAME.oldax+1]
18381 000019D9 80FC03                             cmp     ah, 3           ; romwrite
18382 000019DC 7525                               jnz     short doread    ;
18383
18384                                   ; copy the offending sector into local buffer
18385
18386 000019DE 06                                 push    es
18387 000019DF 1E                                 push    ds
18388 000019E0 56                                 push    si
18389 000019E1 57                                 push    di
18390 000019E2 1E                                 push    ds              ; exchange segment registers
18391 000019E3 06                                 push    es
18392 000019E4 1F                                 pop     ds
18393 000019E5 07                                 pop     es
18394 000019E6 BF[5201]                           mov     di, disksector ; where to move
18395 000019E9 57                                 push    di              ; save it
18396 000019EA 89DE                               mov     si, bx          ; source
18397 000019EC E856FD                             call    move_sector     ; move sector into local buffer
18398 000019EF 5B                                 pop     bx              ; new transfer address
18399                                                                     ; (es:bx = Bios_Data:diskbuffer)
18400 000019F0 5F                                 pop     di              ; restore caller's di & si
18401 000019F1 5E                                 pop     si
18402 000019F2 1F                                 pop     ds              ; restore Bios_Data
18403
18404                                   ; see if we are wrapping around a track or head
18405
18406 000019F3 B001                                mov     al, 1           ; [bp+INT13FRAME.olddx]
18407                                                                     ; get drive number
18408 000019F5 8A5608                             mov     dl, [bp+8]
18409 000019F8 E865FD                             call    check_wrap      ; sets up registers if wrap-around
18410                                                                     ;
18411                                                                     ; ah is function
18412                                                                     ; al is 1 for single sector transfer
18413                                                                     ; es:bx is local transfer addres
18414                                                                     ; cx is track/sector number
18415                                                                     ; dx is head/drive number
18416                                                                     ; si,di unchanged
18417 000019FB E8E9FD                             call    doint
18418 000019FE 07                                 pop     es              ; restore caller's dma segment
18419 000019FF 723A                               jb      short bad13     ; go clean up
18420 00001A01 EB22                               jmp     short dotail
18421                                   ; ---------------------------------------------------------------------------
18422
18423                                   ; reading a sector. do int first, then move things around
18424
18425                                   doread:
18426 00001A03 06                                 push    es
18427 00001A04 53                                 push    bx
18428 00001A05 1E                                 push    ds              ; es = Bios_Code
18429 00001A06 07                                 pop     es
18430 00001A07 BB[5201]                           mov     bx, disksector
18431 00001A0A B001                               mov     al, 1
18432 00001A0C 8A5608                             mov     dl, [bp+8]      ; [bp+INT13FRAME.olddx]
18433                                                                     ; get drive number
18434 00001A0F E84EFD                             call    check_wrap      ;
18435                                                                     ; ah = function
18436                                                                     ; al = 1 for single sector
18437                                                                     ; es:bx points to local       buffer
18438                                                                     ; cx, dx are track/sector, head/drive
18439 00001A12 E8D2FD                             call    doint
18440 00001A15 5B                                 pop     bx
18441 00001A16 07                                 pop     es
18442 00001A17 7222                               jb      short bad13
18443 00001A19 56                                 push    si
18444 00001A1A 57                                 push    di
18445 00001A1B 89DF                               mov     di, bx
18446 00001A1D BE[5201]                           mov     si, disksector
18447 00001A20 E822FD                             call    move_sector
18448 00001A23 5F                                 pop     di
18449 00001A24 5E                                 pop     si
18450
18451                                   ; note the fact that we've done 1 more sector
18452
18453                                   dotail:
18454 00001A25 5B                                 pop     bx              ; retrieve new dma area
18455 00001A26 80C702                             add     bh, 2           ; advance over sector
18456 00001A29 41                                 inc     cx
18457 00001A2A 8A4602                             mov     al, [bp+2]      ; [bp+INT13FRAME.oldax]
18458 00001A2D F8                                 clc
18459 00001A2E FEC8                               dec     al
18460 00001A30 7409                               jz      short bad13     ; no more i/o
18461
18462                                   ; see if we wrap around a track or head boundary with starting sector
18463                                   ; we already have the correct head number to pass to check_wrap
18464
18465 00001A32 8A5608                             mov     dl, [bp+8]      ; [bp+INT13FRAME.olddx]
18466 00001A35 E828FD                             call    check_wrap
18467 00001A38 E8ACFD                             call    doint
18468
18469                                   ; we are done. ax has the final code; we throw away what we got before
18470
18471                                   ; M046  -- okay gang. Now we've either terminated our DMA loop,
18472                                   ;       or we've finished. If carry is set now, our only
18473                                   ;       hope for salvation is that it was a read operation
18474                                   ;       and the error code is ECC error. In that case, we'll
18475                                   ;       just pop the registers and go do the old ECC thing.
18476                                   ;       When the DMA error that got us here in the first
18477                                   ;       place occurs, it'll handle it.
18478
18479                                   bad13:
18480 00001A3B 89EC                               mov     sp, bp
```

```
18481 00001A3D 5D                          pop     bp
18482 00001A3E 5B                          pop     bx
18483 00001A3F 5B                          pop     bx
18484 00001A40 59                          pop     cx
18485 00001A41 5A                          pop     dx
18486 00001A42 7203                        jb      short xgoterr13_xxxx ; go handle ECC errors
18487 00001A44 E935FE                      jmp     ret_from_i13   ; non-error exit
18488                                  ; ---------------------------------------------------------------------------
18489
18490                                  xgoterr13_xxxx:
18491 00001A47 E958FE                      jmp     goterr13_xxxx
18492
18493                                  ; ---------------------------------------------------------------------------
18494                                          ; 10/12/2022
18495                                          ;db     0
18496                                  ; ---------------------------------------------------------------------------
18497
18498                                  ;Bios_Code ends
18499
18500                                  ; 16/10/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 IO.SYS)
18501
18502                                  ;--------------------------------------------------------------------------
18503                                  ; MSBIO2.ASM - MSDOS 6.0 - 1991
18504                                  ;--------------------------------------------------------------------------
18505                                  ; 17/03/2019 - Retro DOS v4.0
18506
18507                                          ; 19/10/2022
18508                                  dsk_init:                       ; 2C7h:1742h = 70h:3CB2h
18509 00001A4A 8A26[7500]                      mov     ah, [drvmax]
18510 00001A4E BF[3C05]                        mov     di, dskdrvs
18511 00001A51 1E                              push    ds              ; pass result in es:di
18512 00001A52 07                              pop     es
18513 00001A53 E934EC                          jmp     SetPtrSav
18514
18515                                  ; =============== S U B   R O U T I N E =======================================
18516
18517                                  ;--------------------------------------------------------------------------
18518                                  ; install_bds installs a bds at location es:di into the current linked list of
18519                                  ; bds maintained by this device driver. it places the bds at the end of the
18520                                  ; list. Trashes (at least) ax, bx, di, si
18521                                  ;--------------------------------------------------------------------------
18522
18523                                          ; 26/12/2023 - Retro DOS v5.0
18524                                          ; PCDOS 7.1 IBMBIO.COM - BIOSCODE:1AE0h
18525                                  install_bds:
18526 00001A56 1E                              push    ds              ; save Bios_Data segment
18527 00001A57 BE[1901]                        mov     si, start_bds   ; beginning of chain
18528
18529                                          ; ds:si now points to link to first bds
18530                                          ; assume bds list is non-empty
18531                                  loop_next_bds:
18532 00001A5A C534                            lds     si, [si]        ; [si+BDS.link]
18533                                                                  ; fetch next bds
18534 00001A5C 268A4504                        mov     al, [es:di+4]   ; [es:di+BDS.drivenum]
18535 00001A60 384404                          cmp     [si+4], al      ; does this one       share a physical
18536                                                                  ; drive with new one?
18537 00001A63 7518                            jnz     short next_bds
18538 00001A65 B310                            mov     bl, 10h         ; fi_am_mult
18539                                          ; 26/12/2023
18540 00001A67 26085D3F                        or      [es:di+3Fh], bl
18541                                          ;or     [es:di+23h], bl       ; [es:di+BDS.flags]
18542                                                                  ; set both of them to i_am_mult      if so
18543 00001A6B 085C3F                         or      [si+3Fh], bl
18544                                          ;or     [si+23h], bl    ; [si+BDS.flags]
18545 00001A6E 2680653FDF                      and     byte [es:di+3Fh], 0DFh
18546                                          ;and    byte [es:di+23h], 0DFh ; [es:di+BDS.flags],~fi_own_physical
18547                                                                  ; we don't own it
18548 00001A73 8A5C3F                          mov     bl, [si+3Fh]
18549                                          ;mov    bl, [si+23h]    ; [si+BDS.flags]
18550                                                                  ; determine if changeline available
18551 00001A76 80E302                          and     bl, 2           ; fchangeline
18552 00001A79 26085D3F                        or      [es:di+3Fh], bl
18553                                          ;or     [es:di+23h], bl       ; [es:di+BDS.flags]
18554                                  next_bds:
18555                                          ; 02/09/2023 (PCDOS 7.1)
18556 00001A7D B8FFFF                          mov     ax, 0FFFFh      ; -1
18557 00001A80 3904                            cmp     [si], ax        ; [si+BDS.link],-1
18558                                          ;cmp    word [si], 0FFFFh ; [si+BDS.link],-1
18559                                                                  ; are we at end       of list?
18560 00001A82 75D6                            jnz     short loop_next_bds
18561 00001A84 8C4402                          mov     [si+2], es      ; [si+BDS.link+2],es
18562                                                                  ; install bds
18563 00001A87 893C                            mov     [si], di
18564 00001A89 268905                          mov     [es:di], ax     ; [es:di+BDS.link],-1
18565                                          ;mov    word [es:di], 0FFFFh ; [es:di+BDS.link],-1
18566                                                                  ; set next pointer to null
18567 00001A8C 1F                              pop     ds
18568
18569                                  ; 01/07/2023 - Retro DOS v4.2 (MSDOS 6.21 IO.SYS - BIOSCODE:1785h)
18570                                  ; 16/10/2022 (MSDOS 6.0 Code)
18571
18572                                  ; **** If the new drive has a higher EOT value, we must alter the
18573                                  ;      'eot' variable appropriately.
18574
18575                                              ; 26/12/2023
18576 00001A8D 268A4550                        mov     al, [es:di+50h]       ; [es:di+BDS.rsecpertrack]
18577                                          ; 01/06/2019
18578                                          ;mov    al,[es:di+52]
18579                                          ; 22/07/2023
18580                                          ;mov    al,[es:di+BDS.rsecpertrack]
18581 00001A91 3A06[2C01]                      cmp     al,[eot]
18582 00001A95 7603                            jbe     short _eot_ok
18583 00001A97 A2[2C01]                        mov     [eot],al
18584                                  _eot_ok:
18585 00001A9A C3                              retn
18586
18587                                  ; ---------------------------------------------------------------------------
18588
18589                                  ; 17/10/2022
18590                                  ;DRVLET     equ drvlet - DOSBIOSEG_2C7h
18591                                  ;SNGMSG     equ sngmsg - DOSBIOSEG_2C7h
18592                                  ; 09/12/2022
18593                                  DRVLET equ drvlet
18594                                  SNGMSG equ sngmsg
18595
18596                                  ; 16/10/2022
18597
18598                                  ;--------------------------------------------------------------------------
18599                                  ;  ask to swap the disk in drive a:
18600                                  ;   es:di -> bds
18601                                  ;   ds -> Bios_Data
18602                                  ;--------------------------------------------------------------------------
18603
18604                                          ; 26/12/2023 - Retro DOS v5.0
```

```
18605
18606                                           ; 19/10/2022
18607 00001A9B F606[1208]01             swpdsk:         test    byte [IsWin386], 1
18608                                           ;test   ds:IsWin386, 1 ; Is win386 present?
18609 00001AA0 7405                             jz      short no_win386       ; no, skip SetFocus
18610
18611                                           ; set focus to the correct VM
18612                                           ;call   far ptr 70h:813h ; PCDOS 7.1 IBMBIO.COM BIOSCODE:1B2Ch
18613                                           ;;call  far 70h:8D1h   ; MSDOS 6.21 IO.SYS BIOSCODE:179Ah
18614                                           ; 17/10/2022
18615 00001AA2 9A[1308]7000                     call    DOSBIOSSEG:V86_Crit_SetFocus ; BIOSDATA:V86_Crit_SetFocus
18616                                           ;call   far ptr V86_Crit_SetFocus ; call far 70h:8D1h
18617                                                                   ; call far KERNEL_SEGMENT:V86_Crit_SetFocus
18618                                   no_win386:
18619 00001AA7 51                               push    cx
18620 00001AA8 52                               push    dx
18621 00001AA9 268A5505                         mov     dl, [es:di+5]  ; [es:di+BDS.drivelet]
18622                                                                   ; get the drive       letter
18623
18624                                   ; WARNING : next two instructions assume that if the new disk is for drive B
18625                                   ;           then existing dsk is drive A & vice versa
18626
18627 00001AAD 88D6                             mov     dh, dl
18628 00001AAF 80F601                           xor     dh, 1
18629 00001AB2 29C9                             sub     cx, cx          ; nobody has handled swap disk
18630 00001AB4 B8004A                           mov     ax, 4A00h       ; multMULT<<8)|multMULTSWPDSK
18631                                                                   ; broadcast code for swap disk
18632                                                                   ; Broadcast it
18633 00001AB7 CD2F                             int     2Fh
18634 00001AB9 41                               inc     cx              ; cx == -1 ?
18635 00001ABA 741E                             jz      short swpdsk9   ; somebody has handled it
18636
18637                                   ; using a different drive in a one drive system so request the user change disks
18638
18639 00001ABC 80C241                           add     dl, 'A'
18640                                           ; 17/10/2022
18641 00001ABF 2E8816[F91A]                     mov     [cs:DRVLET], dl       ; "A: and press any key when ready\r\n\n"
18642                                           ; 16/10/2022
18643                                           ;;mov   byte [cs:drvlet], dl
18644                                           ;mov    byte ptr cs:17E4h, dl ; [cs:drvlet]
18645                                                                   ; 0070h:3D54h =       2C7h:17E4h
18646 00001AC4 BE[DD1A]                        mov     si, SNGMSG      ; "\r\nInsert diskette for drive "
18647                                           ;mov    si, 17C8h       ; sngmsg
18648                                                                   ; 0070h:3D38h =       2C7h:17C8h
18649 00001AC7 53                               push    bx
18650 00001AC8 2E                               cs
18651 00001AC9 AC                               lodsb                   ; get the next character of the message
18652                                           ;lods   byte ptr cs:[si]
18653                                   wrmsg_loop:
18654 00001ACA CD29                             int     29h             ; DOS 2+ internal - FAST PUTCHAR
18655                                                                   ; AL = character to display
18656 00001ACC 2E                               cs
18657 00001ACD AC                               lodsb
18658                                           ;lods   byte ptr cs:[si] ; cs lodsb
18659                                                                   ; get the next character of the      message
18660 00001ACE 08C0                             or      al, al
18661 00001AD0 75F8                             jnz     short wrmsg_loop
18662 00001AD2 E833E7                           call    con_flush       ; flush out keyboard queue
18663                                                                   ; call rom-bios
18664 00001AD5 30E4                             xor     ah, ah
18665 00001AD7 CD16                             int     16h             ; KEYBOARD - READ CHAR FROM BUFFER, WAIT IF EMPTY
18666                                                                   ; Return: AH = scan code, AL = character
18667 00001AD9 5B                               pop     bx
18668                                   swpdsk9:
18669 00001ADA 5A                               pop     dx
18670 00001ADB 59                               pop     cx
18671 00001ADC C3                               retn
18672
18673                                   ; ---------------------------------------------------------------------------
18674
18675                                   ; 16/10/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 IO.SYS)
18676
18677                                   ;--------------------------------------------------------
18678                                   ; include msbio.cl2 (MSDOS 6.0, 1991)
18679                                   ;--------------------------------------------------------
18680                                   ; (MSDOS 6.21 IO.SYS BIOSCODE:17D5h)
18681                                   ;--------------------------------------------------------
18682                                   ; 17/03/2019 - Retro DOS v4.0
18683                                   ; 26/12/2023 - Retro DOS v5.0
18684
18685                                           ; MSDOS 5.0 IO.SYS offset 0070h:3D38h or 02C7h:17C8h
18686                                           ; PCDOS 7.1 IBMBIO.COM - BIOSCODE:1B67h
18687 00001ADD 0D0A                     sngmsg:         db 0Dh,0Ah
18688 00001ADF 496E73657274206469-               db 'Insert diskette for drive '
18688 00001AE8 736B6574746520666F-
18688 00001AF1 7220647269766520
18689
18690                                           ; MSDOS 5.0 IO.SYS offset 0070h:3D54h or 02C7h:17E4h
18691                                           ; PCDOS 7.1 IBMBIO.COM - BIOSCODE:1B83h
18692 00001AF9 413A20616E64207072-      drvlet:         db 'A: and press any key when ready',0Dh,0Ah
18692 00001B02 65737320616E79206B-
18692 00001B0B 6579207768656E2072-
18692 00001B14 656164790D0A
18693 00001B1A 0A00                             db 0Ah,0
18694
18695                                   ; =============== S U B   R O U TI N E =======================================
18696
18697                                   ;---------------------------------------------------------------------------
18698                                   ; input : es:di points to current bds for drive.
18699                                   ; return : zero set if no open files
18700                                   ;      zero reset if open files
18701                                   ;---------------------------------------------------------------------------
18702
18703                                           ; 26/12/2023 - Retro DOS v5.0
18704                                   chkopcnt:
18705 00001B1C 26837D3C00                       cmp     word [es:di+3Ch], 0
18706                                           ;cmp    word [es:di+20h], 0 ; [es:di+BDS.opcnt]
18707 00001B21 C3                               retn
18708
18709                                   ; =============== S U B   R O U TI N E =======================================
18710
18711                                   ;---------------------------------------------------------------------------
18712                                   ; at media check time, we need to really get down and check what the change is.
18713                                   ; this is guaranteed to be expensive.
18714                                   ;
18715                                   ;   es:di -> bds, ds -> Bios_Data
18716                                   ;---------------------------------------------------------------------------
18717
18718                                           ; 26/12/2023 - Retro DOS v5.0
18719                                           ; PCDOS 7.1 IBMBIO.COM - BIOSCODE:1BA6h
18720                                   mediacheck:
18721 00001B22 E852EE                           call    checksingle    ; make sure correct disk is in place
18722 00001B25 31F6                             xor     si, si
18723 00001B27 E86101                           call    haschange
```

```
18724 00001B2A 742F                             jz      short mediaret
18725                                           ; 26/12/2023
18726                                           ;test   byte [es:di+3Fh], 40h ; [es:di+BDS.flags], fchanged ; 40h
18727 00001B2C E85001                           call    checkromchange
18728 00001B2F 752B                             jnz     short mediadovolid
18729 00001B31 50                               push    ax
18730 00001B32 52                               push    dx
18731 00001B33 268A5504                         mov     dl, [es:di+4]  ; [es:di+BDS.drivenum]
18732                                                                  ; set logical drive number
18733 00001B37 B416                             mov     ah, 16h
18734 00001B39 CD13                             int     13h            ; DISK - FLOPPY       DISK - CHANGE OF DISK STATUS (AT,XT2,XT286,CONV,PS)
18735                                                                  ; DL = drive to       check
18736                                                                  ; Return: AH = disk change status
18737 00001B3B 5A                               pop     dx
18738 00001B3C 58                               pop     ax
18739 00001B3D 721D                             jb      short mediadovolid
18740 00001B3F BE0100                           mov     si, 1          ; signal no change
18741
18742                                   ; there are some drives with changeline that "lose" the changeline indication
18743                                   ; if a different drive is accessed after the current one. in order to avoid
18744                                   ; missing a media change, we return an "i don't know" to dos if the changeline
18745                                   ; is not active and we are accessing a different drive from the last one.
18746                                   ; if we are accessing the same drive, then we can safely rely on the changeline
18747                                   ; status.
18748                                             ; 19/10/2022
18749 00001B42 8A1E[1E01]                         mov     bl, [tim_drv]  ; get last drive accessed
18750 00001B46 26385D04                           cmp     [es:di+4], bl  ; [es:di+BDS.drivenum]
18751                                                                    ; (If the last drive accessed is not current drive
18752                                                                    ; media change status may be incorrect.      So,
18753                                                                    ; "I don't now" will be returned even if it is indicated
18754                                                                    ; as media is not changed.)
18755 00001B4A 740F                               jz      short mediaret ; (same drive,
18756                                                                    ; media changeline indication is reliable)
18757
18758                                   ; do the 2 second twiddle. if time >= 2 seconds, do a volid check.
18759                                   ; otherwise return "i don't know" (strictly speaking, we should return a
18760                                   ; "not changed" here since the 2 second test said no change.)
18761
18762 00001B4C 50                                 push    ax
18763 00001B4D 51                                 push    cx
18764 00001B4E 52                                 push    dx
18765 00001B4F E8D8EA                             call    Check_Time_Of_Access
18766 00001B52 5A                                 pop     dx
18767 00001B53 59                                 pop     cx
18768 00001B54 58                                 pop     ax
18769 00001B55 09F6                               or      si, si
18770 00001B57 7403                               jz      short mediadovolid ; check_time      says ">= 2 secs       passed"
18771                                                                        ; (volume id will be checked)
18772 00001B59 31F6                               xor     si, si         ; return "i don't know"
18773                                   mediaret:
18774 00001B5B C3                                 retn
18775                                   ; ---------------------------------------------------------------------------
18776
18777                                   ; somehow the media was changed. look at vid to see. we do not look at fat
18778                                   ; because this may be different since we only set medbyt when doing a read
18779                                   ; or write.
18780
18781                                   mediadovolid:
18782 00001B5C E877EB                             call    GetBp          ; build a new bpb in current bds
18783 00001B5F 72FA                               jb      short mediaret
18784 00001B61 E82D00                             call    check_vid
18785 00001B64 73F5                               jnb     short mediaret
18786 00001B66 E940F2                             jmp     maperror       ; fix up al for       return to dos
18787                                   ; ---------------------------------------------------------------------------
18788
18789                                   ; simple, quick check of latched change. if no indication, then return
18790                                   ; otherwise do expensive check. if the expensive test fails, pop off the
18791                                   ; return and set al = 15 (for invalid media change) which will be returned to
18792                                   ; dos.
18793                                   ;
18794                                   ; for dos 3.3, this will work only for the drive that has changeline.
18795
18796                                   ;   call with es:di -> bds, ds -> Bios_Data
18797                                   ;   ***** warning:  this routine will return one level up on the stack
18798                                   ;                   if an error occurs!
18799
18800                                   checklatchio:
18801
18802                                   ; if returning fake bpb then assume the disk has not changed
18803
18804                                             ; 26/12/2023
18805                                             ;cmp   word [es:di+3Ch], 0 ; [es:di+BDS.opcnt]
18806 00001B69 E8B0FF                           call    chkopcnt
18807 00001B6C 741B                             jz      short checkret ; done if zero
18808
18809                                   ; check for past rom indications. if no rom change indicated, then return ok.
18810
18811                                             ; 26/12/2023
18812                                             ;test   word [es:di+3Fh], 40h
18813                                             ;                      ; test [es:di+BDS.flags], fchanged ; 40h
18814 00001B6E E80E01                           call    checkromchange
18815 00001B71 7416                             jz      short checkret
18816
18817                                   ; we now see that a change line has been seen in the past. let's do the
18818                                   ; expensive verification.
18819
18820 00001B73 E860EB                           call    GetBp          ; build bpb in current bds
18821 00001B76 720F                             jb      short ret_no_error_map ; getbp has already called maperror
18822 00001B78 E81600                           call    check_vid
18823 00001B7B 7207                             jb      short checklatchret ; disk error trying     to read in.
18824 00001B7D 09F6                             or      si, si         ; is changed for sure?
18825 00001B7F 7908                             jns     short checkret
18826 00001B81 E88F00                           call    returnvid
18827                                   checklatchret:
18828 00001B84 E822F2                           call    maperror       ; fix up al for       return to dos
18829                                   ret_no_error_map:
18830 00001B87 F9                               stc
18831 00001B88 5E                               pop     si             ; pop off return address
18832                                   checkret:
18833 00001B89 C3                               retn
18834                                   ; ---------------------------------------------------------------------------
18835
18836                                   ; check the fat and the vid. return in di -1 or 0. return with carry set
18837                                   ; only if there was a disk error. return that error code in ax.
18838                                   ;
18839                                   ;   called with es:di -> bds, ds -> Bios_Data
18840
18841                                   checkfatvid:
18842 00001B8A E8D101                           call    fat_check      ; check the fat and the vid
18843 00001B8D 09F6                             or      si, si
18844 00001B8F 7835                             js      short changed_drv
18845
18846                                   ; the fat was the same. fall into check_vid and check volume id.
18847
```

```
                                        ; fall into check_vid

                                ; ============== S U B   R O U T I N E =======================================

                                ; now with the extended boot record, the logic should be enhanced.
                                ;
                                ; if it is the extended boot record, then we check the volume serial
                                ; number instead of volume id. if it is different, then set si to -1.
                                ;
                                ; if it is same, then si= 1 (no change).
                                ;
                                ; if it is not the extended boot record, then just follows the old
                                ; logic. dos 4.00 will check if the # of fat in the boot record bpb
                                ; is not 0.  if it is 0 then it must be non_fat based system and
                                ; should have already covered by extended boot structure checking.
                                ; so, we will return "i don't know" by setting si to 0.
                                ;
                                ; this routine assume the newest valid boot record is in cs:[disksector].
                                ; (this will be gauranteed by a successful getbp call right before this
                                ; routine.)
                                ;
                                ;   called with es:di -> bds, ds -> bds

                                        ; 26/12/2023 - Retro DOS v5.0
                                        ; 19/10/2022
                                check_vid:

                                ; check the disksector.EXT_BOOT_SIG variable for the extended
                                ; boot signature. if it is set then go to do the extended
                                ; id check otherwise continue with code below

                                        ; 26/12/2023
                                        ;;;
00001B91 833E[6801]00                   cmp     word [disksector+16h], 0 ; BPB_FATSz16
00001B96 7507                           jnz      short chk_vid_1
00001B98 803E[9401]29                   cmp      byte [disksector+42h], 29h ; BS_FAT32_BootSig
                                                        ; [disksector+EXT_BOOT.SIG],EXT_BOOT_SIGNATURE
00001B9D EB05                           jmp      short chk_vid_2
                                chk_vid_1:
                                        ;;;
00001B9F 803E[7801]29                   cmp     byte [disksector+26h], 29h
                                                        ; [disksector+EXT_BOOT.SIG],
                                                        ; EXT_BOOT_SIGNATURE
                                chk_vid_2:      ; 26/12/2023
00001BA4 7427                           jz      short do_ext_check_id
00001BA6 E8E200                         call    haschange
00001BA9 74DE                           jz      short checkret
00001BAB 31F6                           xor     si, si
00001BAD 803E[6201]00                   cmp     byte [disksector+10h], 0 ; BPB_NumFATs
                                                        ; [disksector+EXT_BOOT.BPB+EBPB.NUMBEROFFATS]
00001BB2 7411                           jz      short checkfatret ; don't read vol id
                                                        ; if not fat system
00001BB4 E8F400                         call    read_volume_id
00001BB7 720C                           jb      short checkfatret
00001BB9 E89901                         call    check_volume_id
00001BBC BEFFFF                         mov     si, 0FFFFh      ; -1
                                                        ; definitely changed
00001BBF 7505                           jnz     short changed_drv

00001BC1 46                             inc     si              ; not changed
                                vid_no_changed:
00001BC2 E8C000                         call    resetchanged
                                        ; 12/12/2022
                                        ; cf=0 ('and' instruction in 'resetchanged' clears cf)
                                        ;clc
                                checkfatret:
00001BC5 C3                             retn
                                ; ---------------------------------------------------------------------------

                                        ; 12/12/2022
                                changed_drv:
00001BC6 F8                             clc                     ; cas -- return       no error
00001BC7 C606[1E01]FF                   mov     byte  [tim_drv], 0FFh
                                                        ; ensure that we ask rom for media
00001BCC C3                             retn                    ; check next time round
                                ; ---------------------------------------------------------------------------

                                ; extended id check

                                ; 16/10/2022

                                ; the code to check extended id is basically a check to see if the
                                ; volume serial number is still the same. the volume serial number
                                ; previously read is in cs:disksector.EXT_BOOT_SERIAL
                                ; ds:di points to the bds of the drive under consideration.
                                ; the bds has fields containing the high and low words
                                ; of the volume serial number of the media in the drive.
                                ; compare these fields to the fields mentioned above. if these fields
                                ; do not match the media has changed and so we should jump to the code
                                ; starting at ext_changed else return "i don't know" status
                                ; in the register used for the changeline status and continue executing
                                ; the code given below. for temporary storage use the register which
                                ; has been saved and restored around this block.
                                ;
                                ; bds fields in inc\msbds.inc

                                        ; 26/12/2023 - Retro DOS v5.0
                                        ; 19/10/2022
                                do_ext_check_id:
                                        ; 26/12/2023
                                        ;push   ax
                                        ;;mov   ax, word ptr ds:disksector+27h
                                        ;                       ; [DiskSector+EXT_BOOT.SERIAL]
                                        ;mov    ax, [disksector+27h]
                                ; 26/12/2023
                                %if 1
                                        ;;;
00001BCD 57                             push    di
00001BCE BE[9501]                       mov     si, disksector+43h ; BS_FAT32_VolID
                                                        ; [DiskSector+FAT32_EXT_BOOT.SERIAL]
00001BD1 833E[6801]00                   cmp     word [disksector+16h], 0 ; BPB_FATSz16
00001BD6 7403                           jz      short chk_vid_3
00001BD8 83EE1C                         sub     si, 28          ; BS_VolID
                                        ; si = disksector+27h ; [DiskSector+EXT_BOOT.SERIAL]
                                chk_vid_3:
                                        ; [es:di+89h] = [es:di+BDS.vol_serial]
00001BDB 81C78900                       add     di, 137         ; BDS.vol_serial
00001BDF A7                             cmpsw   ; [DiskSector+EXT_BOOT.SERIAL] ; (or FAT32_EXT_BOOT)
                                                ;               = [di+BDS.vol_serial] ?
00001BE0 7501                           jnz     short chk_vid_4
00001BE2 A7                             cmpsw   ; [DiskSector+EXT_BOOT.SERIAL+2] ; (or FAT32_EXT_BOOT)
                                                ;               = [di+BDS.vol_serial+2] ?
                                chk_vid_4:
00001BE3 5F                             pop     di
```

```
18972                                           ;pop    ax
18973 00001BE4 7504                             jnz     short ext_changed ; not equal/same
18974 00001BE6 31F6                             xor     si, si          ; 0 ; don't know
18975 00001BE8 EBD8                             jmp     short vid_no_changed ; reset the flag
18976                                           ;;;
18977                                   %else
18978                                           ; 02/09/2023
18979                                           xor     si, si ; 0
18980                                           cmp     ax, [es:di+57h]       ; [di+BDS.vol_serial]
18981                                           jnz     short ext_changed
18982                                           mov     ax, [disksector+29h] ; [DiskSector+EXT_BOOT.SERIAL+2]
18983                                           cmp     ax, [es:di+59h]       ; [di+BDS.vol_serial+2]
18984                                           jnz     short ext_changed
18985                                           ;xor    si, si          ; 0
18986                                                                   ; don't know
18987                                           pop     ax
18988                                           jmp     short vid_no_changed
18989                                                                   ; reset the flag
18990                                   %endif
18991
18992                                   ; ---------------------------------------------------------------------------
18993
18994                                   ext_changed:
18995                                           ; 26/12/2023
18996                                           ;pop    ax
18997                                           ; 02/09/2023
18998                                           ;dec    si ; mov si, 0FFFFh ; -1
18999 00001BEA BEFFFF                           mov     si, 0FFFFh      ; -1
19000                                                                   ; disk changed!
19001                                           ; 12/12/2022
19002                                           ; ('changed_drv' clears cf)
19003                                           ;clc
19004 00001BED EBD7                             jmp     short changed_drv
19005
19006                                   ; ---------------------------------------------------------------------------
19007
19008                                   ; at i/o time, we detected the error. now we need to determine whether the
19009                                   ; media was truly changed or not. we return normally if media change unknown.
19010                                   ; and we pop off the call and jmp to harderr if we see an error.
19011                                   ;
19012                                   ; es:di -> bds
19013
19014                                   checkio:
19015 00001BEF 80FC06                           cmp     ah, 6
19016 00001BF2 75D1                             jnz     short checkfatret
19017 00001BF4 E825FF                           call    chkopcnt
19018 00001BF7 74CC                             jz      short checkfatret
19019 00001BF9 E8DAEA                           call    GetBp
19020 00001BFC 7212                             jb      short no_error_map
19021 00001BFE E889FF                           call    checkfatvid
19022 00001C01 7209                             jb      short checkioret ; disk      error trying to       read in.
19023 00001C03 09F6                             or      si, si          ; is changed for sure?
19024 00001C05 7802                             js      short checkioerr ; yes changed
19025 00001C07 45                               inc     bp              ; allowa retry
19026 00001C08 C3                               retn
19027                                   ; ---------------------------------------------------------------------------
19028
19029                                   checkioerr:
19030 00001C09 E80700                           call    returnvid
19031
19032                                   checkioret:
19033 00001C0C F9                               stc                     ; make sure carry gets passed through
19034 00001C0D E955F1                           jmp     harderr
19035                                   ; ---------------------------------------------------------------------------
19036
19037                                   no_error_map:
19038 00001C10 E955F1                           jmp     harderr2
19039
19040                                   ; =============== S U B   R O U TI N E =======================================
19041
19042                                   ; return vid sets up the vid for a return to dos.
19043                                   ;  es:di -> bds, returns pointer in packet to bds_volid
19044                                   ;  **** trashes si! ****
19045
19046                                   returnvid:
19047 00001C13 BE1600                           mov     si, 22          ; extra
19048                                                                   ; offset into pointer to return      value
19049 00001C16 E80700                           call    vid_into_packet
19050 00001C19 B406                             mov     ah, 6
19051 00001C1B F9                               stc
19052 00001C1C C3                               retn
19053
19054                                   ; ---------------------------------------------------------------------------
19055
19056                                   ; moves the pointer to the volid for the drive into the original request packet
19057                                   ; no attempt is made to preserve registers.
19058                                   ;
19059                                   ; assumes es:di -> bds
19060                                   ; **trashes si**
19061
19062                                   media_set_vid:
19063 00001C1D BE0F00                           mov     si, 15          ; trans+1
19064                                                                   ; return the value here      in packet
19065
19066                                           ; fall into vid_into_packet
19067
19068                                   ; =============== S U B   R O U TI N E =======================================
19069
19070                                   ; return pointer to vid in bds at es:di in packet[si]
19071
19072                                           ; 14/04/2024 - Retro DOS v5.0 (PCDOS 7.1 IBMBIO.COM)
19073                                           ; 19/10/2022
19074                                   vid_into_packet:
19075 00001C20 1E                               push    ds              ; return pointer to vid      in bds at es:di       in packet[si]
19076 00001C21 C51E[1200]                       lds     bx, [ptrsav]
19077                                           ;add    di, 75          ; BDS.volid
19078                                           ; 14/04/2024
19079 00001C25 83C77D                           add     di, 125 ; (PCDOS 7.1)
19080 00001C28 8938                             mov     [bx+si], di
19081                                           ;sub    di, 75          ; BDS.volid
19082 00001C2A 83EF7D                           sub     di, 125
19083 00001C2D 8C4002                           mov     [bx+si+2], es
19084 00001C30 1F                               pop     ds
19085                                   dofloppy:   ; 18/12/2022
19086 00001C31 C3                               retn
19087
19088                                   ; ---------------------------------------------------------------------------
19089
19090                                   ;---------------------------------------------------------------------------
19091                                   ;   hidensity - examine a drive/media descriptor to set the media type. if
19092                                   ;   the media descriptor is not f9 (not 96tpi or 3 1/2), we return and let the
19093                                   ;   caller do the rest. otherwise, we pop off the return and jump to the tail
19094                                   ;   of getbp. for 3.5" media, we just return.
19095                                   ;
```

```
19096                                   ;   inputs: es:di point to correct bds for this drive
19097                                   ;           ah has media byte
19098                                   ;
19099                                   ;   outputs:       carry clear
19100                                   ;               no registers modified
19101                                   ;           carry set
19102                                   ;               al = sectors/fat
19103                                   ;               bh = number of root directory entries
19104                                   ;               bl = sectors per track
19105                                   ;               cx = number of sectors
19106                                   ;               dh = sectors per allocation unit
19107                                   ;               dl = number of heads
19108                                   ;
19109                                   ;--------------------------------------------------------------------------
19110
19111                                               ; 26/12/2023 - Retro DOS v5.0
19112                                   hidensity:
19113
19114                                   ; check for correct drive
19115
19116                                               ; 26/12/2023
19117 00001C32 26F6453F02                           test   byte [es:di+3Fh], 2 ; is it special?
19118                                               ; 12/12/2022
19119                                               ;test  byte [es:di+23h], 2
19120                                               ;;test  word [es:di+23h], 2 ; is it special?
19121                                                                           ; [es:di+BDS.flags], fchangeline
19122 00001C37 74F8                                 jz     short dofloppy ; no, do normal floppy test
19123
19124                                   ; we have a media byte that is pretty complex. examine drive information
19125                                   ; table to see what kind it is.
19126
19127                                               ; 26/12/2023
19128 00001C39 26807D3E02                           cmp    byte [es:di+3Eh], 2 ; is it single-media?
19129                                               ;cmp   byte [es:di+22h], 2 ; is it single-media?
19130 00001C3E 74F1                                 jz     short dofloppy ; [es:di+BDS.formfactor], ffSmall
19131                                                                      ; yes, use fatid...
19132                                   ; 96 tpi drive?
19133 00001C40 80FCF9                               cmp    ah, 0F9h
19134 00001C43 75EC                                 jnz    short dofloppy
19135
19136                                   ;------ If formfactor of drive = ffother or ff288 it has to be
19137                                   ;------ a 720K diskette
19138
19139                                               ; 02/09/2023 (PCDOS 7.1)
19140                                               ; 26/12/2023
19141 00001C45 268A453E                             mov    al, [es:di+3Eh] ; [es:di+BDS.formfactor]
19142                                               ;mov   al, [es:di+22h]       ; [es:di+BDS.formfactor]
19143 00001C49 3C07                                 cmp    al, 7
19144                                               ;cmp   byte [es:di+22h], 7 ; [es:di+BDS.formfactor]
19145                                                                          ; ffOther
19146 00001C4B 7413                                 jz     short Is720K
19147 00001C4D 3C09                                 cmp    al, 9
19148                                               ;cmp   byte [es:di+22h], 9 ; [es:di+BDS.formfactor]
19149                                                                          ; ff288
19150 00001C4F 740F                                 jz     short Is720K
19151 00001C51 B007                                 mov    al, 7           ; seven sectors / fat
19152 00001C53 BB0FE0                               mov    bx, 57359       ; 224*256+0Fh
19153                                                                      ; 224 root dir entries
19154                                                                      ; & 0Fh sector max
19155 00001C56 B96009                               mov    cx, 2400        ; 80*15*2
19156                                                                      ; 80 tracks, 15 sectors/track,
19157                                                                      ; 2 sides
19158                                               ; 02/09/2023
19159 00001C59 5A                                   pop    dx              ; pop off return address
19160 00001C5A BA0201                               mov    dx, 258         ; 1*256+2
19161                                                                      ; sectors/allocation unit
19162                                                                      ; & head max
19163                                               ;add   sp, 2           ; pop off return address
19164 00001C5D E9EAEA                               jmp    Has1            ; return to tail of getbp
19165                                   ; --------------------------------------------------------------------------
19166
19167                                   Is720K:
19168                                               ; 02/09/2023
19169 00001C60 5B                                   pop    bx              ; pop off return address
19170                                               ;add   sp, 2           ; pop off return address
19171 00001C61 E9A9EA                               jmp    Has720K         ; return to 720K code
19172                                   ; --------------------------------------------------------------------------
19173
19174                                               ; 18/12/2022
19175                                   ;dofloppy:
19176                                               ;retn
19177
19178                                   ; =============== S U B   R O U T I N E =======================================
19179
19180                                   ; 16/10/2022
19181
19182                                   ;--------------------------------------------------------------------------
19183                                   ; set_changed_dl - sets flag bits according to bits set in bx.
19184                                   ;              essentially used to indicate changeline, or format.
19185                                   ;
19186                                   ;   inputs: dl contains physical drive number
19187                                   ;           bx contains bits to set in the flag field in the bdss
19188                                   ;   outputs:       none
19189                                   ;   registers modified: flags
19190                                   ;
19191                                   ;   called from int13 hooker.  Must preserve ALL registers!!!
19192                                   ;
19193                                   ; in the virtual drive system we *must* flag the other drives as being changed
19194                                   ;--------------------------------------------------------------------------
19195
19196                                               ; 26/12/2023 - Retro DOS v5.0
19197                                   set_changed_dl:
19198 00001C64 06                                   push   es
19199 00001C65 57                                   push   di
19200                                               ;les   di, ds:start_bds
19201                                               ; 19/10/2022
19202 00001C66 C43E[1901]                           les    di, [start_bds]
19203
19204                                   ; note: we assume that the list is non-empty
19205
19206                                   scan_bds:
19207 00001C6A 26385504                             cmp    [es:di+4], dl  ; [es:di+BDS.drivenum]
19208 00001C6E 7504                                 jnz    short get_next_bds
19209
19210                                   ; someone may complain, but this *always* must be done when a disk change is
19211                                   ; noted. there are *no* other compromising circumstances.
19212
19213                                               ; 26/12/2023
19214 00001C70 26095D3F                             or     [es:di+3Fh], bx       ; [es:di+BDS.flags]
19215                                               ;or    [es:di+23h], bx       ; [es:di+BDS.flags]
19216                                                                        ; signal change       on other drive
19217                                   get_next_bds:
19218 00001C74 26C43D                               les    di, [es:di]    ; [es:di+BDS.link]
19219                                                                      ; go to next bds
```

```
19220 00001C77 83FFFF                          cmp     di, 0FFFFh
19221 00001C7A 75EE                            jnz     short scan_bds ; loop unless we hit end of chain
19222 00001C7C 5F                              pop     di
19223 00001C7D 07                              pop     es
19224 00001C7E C3                              retn
19225
19226                                  ; =============== S U B   R O U T I N E =======================================
19227
19228                                  ;---------------------------------------------------------------------------
19229                                  ; checkromchange - see if external program has diddled rom change line.
19230                                  ;
19231                                  ;   inputs: es:di points to current bds.
19232                                  ;   outputs:       zero set - no change
19233                                  ;           zero reset - change
19234                                  ;   registers modified: none
19235                                  ;---------------------------------------------------------------------------
19236
19237                                              ; 26/12/2023 - Retro DOS v5.0
19238                                  checkromchange:
19239                                              ;test  word [es:di+BDS.flags], fchanged ; 40h
19240                                              ; 26/12/2023
19241 00001C7F 26F6453F40                          test   byte [es:di+3Fh], 40h
19242                                              ; 10/12/2022
19243                                              ;test  byte [es:di+23h], 40h
19244                                              ;;test  word [es:di+23h], 40h ; [es:di+BDS.flags]
19245                                                                            ; fchanged
19246 00001C84 C3                                  retn
19247
19248                                  ; =============== S U B   R O U T I N E =======================================
19249
19250                                  ;---------------------------------------------------------------------------
19251                                  ; resetchanged - restore value of change line
19252                                  ;
19253                                  ;   inputs: es:di points to current bds
19254                                  ;   outputs:       none
19255                                  ;   registers modified: none
19256                                  ;---------------------------------------------------------------------------
19257
19258                                              ; 26/12/2023 - Retro DOS v5.0
19259                                  resetchanged:
19260                                              ;and   word [es:di+BDS.flags], ~fchanged ; 0FFBFh
19261                                              ; 26/12/2023
19262 00001C85 2680653FBF                          and    byte [es:di+3Fh], 0BFh
19263                                              ; 10/12/2022
19264                                              ;and   byte [es:di+23h], 0BFh
19265                                              ;;and  word [es:di+23h], 0FFBFh ; [es:di+BDS.flags]
19266                                                                            ; ~fchanged
19267 00001C8A C3                                  retn
19268
19269                                  ; =============== S U B   R O U T I N E =======================================
19270
19271                                  ;---------------------------------------------------------------------------
19272                                  ; haschange - see if drive can supply change line
19273                                  ;
19274                                  ;   inputs: es:di points to current bds
19275                                  ;   outputs:       zero set - no change line available
19276                                  ;           zero reset - change line available
19277                                  ;   registers modified: none
19278                                  ;---------------------------------------------------------------------------
19279
19280                                              ; 26/12/2023 - Retro DOS v5.0
19281                                  haschange:
19282                                              ;test  word [es:di+BDS.flags], fchangeline ; 2
19283                                              ; 26/12/2023
19284 00001C8B 26F6453F02                          test   byte [es:di+3Fh], 2
19285                                              ; 10/12/2022
19286                                              ;test  byte [es:di+23h], 2
19287                                              ;;test  word [es:di+23h], 2 ; [es:di+BDS.flags]
19288                                                                            ; fchangeline
19289 00001C90 C3                                  retn
19290
19291                                  ; ---------------------------------------------------------------------------
19292
19293                                  ; 16/10/2022
19294
19295                                  ;---------------------------------------------------------------------------
19296                                  ; set_volume_id      -    main routine, calls other routines.
19297                                  ; read_volume_id     -    read the volume id and tells if it has been changed.
19298                                  ; transfer_volume_id -    copy the volume id from tmp to special drive.
19299                                  ; check_volume_id    -    compare volume id in tmp area with one expected for drive.
19300                                  ; fat_check          -    see of the fatid has changed in the specified drive.
19301                                  ;---------------------------------------------------------------------------
19302
19303                                  ; set_volume_id
19304                                  ;   if drive has changeline support, read in and set the volume_id
19305                                  ; and the last fat_id byte. if no change line support then do nothing.
19306                                  ;
19307                                  ;   on entry:
19308                                  ;   es:di points to the bds for this disk.
19309                                  ;   ah contains media byte
19310                                  ;
19311                                  ;   on exit:
19312                                  ;   carry clear:
19313                                  ;      successful call
19314                                  ;   carry set
19315                                  ;      error and ax has error code
19316
19317                                  set_volume_id:
19318 00001C91 52                                  push   dx              ; save registers
19319 00001C92 50                                  push   ax
19320 00001C93 E8F5FF                              call   haschange       ; does drive have changeline support?
19321 00001C96 740B                                jz     short setvret   ; no, get out
19322 00001C98 E81000                              call   read_volume_id
19323 00001C9B 7209                                jb     short seterr
19324 00001C9D E8A900                              call   transfer_volume_id ; copy the volume id     to special drive
19325 00001CA0 E8E2FF                              call   resetchanged    ; restore value       of change line
19326                                  setvret:
19327                                              ; 10/12/2022
19328                                              ; cf = 0
19329                                              ;clc                   ; no error, clear carry flag
19330 00001CA3 58                                  pop    ax              ; restore registers
19331 00001CA4 5A                                  pop    dx
19332 00001CA5 C3                                  retn
19333                                  ; ---------------------------------------------------------------------------
19334
19335                                  seterr:
19336 00001CA6 5A                                  pop    dx              ; pop stack but don't overwrite ax
19337 00001CA7 5A                                  pop    dx              ; restore dx
19338 00001CA8 C3                                  retn
19339                                  ; ---------------------------------------------------------------------------
19340 00001CA9 0000                    root_sec:   dw 0                   ; root sector #
19341
19342                                  ; 16/10/2022
19343                                  ;ROOTSEC equ root_sec - DOSBIOSEG_2C7h
```

```
                                ; 09/12/2022
                                ROOTSEC equ root_sec

                                ; =============== S U B   R O U T I N E =======================================

                                ; 16/10/2022

                                ; read_volume_id read the volume id and tells if it has been changed.
                                ;
                                ;   on entry:
                                ;   es:di points to current bds for drive.
                                ;
                                ;   on exit:
                                ;   carry clear
                                ;       si = 1  no change
                                ;       si = 0  ?
                                ;       si = -1 change
                                ;
                                ;   carry set:
                                ;       error and ax has error code.

                                read_volume_id:
00001CAB 52                             push    dx              ; preserve registers
00001CAC 51                             push    cx
00001CAD 53                             push    bx
00001CAE 50                             push    ax
00001CAF 06                             push    es              ; stack the bds last
00001CB0 57                             push    di
00001CB1 1E                             push    ds              ; point es to BIOS_Data
00001CB2 07                             pop     es
00001CB3 BF[4008]                       mov     di, tmp_vid     ; "NO NAME      "
00001CB6 BE[6305]                       mov     si, nul_vid     ; "NO NAME      "
                                        ; 26/12/2023
00001CB9 B90B00                         mov     cx, 11          ; PCDOS 7.1 - 02/09/2023
                                        ;mov    cx, 12          ; initialize tmp_vid to       null vi_id

                                        ;rep    movsb
                                        ; 26/12/2023
                                        ;rep movs byte ptr es:[di], byte ptr cs:[si]
                                        ;db 0FBh,2Eh,0A4h
                                        ;cs     ; nul_vid is in BIOSCODE segment
                                        ;rep movsb
00001CBC F3                             rep
00001CBD 2E                             cs
00001CBE A4                             movsb

00001CBF 5F                             pop     di
00001CC0 07                             pop     es
00001CC1 268A450B                       mov     al, [es:di+11] ; [es:di+BDS.fats]
                                                                ; # of fats
00001CC5 268B4D11                       mov     cx, [es:di+17] ; [es:di+BDS.fatsecs]
                                                                ; sectors / fat
00001CC9 F6E1                           mul     cl              ; size taken by       fats
00001CCB 26034509                       add     ax, [es:di+9]  ; [es:di+BDS.resectors]
                                                                ; add on reserved sectors
                                                                ;
                                                                ; ax is now sector # (0       based)
                                        ; 17/10/2022
00001CCF 2EA3[A91C]                     mov     [cs:ROOTSEC], ax
                                        ;mov    word ptr cs:198Fh, ax ; [cs:root_sec]
                                                                ; 0070h:3EFFh =       2C7h:198Fh
00001CD3 268B450C                       mov     ax, [es:di+12] ; [es:di+BDS.direntries]
                                                                ; # root dir entries
00001CD7 B104                           mov     cl, 4           ; 16 entries/sector
00001CD9 D3E8                           shr     ax, cl          ; divide by 16
                                        ;mov    cx, ax          ; cx is # of sectors to       scan
                                        ; 02/09/2023 (PCDOS 7.1, one byte opcode)
00001CDB 91                             xchg    ax, cx          ; cx is # of sectors to       scan
                                next_sec:
00001CDC 51                             push    cx              ; save outer loop counter
00001CDD 2EA1[A91C]                     mov     ax, [cs:ROOTSEC]
                                        ;mov    ax, word ptr cs:198Fh ; [cs:root_sec]
                                                                ; get sector #
00001CE1 268B4D13                       mov     cx, [es:di+19] ; [es:di+BDS.secpertrack]
                                                                ; sectors / track
00001CE5 31D2                           xor     dx, dx
00001CE7 F7F1                           div     cx

                                ; set up registers for call to read_sector

00001CE9 42                             inc     dx              ; dx= sectors into track
                                                                ; ax= track count from 0
00001CEA 88D1                           mov     cl, dl          ; sector to read
00001CEC 31D2                           xor     dx, dx
00001CEE 26F77515                       div     word [es:di+21] ; [es:di+BDS.heads]
                                                                ; # heads on this disc
00001CF2 88D6                           mov     dh, dl          ; head number
00001CF4 88C5                           mov     ch, al          ; track #
00001CF6 E8BDEB                         call    read_sector     ; get first sector of the root directory,
                                                                ; ds:bx -> directory sector
00001CF9 723F                           jb      short readviderr
00001CFB B91000                         mov     cx, 16          ; # of dir entries in a       block of root
00001CFE B008                           mov     al, 8           ; volume label bit
                                fvid_loop:
                                        ; 02/09/2023 (PCDOS 7.1)
00001D00 382F                           cmp     [bx], ch ; 0
                                        ;cmp    byte [bx], 0 ; end of dir?
00001D02 7433                           jz      short no_vid    ; yes, no vol id
00001D04 803FE5                         cmp     byte [bx], 0E5h ; empty entry?
00001D07 7405                           jz      short ent_loop ; yes, skip
00001D09 84470B                         test    [bx+11], al     ; is volume label bit set in fcb?
00001D0C 750F                           jnz     short found_vid       ; jmp yes
                                ent_loop:
00001D0E 83C320                         add     bx, 32          ; add length of       directory entry
00001D11 E2ED                           loop    fvid_loop
00001D13 59                             pop     cx              ; outer loop
00001D14 2EFF06[A91C]                   inc     word [cs:ROOTSEC]
                                        ;inc    word ptr cs:198Fh ; inc word [root_sec]
                                                                ; next sector
00001D19 E2C1                           loop    next_sec        ; continue
                                notfound:
                                        ; 02/09/2023
                                        ;xor    si, si
00001D1B EB13                           jmp     short fvid_ret
                                ; ---------------------------------------------------------------------------

                                found_vid:
                                        ; 02/09/2023
                                        ; cf = 0  ('test' instruction clears cf)
00001D1D 59                             pop     cx              ; clean stack of outer loop counter
00001D1E 89DE                           mov     si, bx          ; point to volume_id
00001D20 06                             push    es              ; preserve current bds
00001D21 57                             push    di
00001D22 1E                             push    ds
```

```
00001D23 07                             pop     es              ; point es to Bios_Data
00001D24 BF[4008]                       mov     di, tmp_vid     ; "NO NAME      "
00001D27 B90B00                         mov     cx, 11          ; VOLID_SIZ-1
                                                                ; length of string minus nul
00001D2A F3A4                           rep movsb               ; mov volume label to tmp_vid
                                        ;xor    al, al
                                        ; 02/09/2023
00001D2C 91                             xchg    ax, cx          ; ax = 0
00001D2D AA                             stosb                   ; null terminate
                                        ;;xor   si, si
                                        ; 02/09/2023
                                        ;xchg   ax, si          ; si = 0
00001D2E 5F                             pop     di              ; restore current bds
00001D2F 07                             pop     es
                                fvid_ret:
                                        ; 02/09/2023
00001D30 31F6                           xor     si, si ; 0

00001D32 58                             pop     ax
                                        ; 10/12/2022
                                        ; cf = 0
                                        ;clc
                                rvidret:
00001D33 5B                             pop     bx              ; restore registers
00001D34 59                             pop     cx
00001D35 5A                             pop     dx
00001D36 C3                             retn
                                ; ---------------------------------------------------------------------------

                                no_vid:
00001D37 59                             pop     cx              ; clean stack of outer loop counter
                                        ;jmp    short notfound ; not found
                                        ; 02/09/2023
00001D38 EBF6                           jmp     short fvid_ret
                                ; ---------------------------------------------------------------------------

                                readviderr:
00001D3A 5E                             pop     si              ; trash the outer loop counter
00001D3B 5E                             pop     si              ; caller's ax, return error code instead
00001D3C EBF5                           jmp     short rvidret

                                ; ---------------------------------------------------------------------------
                                        ; 26/12/2023 - Retro DOS v5.0
                                        ; 02/09/2023 - Retro DOS v4.2 (IO.SYS optimization)
                                        ; PCDOS 7.1 - IBMBIO.COM - BIOSCODE:1DCFh
                                preset_volid_addr:
00001D3E BE[4008]                       mov     si, tmp_vid     ; "NO NAME    "
                                        ; 26/12/2023
                                        ; PCDOS 7.1
00001D41 83C77D                         add     di, 125         ; BDS.volid
00001D44 B90B00                         mov     cx, 11          ; VOLID_SIZ (12 for MSDOS 5.0-6.22 versions)
                                        ; MSDOS 6.21 (MSDOS 5.0 & 6.?)
                                        ;add    di, 75          ; BDS.volid
                                        ;mov    cx, 12          ; VOLID_SIZ
                                        ;
00001D47 FC                             cld
00001D48 C3                             retn

                                ; =============== S U B   R O U T I N E =======================================

                                ; transfer_volume_id - copy the volume id from tmp to special drive
                                ;
                                ; inputs:   es:di has current bds
                                ; outputs:  bds for drive has volume id from tmp

                                        ; 27/12/2023 - Retro DOS v5.0
                                transfer_volume_id:
00001D49 57                             push    di              ; copy the volume id from tmp to special drive
                                        ;push   si
00001D4A 51                             push    cx
                                        ; 27/12/2023
00001D4B 56                             push    si

                                        ;mov    si, tmp_vid     ; "NO NAME      "
                                        ;;add   di, BDS.volid
                                        ;add    di, 75          ; BDS.volid
                                        ;;mov   cx, VOLID_SIZ
                                        ;mov    cx, 12          ; VOLID_SIZ
                                        ;cld
                                        ; 02/09/2023 (PCDOS 7.1)
00001D4C E8EFFF                         call    preset_volid_addr

00001D4F F3A4                           rep movsb

                                        ; 27/12/2023
00001D51 5E                             pop     si
                                chk_volid_ok:
00001D52 59                             pop     cx
                                        ;pop    si
00001D53 5F                             pop     di
00001D54 C3                             retn

                                ; =============== S U B   R O U T I N E =======================================

                                ;  check_volume_id - compare volume id in tmp area with
                                ;                one expected for drive
                                ;
                                ;   inputs: es:di has current bds for drive
                                ;   outputs:       zero true means it matched

                                        ; 27/12/2023 - Retro DOS v5.0
                                check_volume_id:
00001D55 57                             push    di
00001D56 51                             push    cx

                                        ;mov    si, tmp_vid     ; "NO NAME      "
                                        ;;add   di, BDS.volid
                                        ;add    di, 75          ; BDS.volid
                                        ;;mov   cx, VOLID_SIZ
                                        ;mov    cx, 12          ; VOLID_SIZ
                                        ;cld
                                        ; 02/09/2023 (PCDOS 7.1)
00001D57 E8E4FF                         call    preset_volid_addr

00001D5A F3A6                           repe cmpsb              ; are the 2 volume_ids the same?

                                        ; 27/12/2023
                                        ;pop    cx
                                        ;pop    di
                                        ;retn
00001D5C EBF4                           jmp     short chk_volid_ok

                                ; =============== S U B   R O U T I N E =======================================

```

```
                                ;   fat_check - see of the fatid has changed in the specified drive.
                                ;         - uses the fat id obtained from the boot sector.
                                ;
                                ;   inputs: medbyt is expected fat id
                                ;           es:di points to current bds
                                ;
                                ;   output: si = -1 if fat id different,
                                ;           si = 0 otherwise
                                ;
                                ;   no other registers changed.

                                fat_check:
00001D5E 50                             push    ax
00001D5F 31F6                           xor     si, si          ; say fat id's are same.
00001D61 A0[1F01]                       mov     al, [medbyt]    ; 19/10/2022
00001D64 263A4510                       cmp     al, [es:di+10h]         ; [es:di+BDS.media]
                                                                ; compare it with the bds medbyte
00001D68 7401                           jz      short okret1    ; carry clear
00001D6A 4E                             dec     si
                                okret1:
00001D6B 58                             pop     ax
00001D6C C3                             retn

                                ; ---------------------------------------------------------------------------

                                ; BIOSCODE:1DFEh (PCDOS 7.1 IBMBIO.COM) ; 27/12/2023
                                        ;times 2 db 0

                                ; BIOSCODE:1A69h (MSDOS 6.21 IO.SYS) ((& MSDOS 6.22 IO.SYS))
                                        ;times 7 db 0

                                ; BIOSCODE:180Bh (MSDOS 5.0 IO.SYS)

                                        ; 09/12/2022
                                        ;times 4 db 0  ; 17/10/2022
                                        ;db 4 dup(0)   ; times 4 db 0

                                ; ---------------------------------------------------------------------------

                                        ; 09/12/2022
                                        ;db 0

                                number2div  equ ($-BCode_start)
                                number2mod  equ (number2div % 16)

                                %if (number2mod>0) & (number2mod<16) ; 17/09/2023
00001D6D 00<rep 3h>                     times (16-number2mod) db 0
                                %endif

                                ;align 16

                                ; 09/12/2022
                                BCODE_END   equ $ - BCode_start
                                ; 29/09/2023
                                BCODEEND:
                                ;SYSINITSEG equ IOSYSCODESEG+(BCODE_END>>4)
                                ; 13/12/2022
                                SYSINITOFFSET      equ BCODE_END
                                ; 29/09/2023
                                ;SYSINITOFFSET     equ $-$$
                                SYSINITSEG  equ IOSYSCODESEG+(SYSINITOFFSET>>4)

                                ; 28/09/2023
                                S2SIZE equ $-$$

                                ;--- End of DOSBIOS code segment ---------------------------------------------

                                ; 16/10/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 IO.SYS)
                                ; 01/05/2019 - Retro DOS v4.0
                                ; ============================================================================
                                ; end of BIOSCODE

                                ; ---------------------------------------------------------------------------
                                ; %include sysinit5.s     ; 09/12/2022
                                ; ---------------------------------------------------------------------------

                                ;============================================================================
                                ; (IO.SYS) SYSINIT SEGMENT
                                ;============================================================================
                                ; 09/12/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 IO.SYS)
                                ; 30/12/2022 - Retro DOS v4.2 (Modified MSDOS 6.21 IO.SYS)
                                ;
                                ; 27/12/2023 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMBIO.COM)

                                section .SYSINIT vstart=0

                                ; ****************************************************************************
                                ; SYSINIT.BIN (MSDOS 6.21 IO.SYS) - RETRO DOS v4.0 by ERDOGAN TAN - 21/10/2022
                                ; ----------------------------------------------------------------------------
                                ; Last Update: 04/01/2023 (Modified IO.SYS)  ((Previous: 30/12/2022))
                                ; ----------------------------------------------------------------------------
                                ; Beginning: 03/06/2018 (Retro DOS 3.0), 21/03/2019 (Retro DOS 4.0)
                                ; ----------------------------------------------------------------------------
                                ; Assembler: NASM version 2.15
                                ; ----------------------------------------------------------------------------
                                ;      ((nasm sysinit6.s -l sysinit6.lst -o SYSINIT6.BIN -Z error.txt))
                                ; ----------------------------------------------------------------------------
                                ; Modified from 'sysinit2.s' (SYSINIT2.BIN) file of Retro DOS v3.0 (6/7/2018)
                                ; ----------------------------------------------------------------------------
                                ; Derived from 'SYSINIT1.ASM' and 'SYSINIT2.ASM' files of MSDOS 6.0
                                ; source code by Microsoft, 1991
                                ; ----------------------------------------------------------------------------
                                ; Derived from 'SYSINIT.ASM' file of MSDOS 2.0 (IBM PCDOS v2.0) source code
                                ; by Microsoft, 12/10/1983
                                ; ****************************************************************************
                                ; main file: 'retrodos4.s'
                                ; incbin 'SYSINIT3.BIN' ; (SYINITSEG)

                                ; 30/12/2022 - Retro DOS v4.2
                                ; Retro DOS v4.0 - 2019
                                ; SYSINIT (MSDOS 6.21 IO.SYS) draft: 'sysinit3.s' (01/07/2019)

                                ; 21/10/2022
                                ; ----------------------------------------------------------------------------
                                ; This source code (version) is based on SYSINIT source code of disassembled
                                ; MSDOS 5.0 IO.SYS file (SYSINIT.BIN)
                                ; Dissassembler: Hex-Rays Interactive Disassembler (IDA)
                                ; ----------------------------------------------------------------------------
                                ; Binary file splitter & joiner: FFSJ v3.3

                                ;--------------------------------------------------------------
                                ; SYSINIT.TXT (27/01/1983)
                                ;--------------------------------------------------------------
                                ;    SYSINIT is  a module linked behind the OEM bios.  It takes
```

```
19716                           ;over  the  system  initialization  after  the  OEM  bios   has
19717                           ;performed any  initialization  it  needs  to  do.   Control is
19718                           ;transfered with a long jump to the external  variable  SYSINIT
19719                           ;
19720                           ;
19721                           ;   The OEM  has  the  following  variables declared external:
19722                           ;
19723                           ;   CURRENT_DOS_LOCATION    WORD
19724                           ;
19725                           ;This word  contains  the  segment  number of the DOS before it
19726                           ;is relocated.  The OEM bios must set this value.
19727                           ;
19728                           ;   FINAL_DOS_LOCATION      WORD
19729                           ;
19730                           ;This word contains the segment number of the DOS after SYSINIT
19731                           ;moves it.  The OEM bios must set this value.
19732                           ;
19733                           ;   DEVICE_LIST             DWORD
19734                           ;
19735                           ;This  double  word  pointer  points  to  the  linked  list  of
19736                           ;character and block device drivers.  The  OEM  must  set  this
19737                           ;value.
19738                           ;
19739                           ;   MEMORY_SIZE             WORD
19740                           ;
19741                           ;This word  contains  the  number  of  RAM  paragraphs.  If the
19742                           ;bios doesn't set  this  variable  SYSINIT  will  automatically
19743                           ;calculate it.   NOTE:  systems with PARITY checked memory must
19744                           ;size memory in the BIOS.  SYSINITs method is to  write  memory
19745                           ;and read it back until it gets a mismatch.
19746                           ;
19747                           ;   DEFAULT_DRIVE           BYTE
19748                           ;
19749                           ;This is  the initial default drive when the system first comes
19750                           ;up.  drive a=0, drive b=1,  etc.   If  the  bios  doesn't  set
19751                           ;it then drive a is assumed.
19752                           ;
19753                           ;   BUFFERS                 BYTE
19754                           ;
19755                           ;This is  the  default  number of buffers for the system.  This
19756                           ;value may be overridden by the user in  the  CONFIG.SYS  file.
19757                           ;It is DBed to 2 in SYSINIT it should be greater than 1.
19758                           ;
19759                           ;   FILES                   BYTE
19760                           ;
19761                           ;This is  the  default  number  of  files for the system.  This
19762                           ;value may be overridden by the user in  the  CONFIG.SYS  file.
19763                           ;It is  DBed  to  8 in SYSINIT, values less than 5 are ignored.
19764                           ;
19765                           ;   SYSINIT                 FAR
19766                           ;
19767                           ;The entry  point  of  the  SYSINIT  module.  OEM BIOS jumps to
19768                           ;this label at the end of its INIT code.
19769                           ;
19770                           ;   The OEM  has  the  following  variables declared public:
19771                           ;
19772                           ;   RE_INIT                 FAR
19773                           ;
19774                           ;This is an entry point which allows the BIOS to do some INIT
19775                           ;work  after  the  DOS is initialized.  ALL REGISTERS MUST BE
19776                           ;PRESERVED.  On entry DS points to the first available memory
19777                           ;(after  the DOS).  DS:0 points to a 100H byte program header
19778                           ;prefix which represents  the  "program"  currently  running.
19779                           ;This  program  should  be  thought  of  as  the OEM BIOS and
19780                           ;SYSINIT taken together.  This is not  a  normal  program  in
19781                           ;that  no  memory  is  allocated to it, it is running in free
19782                           ;memory.
19783                           ;NOTES:
19784                           ;     At the time this routine is called SYSINIT occupies the
19785                           ;highest 10K of memory ("highest" is determined by the  value
19786                           ;of the MEMORY_SIZE variable), DO NOT DO WRITES THERE.
19787                           ;     Since this is called AFTER DOS is initialized, you can
19788                           ;make system calls.  This also implies that the code for this
19789                           ;routine    CANNOT   be   thrown   away   by   use   of   the
19790                           ;FINAL_DOS_LOCATION since the DOS has already been moved.
19791                           ;     If you don't want  anything done just set this to point
19792                           ;at a FAR RET instruction.
19793                           
19794                           ; ----------------------------------------------------------------------
19795                           ; TITLE   BIOS SYSTEM INITIALIZATION
19796                           ; ----------------------------------------------------------------------
19797                           
19798                           ;include version.inc
19799                           ; ----------------------------------------------------------------------
19800                           
19801                           ;FALSE   EQU     0
19802                           ;TRUE    EQU     0FFFFh
19803                           
19804                           ;IBMVER         EQU     TRUE
19805                           ;IBMCOPYRIGHT EQU   FALSE
19806                           ;STACKSW    EQU        TRUE             ;Include Switchable Hardware Stacks
19807                           ;IBMJAPVER  EQU     FALSE         ; If TRUE set KANJI true also
19808                           ;MSVER      EQU     FALSE
19809                           ;ALTVECT    EQU     FALSE         ; Switch to build ALTVECT version
19810                           ;KANJI      EQU     FALSE
19811                           
19812                           ;(MSDOS 6.0, versiona.inc, 1991)
19813                           ; ----------------------------------------------------------------------
19814                           ;MAJOR_VERSION  EQU       6
19815                           ;;MINOR_VERSION    EQU     0       ;6.00
19816                           ;MINOR_VERSION  EQU       21      ;6.21  ; 21/03/2019 - Retro DOS v4.0
19817                           
19818                           ; 22/10/2022 - Retro DOS v4.0 (Modified MSDOS 5.0)
19819                           ; ----------------------------------------------------------------------
19820                           ;MAJOR_VERSION   EQU      5
19821                           ;MINOR_VERSION   EQU      0
19822                           
19823                           ; 01/01/2023 - Retro DOS v4.2 (Modified MSDOS 6.21)
19824                           ;MAJOR_VERSION   EQU      6
19825                           ;MINOR_VERSION   EQU      22
19826                           
19827                           ; 21/02/2024 - Retro DOS v5.0 (Modified PCDOS 7.1)
19828                           MAJOR_VERSION   EQU       7
19829                           MINOR_VERSION   EQU       10
19830                           
19831                           expected_version equ      (MINOR_VERSION<<8)+MAJOR_VERSION
19832                           
19833                           ;DOSREVNM equ      00000000b       ; m037 - bits 0-2 = revision number of DOS
19834                                                              ; currently 0.
19835                           DOSREVNM equ       00000111b      ; [[[ 7 for Retro DOS v4.0 ]]] (21/03/2019)
19836                           DOSINROM equ       00001000B       ; bit 3 of ver flags returned in BH
19837                           DOSINHMA equ       00010000B       ; bit 4 of ver flags
19838                           
19839                           ;      if1
```

```
                                ;       %OUT  ... for DOS Version 5.00 ...
                                ;       endif
                                
                                        ;*****************************
                                        ;Each assembler program should:
                                        ;  mov ah,030h                  ;DOS Get Version function
                                        ;  int 021h                     ;Version ret. in AX,minor version first
                                        ;  cmp ax,expected_version      ;ALL utilities should check for an
                                        ;  jne error_handler            ; EXACT version match.
                                        ;*****************************
                                
                                ; ----------------------------------------------------------------------
                                ; device definitions
                                
                                ;Attribute bit masks
                                DEVTYP  EQU     8000h           ;Bit 15 - 1  if Char, 0 if block
                                DEVIOCTL EQU    4000h           ;Bit 14 - CONTROL mode bit
                                ISFATBYDEV EQU  2000h           ;Bit 13 - Device uses FAT ID bytes, comp media.
                                ISCIN   EQU     0001h           ;Bit 0 - This device is the console input.
                                ISCOUT  EQU     0002h           ;Bit 1 - This device is the console output.
                                ISNULL  EQU     0004h           ;Bit 2 - This device is the null device.
                                ISCLOCK EQU     0008h           ;Bit 3 - This device is the clock device.
                                ISIBM   EQU     0010h           ;Bit 4 - This device is special
                                
                                ; The device table list has the form:
                                struc       SYSDEV
00000000 ????????               .NEXT:              resd 1          ;Pointer to next device header
00000004 ????                   .ATT:               resw 1          ;Attributes of the device
00000006 ????                   .STRAT:             resw 1          ;Strategy entry point
00000008 ????                   .INT:               resw 1          ;Interrupt entry point
0000000A ????????????????       .NAME:              resb 8          ;Name of device (only first byte used for block)
                                .size:
                                endstruc
                                
                                ;Static Reguest Header
                                struc       SRHEAD
00000000 ??                     .REQLEN:    resb 1          ;Length in bytes of request block
00000001 ??                     .REQUNIT:   resb 1          ;Device unit number
00000002 ??                     .REQFUNC:   resb 1          ;Type of request
00000003 ????                   .REQSTAT:   resw 1          ;Status Word
00000005 ????????????????               resb 8          ;Reserved for queue links
                                .size:
                                endstruc
                                
                                ;Status word masks
                                STERR   EQU     8000H           ;Bit 15 - Error
                                STBUI   EQU     0200H           ;Bit 9 - Buisy
                                STDON   EQU     0100H           ;Bit 8 - Done
                                STECODE EQU     00FFH           ;Error code
                                WRECODE EQU     0
                                
                                ;Function codes
                                DEVINIT EQU     0               ;Initialization
                                DINITHL EQU     26              ;Size of init header
                                DEVMDCH EQU     1               ;Media check
                                DMEDHL  EQU     15              ;Size of media check header
                                DEVBPB  EQU     2               ;Get BPB
                                DEVRDIOCTL EQU  3               ;IOCTL read
                                DBPBHL  EQU     22              ;Size of Get BPB header
                                DEVRD   EQU     4               ;Read
                                DRDWRHL EQU     22              ;Size of RD/WR header
                                DEVRDND EQU     5               ;Non destructive read no wait (character devs)
                                DRDNDHL EQU     14              ;Size of non destructive read header
                                DEVIST  EQU     6               ;Input status
                                DSTATHL EQU     13              ;Size of status header
                                DEVIFL  EQU     7               ;Input flush
                                ; 21/02/2024
                                ;DFLSHL EQU     15              ;Size of flush header
                                DFLSHL      equ     13      ; PCDOS 7.1 IBMDOS.COM  ; 21/02/2024
                                DEVWRT  EQU     8               ;Write
                                DEVWRTV EQU     9               ;Write with verify
                                DEVOST  EQU     10              ;Output status
                                DEVOFL  EQU     11              ;Output flush
                                DEVWRIOCTL EQU  12              ;IOCTL write
                                
                                ; ----------------------------------------------------------------------
                                struc       SYS_FCB
00000000 ??                     .fcb_drive: resb 1
00000001 ????????????????       .fcb_name:  resb 8
00000009 ??????                 .fcb_ext:   resb 3
0000000C ????                   .fcb_EXTENT:        resw 1
0000000E ????                   .fcb_RECSIZ:        resw 1  ; Size of record (user settable)
00000010 ????                   .fcb_FILSIZ:        resw 1  ; Size of file in bytes; used with the following
                                                            ; word
00000012 ????                   .fcb_DRVBP: resw 1  ; BP for SEARCH FIRST and SEARCH NEXT
00000014 ????                   .fcb_FDATE: resw 1  ; Date of last writing
00000016 ????                   .fcb_FTIME: resw 1  ; Time of last writing
00000018 ??                     .fcb_DEVID: resb 1  ; Device ID number, bits 0-5 if file.
                                                    ; bit 7=0 for file, bit 7=1 for I/O device
                                                    ; If file, bit 6=0 if dirty
                                                    ; If I/O device, bit 6=0 if EOF (input)
                                                    ;               Bit 5=1 if Raw mode
                                                    ;               Bit 0=1 if console input device
                                                    ;               Bit 1=1 if console output device
                                                    ;               Bit 2=1 if null device
                                                    ;               Bit 3=1 if clock device
00000019 ????                   .fcb_FIRCLUS:       resw 1  ; First cluster of file
0000001B ????                   .fcb_CLUSPOS:       resw 1  ; Position of last cluster accessed
0000001D ????                   .fcb_LSTCLUS:       resw 1  ; Last cluster accessed and directory
0000001F ??                                         resb 1  ; pack 2 12 bit numbers into 24 bits...
00000020 ??                     .fcb_NR:    resb 1  ; Next record
00000021 ????????               .fcb_RR:    resb 4  ; Random record
                                .size:
                                endstruc
                                
                                ; ----------------------------------------------------------------------
                                ; Field definition for I/O buffer information
                                
                                ; 22/03/2019 - Retro DOS v4.0 (MSDOS 6.0, BUFFER.INC, 1991)
                                
                                    ; 03/01/2024 - Retro DOS v5.0 (PCDOS 7.1 IBMDOS.COM)
                                
                                struc BUFFINFO
00000000 ????                   .buf_next:  resw 1          ; Pointer to next buffer in list
00000002 ????                   .buf_prev:  resw 1          ; Pointer to prev buffer in list
00000004 ??                     .buf_ID:    resb 1          ; Drive of buffer (bit 7 = 0)
                                                            ; SFT table index (bit 7 = 1)
                                                            ; = FFH if buffer free
00000005 ??                     .buf_flags: resb 1          ; Bit 7 = 1 if Remote file buffer
                                                            ;       = 0 if Local device buffer
                                                            ; Bit 6 = 1 if buffer dirty
                                                            ; Bit 5 = Reserved
                                                            ; Bit 4 = Search bit (bit 7 = 1)
                                                            ; Bit 3 = 1 if buffer is DATA
```

```
                                                ; Bit 2 = 1 if buffer is DIR
                                                ; Bit 1 = 1 if buffer is FAT
                                                ; Bit 0 = Reserved
00000006 ????????               .buf_sector:    resd 1          ; Sector number of buffer (flags bit 7 = 0)
                                ; The next two items are often refed as a word (flags bit 7 = 0)
0000000A ??                     .buf_wrtcnt:    resb 1          ; For FAT sectors, # times sector written out
0000000B ????                   .buf_wrtcntinc: resw 1          ;  "   "      "    , # sectors between each write
0000000D ????                               resw 1 ; * ; 03/01/2024 ; PCDOS 7.1
                                                   ; hw of sectors per FAT
0000000F ????????               .buf_DPB:   resd 1              ; Pointer to drive parameters
00000013 ????                   .buf_fill:  resw 1              ; How full buffer is (flags bit 7 = 1)
00000015 ??                     .buf_reserved:  resb 1          ; make DWORD boundary for 386
00000016 ????                               resw 1 ; * ; 03/01/2024 ; PCDOS 7.1
                                                  ; reserved word for dword boundary
                                .size:      ; 20 bytes ; MSDOS 5.0 to 6.22
                                    ; 24 bytes ; PCDOS 7.1 ; 03/01/2024
                                endstruc

                                %define buf_offset BUFFINFO.buf_sector ; 22/07/2019
                                                ;For buf_flags bit 7 = 1, this is the byte
                                                ;offset of the start of the buffer in
                                                ;the file pointed to by buf_ID. Thus
                                                ;the buffer starts at location
                                                ;buf_offset in the file and contains
                                                ;buf_fill bytes.

                                bufinsiz    equ     BUFFINFO.size  ; Size of structure in bytes


                                buf_Free    equ     0FFh            ; buf_id of free buffer

                                ;Flag byte masks
                                buf_isnet   EQU     10000000B
                                buf_dirty   EQU     01000000B
                                ;***
                                buf_visit   EQU     00100000B
                                ;***
                                buf_snbuf   EQU     00010000B

                                buf_isDATA  EQU     00001000B
                                buf_isDIR   EQU     00000100B
                                buf_isFAT   EQU     00000010B
                                buf_type_0  EQU     11110001B       ; AND sets type to "none"

                                buf_NetID   EQU     bufinsiz

                                ; ----------------------------------------------------------------------

                                ; ----------------------------------------------------------------------
                                ;** DPB - Drive Parameter Block

                                ; 25/03/2019 - Retro DOS v4.0 (MSDOS 6.0, DPB.INC, 1991)

                                ;   BUGBUG - this isn't authorative - it's my probably incomplete and
                                ;   possibly inaccurate deductions from code study... - jgl
                                ;
                                ;   The DPB is DOS's main structure for describing block devices.
                                ;   It contains info about the "Drive" intermingled with info about
                                ;   the FAT file system which is presumably on the drive.  I don't know
                                ;   how those fields are used if it's not the FAT file system - BUGBUG
                                ;
                                ;   The DPBs are statically allocated and chained off of DPBHead.
                                ;   Users scan this chain looking for a match on DPB_DRIVE.
                                ;   The DPBs are built at init time from info in the SYSDEV structure.

                                ; 08/06/2018 - Retro DOS v3.0 (MSDOS 3.3, DPB.INC, 24/07/1987)

                                ; 12/05/2019 - Retro DOS v4.0

                                ; 01/01/2024
                                %if 0

                                struc       DPB
                                .DRIVE:             resb 1          ; Logical drive # assoc with DPB (A=0,B=1,...)
                                .UNIT:              resb 1          ; Driver unit number of DPB
                                .SECTOR_SIZE:       resw 1          ; Size of physical sector in bytes
                                .CLUSTER_MASK:      resb 1          ; Sectors/cluster - 1
                                .CLUSTER_SHIFT:     resb 1          ; Log2 of sectors/cluster
                                .FIRST_FAT: resw 1          ; Starting record of FATs
                                .FAT_COUNT: resb 1          ; Number of FATs for this drive
                                .ROOT_ENTRIES:      resw 1          ; Number of directory entries
                                .FIRST_SECTOR:      resw 1          ; First sector of first cluster
                                .MAX_CLUSTER:       resw 1          ; Number of clusters on drive + 1
                                ; MSDOS 3.3
                                ;.FAT_SIZE: resb 1          ; Number of records occupied by FAT
                                ; MSDOS 6.0
                                .FAT_SIZE:  resw 1          ; Number of records occupied by FAT
                                .DIR_SECTOR:        resw 1          ; Starting record of directory
                                .DRIVER_ADDR:       resd 1          ; Pointer to driver
                                .MEDIA:             resb 1          ; Media byte
                                .FIRST_ACCESS:      resb 1          ; This is initialized to -1 to force a media
                                                                    ; check the first time this DPB is used
                                .NEXT_DPB:  resd 1          ; Pointer to next Drive parameter block
                                .NEXT_FREE: resw 1          ; Cluster # of last allocated cluster
                                .FREE_CNT:  resw 1          ; Count of free clusters, -1 if unknown
                                .size:
                                endstruc

                                %else

                                ; 01/01/2024 - Retro DOS v5.0 (PCDOS 7.1)

                                struc       DPB
00000000 ??                     .DRIVE:             resb 1  ; 0     ; Logical drive # assoc with DPB (A=0,B=1,...)
00000001 ??                     .UNIT:              resb 1  ; 1     ; Driver unit number of DPB
00000002 ????                   .SECTOR_SIZE:       resw 1  ; 2     ; Size of physical sector in bytes
00000004 ??                     .CLUSTER_MASK:      resb 1  ; 4     ; Sectors/cluster - 1
00000005 ??                     .CLUSTER_SHIFT:     resb 1  ; 5     ; Log2 of sectors/cluster
00000006 ????                   .FIRST_FAT: resw 1  ; 6     ; Starting record of FATs
00000008 ??                     .FAT_COUNT: resb 1  ; 8     ; Number of FATs for this drive
00000009 ????                   .ROOT_ENTRIES:      resw 1  ; 9     ; Number of directory entries
0000000B ????                   .FIRST_SECTOR:      resw 1  ; 11    ; First sector of first cluster
0000000D ????                   .MAX_CLUSTER:       resw 1  ; 13    ; Number of clusters on drive + 1
0000000F ????                   .FAT_SIZE:  resw 1  ; 15    ; Number of records occupied by FAT
00000011 ????                   .DIR_SECTOR:        resw 1  ; 17    ; Starting record of directory
00000013 ????????              .DRIVER_ADDR:       resd 1  ; 19    ; Pointer to driver
00000017 ??                     .MEDIA:             resb 1  ; 23    ; Media byte
00000018 ??                     .FIRST_ACCESS:      resb 1  ; 24    ; This is initialized to -1 to force a media
                                                                    ; check the first time this DPB is used
00000019 ????????              .NEXT_DPB:  resd 1  ; 25    ; Pointer to next Drive parameter block
0000001D ????                   .NEXT_FREE: resw 1  ; 29    ; Cluster # of last allocated cluster
0000001F ????                   .FREE_CNT:  resw 1  ; 31    ; Count of free clusters, -1 if unknown
                                ; FAT32 fs ; 01/01/2024
                                ; ref: https://en.wikibooks.org/wiki/
```

```
20088                                   ;       First_steps_towards_system_programming_under_MS-DOS_7/Appendix
20089                                   ;    -- A.03-1. Structure of Drive Parameters Blocks (DPB) ---
20090 00000021 ????                     .FREE_CNT_HW:       resw 1  ; 33    ; High word of free cluster count
20091 00000023 ????                     .EXT_FLAGS: resw 1  ; 35    ; FAT32 extended flags (active FAT number)
20092 00000025 ????                     .FSINFO_SECTOR:     resw 1  ; 37    ; (FAT32 fs) FSINFO structure sector address
20093 00000027 ????                     .BKBOOT_SECTOR:     resw 1  ; 39    ; (FAT32 fs) Backup Boot Sector address
20094 00000029 ????????                 .FCLUS_FSECTOR: resd 1      ; 41    ; The first cluster's first sector address
20095 0000002D ????????                 .LAST_CLUSTER:      resd 1  ; 45    ; The last cluster number
20096 00000031 ????????                 .FAT32_SIZE:        resd 1  ; 49    ; Number of FAT sectors (for FAT32 fs)
20097 00000035 ????????                 .ROOT_CLUSTER:      resd 1  ; 53    ; Root directory's cluster number (FAT32 fs)
20098                                   ; 01/01/2024 - Retro DOS v5.0
20099 00000039 ????????                 .FAT32_NXTFREE:     resd 1  ; 57    ; The next free cluster (for FAT32 fs)
20100                                   .size:              ; 61 bytes ; 01/01/2024 (PCDOS 7.1)
20101                                   endstruc
20102
20103                                   %endif
20104
20105                                   DPBSIZ  EQU     DPB.size  ; Size of the structure in bytes
20106
20107                                   DSKSIZ  EQU DPB.MAX_CLUSTER       ; Size of disk (temp used during init only)
20108
20109                                   ; ----------------------------------------------------------------------------
20110                                   ; 26/03/2018
20111
20112                                   ; IOCTL SUB-FUNCTIONS
20113                                   IOCTL_GET_DEVICE_INFO     EQU     0
20114                                   IOCTL_SET_DEVICE_INFO     EQU     1
20115                                   IOCTL_READ_HANDLE  EQU    2
20116                                   IOCTL_WRITE_HANDLE EQU    3
20117                                   IOCTL_READ_DRIVE   EQU    4
20118                                   IOCTL_WRITE_DRIVE  EQU    5
20119                                   IOCTL_GET_INPUT_STATUS    EQU     6
20120                                   IOCTL_GET_OUTPUT_STATUS EQU       7
20121                                   IOCTL_CHANGEABLE?  EQU    8
20122                                   IOCTL_SHARING_RETRY       EQU     11
20123                                   GENERIC_IOCTL_HANDLE      EQU     12
20124                                   GENERIC_IOCTL             EQU     13
20125
20126                                   ; GENERIC IOCTL SUB-FUNCTIONS
20127                                   RAWIO                     EQU     8
20128
20129                                   ; RAWIO SUB-FUNCTIONS
20130                                   GET_DEVICE_PARAMETERS     EQU     60H
20131                                   SET_DEVICE_PARAMETERS     EQU     40H
20132                                   READ_TRACK          EQU     61H
20133                                   WRITE_TRACK         EQU     41H
20134                                   VERIFY_TRACK              EQU     62H
20135                                   FORMAT_TRACK              EQU     42H
20136
20137                                   ; DEVICETYPE VALUES
20138                                   MAX_SECTORS_IN_TRACK      EQU     63
20139                                   DEV_5INCH           EQU     0
20140                                   DEV_5INCH96TPI            EQU     1
20141                                   DEV_3INCH720KB            EQU     2
20142                                   DEV_8INCHSS         EQU     3
20143                                   DEV_8INCHDS         EQU     4
20144                                   DEV_HARDDISK              EQU     5
20145                                   DEV_OTHER           EQU     7
20146                                   ;DEV_3INCH1440KB    EQU     7
20147                                   DEV_3INCH2880KB           EQU     9
20148                                   ; Retro DOS v2.0 - 26/03/2018
20149                                   ;;DEV_TAPE          EQU     6
20150                                   ;;DEV_ERIMO         EQU     8
20151                                   ;DEV_3INCH2880KB    EQU     9
20152                                   DEV_3INCH1440KB           EQU     10
20153
20154                                   ;MAX_DEV_TYPE             EQU     9       ; MAXIMUM DEVICE TYPE THAT WE
20155                                                                             ; CURRENTLY SUPPORT.
20156                                   MAX_DEV_TYPE              EQU     10
20157
20158                                   struc A_SECTORTABLE
20159 00000000 ????                     .ST_SECTORNUMBER:  resw   1
20160 00000002 ????                     .ST_SECTORSIZE:           resw   1
20161                                   .size:
20162                                   endstruc
20163
20164                                   ; ----------------------------------------------------------------------------
20165                                   ; structure, equates for devmark for mem command.
20166
20167                                   ; 25/03/2019 - Retro DOS v4.0 (MSDOS 6.0, DEVMARK.INC, 1991)
20168
20169                                   struc devmark
20170 00000000 ??                        .id:           resb 1
20171 00000001 ????                      .seg:          resw 1
20172 00000003 ????                      .size:         resw 1
20173 00000005 ??????                    .dum:          resb 3
20174 00000008 ????????????????         .filename: resb 8
20175                                   endstruc
20176
20177                                   devmark_stk equ     'S'
20178                                   devmark_device      equ     'D'
20179                                   devmark_ifs equ     'I'
20180                                   devmark_buf equ     'B'
20181                                   devmark_cds equ     'L' ; lastdrive
20182                                   devmark_files       equ     'F'
20183                                   devmark_fcbs        equ     'X'
20184                                   devmark_inst        equ     'T' ; used for sysinit base for install= command.
20185                                   devmark_ems_stub equ        'E'
20186
20187                                   setbrkdone  equ     00000001b
20188                                   for_devmark equ     00000010b
20189                                   not_for_devmark equ       11111101b
20190
20191                                   ; ----------------------------------------------------------------------------
20192                                   ; Memory arena structure
20193
20194                                   ; 24/03/2019 - Retro DOS v4.0
20195                                   ; (MSDOS 6.0, ARENA.INC)
20196
20197                                   ;** Arena Header
20198
20199                                   struc ARENA
20200 00000000 ??                       .SIGNATURE: resb 1          ; 4D for valid item, 5A for last item
20201 00000001 ????                     .OWNER:             resw 1          ; owner of arena item
20202 00000003 ????                     .SIZE:              resw 1          ; size in paragraphs of item
20203 00000005 ??????                   .RESERVED   resb 3          ; reserved
20204 00000008 ????????????????         .NAME:              resb 8          ; owner file name
20205                                   endstruc
20206
20207                                   ; 12/04/2019
20208
20209                                   arena_owner_system EQU 0   ; free block indication
20210
20211                                   arena_signature_normal    EQU 4Dh ; valid signature, not end of arena
```

```
                                arena_signature_end       EQU 5Ah ; valid signature, last block in arena
                                
                                ; --------------------------------------------------------------------------
                                ; Process data block (otherwise known as program header)
                                
                                ; 23/03/2019 - Retro DOS v4.0
                                
                                ; (MSDOS 6.0 - PDB.INC, 1991)
                                
                                FILPERPROC  EQU     20
                                
                                struc PDB   ; Process_data_block
00000000 ????                   .EXIT_CALL: resw 1          ; INT int_abort system terminate
00000002 ????                   .BLOCK_LEN: resw 1          ; size of execution block
00000004 ??                                 resb 1
00000005 ??????????             .CPM_CALL:  resb 5          ; ancient call to system
0000000A ????????               .EXIT:              resd 1          ; pointer to exit routine
0000000E ????????               .CTRL_C:    resd 1          ; pointer to ^C routine
00000012 ????????               .FATAL_ABORT:       resd 1          ; pointer to fatal error
00000016 ????                   .PARENT_PID:        resw 1          ; PID of parent (terminate PID)
00000018 <res 14h>              .JFN_TABLE:      resb FILPERPROC ; indices into system table
0000002C ????                   .ENVIRON:   resw 1          ; seg addr of environment
0000002E ????????               .USER_STACK:        resd 1          ; stack of self during system calls
00000032 ????                   .JFN_LENGTH:        resw 1          ; number of handles allowed
00000034 ????????               .JFN_POINTER:       resd 1          ; pointer to JFN table
00000038 ????????               .NEXT_PDB:  resd 1          ; pointer to nested PDB's
0000003C ??                     .INTERCON:  resb 1          ; *** jh-3/28/90 ***
0000003D ??                     .APPEND:    resb 1          ; *** Not sure if still used ***
0000003E ????                   .NOVELL_USED:       resb 2          ; Novell shell (redir) uses these
00000040 ????                   .VERSION:   resw 1          ; DOS version reported to this app
00000042 <res Eh>               .PAD1:              resb 14         ;
00000050 ??????????             .CALL_SYSTEM:       resb 5          ; portable method of system call
00000055 ????????????????       .PAD2:              resb 7          ; reserved so FCB 1 can be used as an extended FCB
0000005C <res 10h>              .FCB1:              resb 16         ; default FCB 1
0000006C <res 10h>              .FCB2:              resb 16         ; default FCB 2
0000007C ????????               .PAD3:              resb 4          ; not sure if this is used by PDB_FCB2
00000080 <res 80h>              .TAIL:              resb 128        ; command tail and default DTA
                                ;.size:
                                endstruc
                                
                                ; --------------------------------------------------------------------------
                                ; <system call definitions>
                                
                                ; 23/03/2019 - Retro DOS v4.0
                                
                                ; (MSDOS 6.0 - SYSCALL.INC, 1991)
                                
                                ABORT                           EQU 0   ;  0      0
                                STD_CON_INPUT                   EQU 1   ;  1      1
                                STD_CON_OUTPUT                  EQU 2   ;  2      2
                                STD_AUX_INPUT                   EQU 3   ;  3      3
                                STD_AUX_OUTPUT                  EQU 4   ;  4      4
                                STD_PRINTER_OUTPUT              EQU 5   ;  5      5
                                RAW_CON_IO                      EQU 6   ;  6      6
                                RAW_CON_INPUT                   EQU 7   ;  7      7
                                STD_CON_INPUT_NO_ECHO           EQU 8   ;  8      8
                                STD_CON_STRING_OUTPUT           EQU 9   ;  9      9
                                STD_CON_STRING_INPUT            EQU 10  ; 10      A
                                STD_CON_INPUT_STATUS            EQU 11  ; 11      B
                                STD_CON_INPUT_FLUSH             EQU 12  ; 12      C
                                DISK_RESET                      EQU 13  ; 13      D
                                SET_DEFAULT_DRIVE               EQU 14  ; 14      E
                                FCB_OPEN                        EQU 15  ; 15      F
                                FCB_CLOSE                       EQU 16  ; 16     10
                                DIR_SEARCH_FIRST                EQU 17  ; 17     11
                                DIR_SEARCH_NEXT                 EQU 18  ; 18     12
                                FCB_DELETE                      EQU 19  ; 19     13
                                FCB_SEQ_READ                    EQU 20  ; 20     14
                                FCB_SEQ_WRITE                   EQU 21  ; 21     15
                                FCB_CREATE                      EQU 22  ; 22     16
                                FCB_RENAME                      EQU 23  ; 23     17
                                GET_DEFAULT_DRIVE               EQU 25  ; 25     19
                                SET_DMA                         EQU 26  ; 26     1A
                                GET_DEFAULT_DPB                 EQU 31  ; 31     1F
                                FCB_RANDOM_READ                 EQU 33  ; 33     21
                                FCB_RANDOM_WRITE                EQU 34  ; 34     22
                                GET_FCB_FILE_LENGTH             EQU 35  ; 35     23
                                GET_FCB_POSITION                EQU 36  ; 36     24
                                SET_INTERRUPT_VECTOR            EQU 37  ; 37     25
                                CREATE_PROCESS_DATA_BLOCK       EQU 38  ; 38     26
                                FCB_RANDOM_READ_BLOCK           EQU 39  ; 39     27
                                FCB_RANDOM_WRITE_BLOCK          EQU 40  ; 40     28
                                PARSE_FILE_DESCRIPTOR           EQU 41  ; 41     29
                                GET_DATE                        EQU 42  ; 42     2A
                                SET_DATE                        EQU 43  ; 43     2B
                                GET_TIME                        EQU 44  ; 44     2C
                                SET_TIME                        EQU 45  ; 45     2D
                                SET_VERIFY_ON_WRITE             EQU 46  ; 46     2E
                                ; Extended functionality group
                                GET_DMA                         EQU 47  ; 47     2F
                                GET_VERSION                     EQU 48  ; 48     30
                                KEEP_PROCESS                    EQU 49  ; 49     31
                                GET_DPB                         EQU 50  ; 50     32
                                SET_CTRL_C_TRAPPING             EQU 51  ; 51     33
                                GET_INDOS_FLAG                  EQU 52  ; 52     34
                                GET_INTERRUPT_VECTOR            EQU 53  ; 53     35
                                GET_DRIVE_FREESPACE             EQU 54  ; 54     36
                                CHAR_OPER                       EQU 55  ; 55     37
                                INTERNATIONAL                   EQU 56  ; 56     38
                                ;   Directory Group
                                MKDIR                           EQU 57  ; 57     39
                                RMDIR                           EQU 58  ; 58     3A
                                CHDIR                           EQU 59  ; 59     3B
                                ;   File Group
                                CREAT                           EQU 60  ; 60     3C
                                OPEN                            EQU 61  ; 61     3D
                                CLOSE                           EQU 62  ; 62     3E
                                READ                            EQU 63  ; 63     3F
                                WRITE                           EQU 64  ; 64     40
                                UNLINK                          EQU 65  ; 65     41
                                LSEEK                           EQU 66  ; 66     42
                                CHMOD                           EQU 67  ; 67     43
                                IOCTL                           EQU 68  ; 68     44
                                XDUP                            EQU 69  ; 69     45
                                XDUP2                           EQU 70  ; 70     46
                                CURRENT_DIR                     EQU 71  ; 71     47
                                ;    Memory Group
                                ALLOC                           EQU 72  ; 72     48
                                DEALLOC                         EQU 73  ; 73     49
                                SETBLOCK                        EQU 74  ; 74     4A
                                ;    Process Group
                                EXEC                            EQU 75  ; 75     4B
                                EXIT                            EQU 76  ; 76     4C
                                WAITPROCESS                EQU 77  ; 77     4D
```

```
20336                           FIND_FIRST                      EQU 78  ; 78     4E
20337                           ;   Special Group
20338                           FIND_NEXT                       EQU 79  ; 79     4F
20339                           ; SPECIAL SYSTEM GROUP
20340                           SET_CURRENT_PDB                 EQU 80  ; 80     50
20341                           GET_CURRENT_PDB                 EQU 81  ; 81     51
20342                           GET_IN_VARS                     EQU 82  ; 82     52
20343                           SETDPB                          EQU 83  ; 83     53
20344                           GET_VERIFY_ON_WRITE             EQU 84  ; 84     54
20345                           DUP_PDB                         EQU 85  ; 85     55
20346                           RENAME                          EQU 86  ; 86     56
20347                           FILE_TIMES                      EQU 87  ; 87     57
20348                           ;
20349                           ALLOCOPER               EQU 88  ; 88     58
20350                           ; Network extention system calls
20351                           GetExtendedError        EQU 89  ; 89     59
20352                           CreateTempFile                  EQU 90  ; 90     5A
20353                           CreateNewFile                   EQU 91  ; 91     5B
20354                           LockOper                EQU 92  ; 92     5C Lock and Unlock
20355                           ServerCall              EQU 93  ; 93     5D CommitAll, ServerDOSCall,
20356                                                           ;           CloseByName, CloseUser,
20357                                                           ;           CloseUserProcess,
20358                                                           ;           GetOpenFileList
20359                           UserOper                EQU 94  ; 94     5E Get and Set
20360                           AssignOper              EQU 95  ; 95     5F On, Off, Get, Set, Cancel
20361                           xNameTrans              EQU 96  ; 96     60
20362                           PathParse               EQU 97  ; 97     61
20363                           GetCurrentPSP                   EQU 98  ; 98     62
20364                           Hongeul                 EQU 99  ; 99     63
20365                           ECS_CALL                EQU 99  ; 99     63  ;; DBCS support
20366                           Set_Printer_Flag        EQU 100 ; 100    64
20367                           GetExtCntry             EQU 101 ; 101    65
20368                           GetSetCdPg              EQU 102 ; 102    66
20369                           ExtHandle               EQU 103 ; 103    67
20370                           Commit                          EQU 104 ; 104    68
20371                           GetSetMediaID                   EQU 105 ; 105    69
20372                           IFS_IOCTL               EQU 107 ; 107    6B
20373                           ExtOpen                 EQU 108 ; 108    6C
20374                           ;
20375                           ;ifdef ROMEXEC
20376                           ;ROM_FIND_FIRST                 EQU 109 ; 109     6D
20377                           ;ROM_FIND_NEXT                  EQU 110 ; 110     6E
20378                           ;ROM_EXCLUDE                    EQU 111 ; 111    6F
20379                           ;endif
20380                           ;
20381                           Set_Oem_Handler         EQU 248 ; 248    F8
20382                           OEM_C1                          EQU 249 ; 249    F9
20383                           OEM_C2                          EQU 250 ; 250    FA
20384                           OEM_C3                          EQU 251 ; 251    FB
20385                           OEM_C4                          EQU 252 ; 252    FC
20386                           OEM_C5                          EQU 253 ; 253    FD
20387                           OEM_C6                          EQU 254 ; 254    FE
20388                           OEM_C7                          EQU 255 ; 255    FF
20389
20390                           ; ---------------------------------------------------------------------------
20391                           ; SYSCONF.ASM (MSDOS 3.3 - 24/07/1987)
20392                           ; ---------------------------------------------------------------------------
20393
20394                           ;;  IF      STACKSW
20395
20396                           ;;
20397                           ;; Internal Stack Parameters
20398                           ;EntrySize          equ     8
20399                           ;
20400                           ;MinCount           equ     8
20401                           ;DefaultCount               equ     9
20402                           ;MaxCount           equ     64
20403                           ;
20404                           ;MinSize            equ     32
20405                           ;DefaultSize                equ     128
20406                           ;MaxSize            equ     512
20407
20408                           ;;  ENDIF
20409
20410                           ; ---------------------------------------------------------------------------
20411                           ; BIOSTRUC.INC (MSDOS 3.3 - 24/07/1987)
20412                           ; ---------------------------------------------------------------------------
20413                                                           ;;Rev 3.30 Modification
20414                           ; ROM BIOS CALL PACKET STRUCTURES
20415
20416                           ;*********************************
20417                           ;System Service call ( Int 15h )
20418                           ;*********************************
20419                           ;Function AH = 0C0h, Return system configuration
20420                           ;For PC and PCJR on return:
20421                           ;   (AH)    = 80h
20422                           ;   (CY)    = 1
20423                           ;For PCXT, PC PORTABLE and PCAT on return:
20424                           ;   (AH)    = 86h
20425                           ;   (CY)    = 1
20426                           ;For all others:
20427                           ;   (AH)    = 0
20428                           ;   (CY)    = 0
20429                           ;   (ES:BX) = pointer to system descriptor vector in ROS
20430                           ; System descriptor :
20431                           ;   DW      xxxx            length of descriptor in bytes,
20432                           ;                           minimum length = 8
20433                           ;   DB      xx              model byte
20434                           ;                           0FFh    = PC
20435                           ;                           0FEh    = PC/XT, Portable
20436                           ;                           0FDh    = PC/JR
20437                           ;                           0FCh    = PC/AT
20438                           ;                           0F9h    = Convertable
20439                           ;                           0F8h    = Model 80
20440                           ;                           0E0 thru 0EFh = reserved
20441                           ;
20442                           ;   DB      xx              secondary model byte
20443                           ;                           000h    = PC1
20444                           ;                           000h    = PC/XT, Portable
20445                           ;                           000h    = PC/JR
20446                           ;                           000h    = PC/AT
20447                           ;                           001h    = PC/AT Model 339
20448                           ;                           003h    = PC/RT
20449                           ;                           000h    = Convertable
20450                           ;
20451                           ;   DB      xx              bios revision level
20452                           ;                           00 for first release, subsequent release
20453                           ;                           of code with same model byte and
20454                           ;                           secondary model byte require revison level
20455                           ;                           to increase by one.
20456                           ;
20457                           ;   DB      xx              feature information byte 1
20458                           ;                           X0000000 = 1, bios use DMA channel 3
20459                           ;                                    = 0, DMA channel 3 not used
```

```
                                ;
                                ;                       0X000000 = 1, 2nd Interrupt chip present
                                ;                              = 0, 2nd Interrupt chip not present
                                ;
                                ;                       00X00000 = 1, Real Time Clock present
                                ;                              = 0, Real Time Clock not present
                                ;
                                ;                       000X0000 = 1, Keyboard escape sequence(INT 15h)
                                ;                                     called in keyboard interrupt
                                ;                                     (Int 09h).
                                ;                              = 0, Keyboard escape sequence not
                                ;                                     called.
                                ;                       0000XXXX reserved
                                ;
                                ;   DB      xx          feature information byte 2 - reserved
                                ;
                                ;   DB      xx          feature information byte 2 - reserved
                                ;
                                ;   DB      xx          feature information byte 2 - reserved
                                ;
                                ;   DB      xx          feature information byte 2 - reserved
                                ;

                                ; 22/03/2019
                                struc ROMBIOS_DESC      ; BIOS_SYSTEM_DESCRIPTOR
00000000 ????                   .bios_sd_leng:          resw 1
00000002 ??                     .bios_sd_modelbyte:     resb 1
                                .bios_sd_scnd_modelbyte:
00000003 ??                                     resb 1
00000004 ??                                     resb 1
00000005 ??                     .bios_sd_featurebyte1:  resb 1
00000006 ????????                               resb 4
                                endstruc

                                ;FeatureByte1      bit map equates
                                DMAchannel3         equ 10000000b
                                ScndIntController   equ 01000000b
                                RealTimeClock       equ 00100000b
                                KeyEscapeSeq        equ 00010000b
                                                        ;;End of Modification

                                ; ----------------------------------------------------------------------
                                ; SYSVAR.INC (MSDOS 6.0 - 1991)
                                ; ----------------------------------------------------------------------
                                ; 22/03/2019 - Retro DOS v4.0

                                ;   SCCSID = @(#)sysvar.asm 1.1 85/04/10

                                struc SysInitVars
                                ; MSDOS 3.3
00000000 ????????               .SYSI_DPB:    resd 1                    ; DPB chain
00000004 ????????               .SYSI_SFT:    resd 1                    ; SFT chain
00000008 ????????               .SYSI_CLOCK:  resd 1                    ; CLOCK device
0000000C ????????               .SYSI_CON:    resd 1                    ; CON device
00000010 ????                   .SYSI_MAXSEC: resw 1                    ; maximum sector size
00000012 ????????               .SYSI_BUF:    resd 1                    ; buffer chain
00000016 ????????               .SYSI_CDS:    resd 1                    ; CDS list
0000001A ????????               .SYSI_FCB:    resd 1                    ; FCB chain
0000001E ????                   .SYSI_KEEP:   resw 1                    ; keep count
00000020 ??                     .SYSI_NUMIO:  resb 1                    ; number of block devices
00000021 ??                     .SYSI_NCDS:   resb 1                    ; number of CDS's
00000022 ????????               .SYSI_DEV:    resd 1                    ; device list
                                ; MSDOS 6.0
00000026 ????                   .SYSI_ATTR:     resw 1            ; null device attribute word
00000028 ????                   .SYSI_STRAT:          resw 1            ; null device strategy entry point
0000002A ????                   .SYSI_INTER:          resw 1            ; null device interrupt entry point
0000002C ????????????????       .SYSI_NAME:     resb 8            ; null device name
                                .SYSI_SPLICE:         resb 0            ; TRUE -> splicees being done
00000034 ????                   .SYSI_IBMDOS_SIZE:  resw 1        ; DOS size in paragraphs
00000036 ????????               .SYSI_IFS_DOSCALL@: resd 1        ; IFS DOS service rountine entry
0000003A ????????               .SYSI_IFS:      resd 1            ; IFS header chain
0000003E ????????               .SYSI_BUFFERS:        resw 2            ; BUFFERS= values (m,n)
00000042 ??                     .SYSI_BOOT_DRIVE:   resb 1        ; boot drive A=1 B=2,..
00000043 ??                     .SYSI_DWMOVE:         resb 1            ; 1 if 386 machine
00000044 ????                   .SYSI_EXT_MEM:        resw 1            ; Extended memory size in KB.
                                .size:
                                endstruc

                                ;This is added for more information exchage between DOS, BIOS.
                                ;DOS will give the pointer to SysInitTable in ES:DI. - J.K. 5/29/86

                                ; 22/03/2019
                                struc SysInitVars_Ext
00000000 ????????               .SYSI_InitVars:       resd 1      ; Points to the above structure.
00000004 ????????               .SYSI_Country_Tab: resd 1 ; DOS_Country_cdpg_info
                                endstruc

                                ; 09/06/2018
                                ; 08/06/2018 - Retro DOS v3.0 (MSDOS 3.3)
                                SYSI_DPB    equ    0
                                SYSI_SFT    equ 4
                                SYSI_CLOCK  equ 8
                                SYSI_CON    equ 12
                                SYSI_MAXSEC equ 16
                                SYSI_BUF    equ 18
                                SYSI_CDS    equ 22
                                SYSI_FCB    equ 26
                                SYSI_KEEP   equ 30
                                SYSI_NUMIO  equ    32
                                SYSI_NCDS   equ    33
                                SYSI_DEV    equ 34
                                ; 22/03/2019 - Retro DOS v4.0 (MSDOS 6.0)
                                SYSI_ATTR     equ 38
                                SYSI_STRAT    equ 40
                                SYSI_INTER    equ 42
                                SYSI_NAME     equ 44
                                SYSI_SPLICE   equ 52
                                SYSI_IBMDOS_SIZE  equ 53
                                SYSI_IFS_DOSCALL@ equ 55
                                SYSI_IFS      equ 59
                                SYSI_BUFFERS        equ 63
                                SYSI_BOOT_DRIVE   equ 67
                                SYSI_DWMOVE   equ 68
                                SYSI_EXT_MEM        equ 69

                                ;The SYSI_BUF of SysInitVars points to the following structure

                                EMS_MAP_BUFF_SIZE EQU 12  ; EMS map buffer size

                                struc BUFFINF      ; BUFFINFO
00000000 ????????               .Buff_Queue:          resd 1      ; Head of list of buffers
00000004 ????                   .Dirty_Buff_Count: resw 1 ; number of dirty buffers in list
00000006 ????????               .Cache_ptr:    resd 1      ; pointer to secondary cache
0000000A ????                   .Cache_count:      resw 1 ; number of secondary cache entries
```

```
20584
20585 0000000C ??                    .Buff_In_HMA:         resb 1      ; flag to indicate that buffers
20586                                                        ; are in HMA
20587 0000000D ????????              .Lo_Mem_Buff:         resd 1      ; Ptr to scratch buff in Low Mem
20588                                                            ;  used to read/write on disks
20589 00000011 ????????              .UU_EMS_FIRST_PAGE:       resw 2
20590 00000015 ????                  .UU_EMS_NPA640:           resw 1
20591 00000017 ??                    .UU_EMS_mode:             resb 1   ; no EMS = -1
20592 00000018 ????                  .UU_EMS_handle:           resw 1   ; EMS handle for buffers
20593 0000001A ????                  .UU_EMS_PageFrame_Number: resw 1 ; EMS page frame number
20594 0000001C ????                  .UU_EMS_Seg_Cnt:   resw 1  ; EMS segment count
20595 0000001E ????                  .UU_EMS_Page_Frame:       resw 1   ; EMS page frame segment address
20596 00000020 ????                  .UU_EMS_reserved:  resw 1  ; EMS segment count
20597 00000022 ??                    .UU_EMS_Map_Buff:  resb 1  ; map buffer
20598                                .size:
20599                                endstruc
20600
20601                                ; --------------------------------------------------------------------------
20602                                ; CURDIR.INC (MSDOS 6.0 - 1991)
20603                                ; --------------------------------------------------------------------------
20604                                ; 22/03/2019 - Retro DOS v4.0
20605
20606                                ;** CDS - Current Directory Structure
20607                                ;
20608                                ; CDS items are used bu the internal routines to store cluster numbers and
20609                                ; network identifiers for each logical name.  The ID field is used dually,
20610                                ; both as net ID and for a cluster number for local devices. In the case
20611                                ; of local devices, the cluster number will be -1 if there is a potential
20612                                ; of the disk being changed or if the path must be recracked.
20613                                ;
20614                                ;   Some pathnames have special preambles, such as
20615                                ;
20616                                ;           \\machine\sharename\...
20617                                ;   For these pathnames we can't allow ".." processing to back us
20618                                ;   up into the special front part of the name.  The CURDIR_END field
20619                                ;   holds the address of the seperator character which marks
20620                                ;   the split between the special preamble and the regular
20621                                ;   path list; ".." processing isn't allowed to back us up past
20622                                ;   (i.e., before) CURDIR_END
20623                                ;   For the root, it points at the leading /. For net
20624                                ;   assignments it points at the end (nul) of the initial assignment:
20625                                ;   A:/    \\foo\bar          \\foo\bar\blech\bozo
20626                                ;     ^             ^                   ^
20627
20628                                DIRSTRLEN   EQU    64+3          ; Max length in bytes of directory strings
20629                                TEMPLEN     EQU    DIRSTRLEN*2
20630
20631                                struc              curdir_list
20632                                ; MSDOS 3.3
20633 00000000 <res 43h>            .cdir_text  resb   DIRSTRLEN     ; text of assignment and curdir
20634 00000043 ????                  .cdir_flags resw   1             ; various flags
20635 00000045 ????????              .cdir_devptr       resd   1            ; local pointer to DPB or net device
20636 00000049 ????????              .cdir_ID    resw   2             ; cluster of current dir (net ID)
20637 0000004D ????                  .cdir_usr_word     resw   1
20638 0000004F ????                  .cdir_end   resw   1             ; end of assignment
20639                                ; MSDOS 6.0
20640 00000051 ??                    .cdir_type: resb   1             ; IFS drive (2=ifs, 4=netuse)
20641 00000052 ????????              .cdir_ifd_hdr:     resd   1            ; Ptr to File System Header
20642 00000056 ????                  .cdir_fsda: resb   2             ; File System Dependent Data Area
20643                                .size:
20644                                endstruc
20645
20646                                curdirlen   EQU    curdir_list.size      ; Needed for screwed up
20647                                                                         ; ASM87 which doesn't allow
20648                                                                         ; Size directive as a macro
20649                                                                         ; argument
20650                                %define curdir_netID      dword [curdir_list.cdir_ID]
20651
20652                                ;** Flag values for CURDIR_FLAGS
20653
20654                                ;Flag word masks
20655                                curdir_isnet       EQU    1000000000000000B
20656                                curdir_isifs       EQU    1000000000000000B
20657                                curdir_inuse       EQU    0100000000000000B
20658                                curdir_splice      EQU    0010000000000000B
20659                                curdir_local       EQU    0001000000000000B
20660
20661                                ; --------------------------------------------------------------------------
20662                                ; SF.INC (MSDOS 6.0 - 1991)
20663                                ; --------------------------------------------------------------------------
20664                                ; 25/03/2019 - Retro DOS v4.0
20665
20666                                ; 09/04/2024 - Retro DOS v4.2 (BugFix)
20667                                ; 09/04/2024 - Retro DOS v5.0
20668
20669                                ; system file table
20670
20671                                ;** System File Table SuperStructure
20672                                ;
20673                                ;   The system file table entries are allocated in contiguous groups.
20674                                ;   There may be more than one such groups; the SF "superstructure"
20675                                ;   tracks the groups.
20676
20677                                struc      SF
20678 00000000 ????????              .SFLink:   resd   1
20679 00000004 ????                  .SFCount:  resw   1              ; number of entries
20680 00000006 ????                  .SFTable:  resw   1              ; beginning of array of the following
20681                                .size:
20682                                endstruc
20683
20684                                ;** System file table entry
20685                                ;
20686                                ;   These are the structures which are at SFTABLE in the SF structure.
20687
20688                                struc      SF_ENTRY
20689 00000000 ????                  .sf_ref_count:     resw   1              ; number of processes sharing entry
20690                                                                       ;   if FCB then ref count
20691 00000002 ????                  .sf_mode:  resw   1              ; mode of access or high bit on if FCB
20692 00000004 ??                    .sf_attr:  resb   1              ; attribute of file
20693 00000005 ????                  .sf_flags: resw   1              ;Bits 8-15
20694                                                                       ; Bit 15 = 1 if remote file
20695                                                                       ;       = 0 if local file or device
20696                                                                       ; Bit 14 = 1 if date/time is not to be
20697                                                                       ;   set from clock at CLOSE.  Set by
20698                                                                       ;   FILETIMES and FCB_CLOSE.  Reset by
20699                                                                       ;   other reseters of the dirty bit
20700                                                                       ;   (WRITE)
20701                                                                       ; Bit 13 = Pipe bit (reserved)
20702                                                                       ;
20703                                                                       ; Bits 0-7 (old FCB_devid bits)
20704                                                                       ; If remote file or local file, bit
20705                                                                       ; 6=0 if dirty Device ID number, bits
20706                                                                       ; 0-5 if local file.
20707                                                                       ; bit 7=0 for local file, bit 7
```

```
                                                        ;      =1 for local I/O device
                                                        ; If local I/O device, bit 6=0 if EOF (input)
                                                        ;             Bit 5=1 if Raw mode
                                                        ;             Bit 0=1 if console input device
                                                        ;             Bit 1=1 if console output device
                                                        ;             Bit 2=1 if null device
                                                        ;             Bit 3=1 if clock device
00000007 ????????               .sf_devptr: resd   1    ; Points to DPB if local file, points
                                                        ; to device header if local device,
                                                        ; points to net device header if
                                                        ; remote
0000000B ????                   .sf_firclus:       resw   1          ; First cluster of file (bit 15 = 0)
                                ;.sf_lstclus:      resw   1 ; *
0000000D ????                   .sf_time:   resw   1    ; Time associated with file
0000000F ????                   .sf_date:   resw   1    ; Date associated with file
00000011 ????????               .sf_size:   resd   1    ; Size associated with file
00000015 ????????               .sf_position:      resd   1          ; Read/Write pointer or LRU count for FCBs
                                ;
                                ; Starting here, the next 7 bytes may be used by the file system to store an
                                ; ID
                                ;
00000019 ????                   .sf_cluspos:       resw   1          ; Position of last cluster accessed
0000001B ????????               .sf_dirsec: resd   1 ; 09/04/2024 ; Sector number of directory sector for this file
0000001F ??                     .sf_dirpos: resb   1    ; Offset of this entry in the above
                                ;
                                ; End of 7 bytes of file-system specific info.
                                ;
00000020 <res Bh>               .sf_name:   resb   11   ; 11 character name that is in the
                                                        ; directory entry.  This is used by
                                                        ; close to detect file deleted and
                                                        ; disk changed errors.
                                ; SHARING INFO
0000002B ????????               .sf_chain:  resd   1    ; link to next SF
0000002F ????                   .sf_UID:    resw   1
00000031 ????                   .sf_PID:    resw   1
00000033 ????                   .sf_MFT:    resw   1
00000035 ????                   .sf_lstclus:       resw   1 ; *      ; Last cluster accessed
00000037 ????????               .sf_IFS_HDR:       resd   1 ; **
                                .size:
                                endstruc

                                ; ---------------------------------------------------------------------------
                                ; DOSCNTRY.INC (MSDOS 3.3 - 24/07/1987)
                                ; ---------------------------------------------------------------------------
                                ; 11/06/2018 - Retro DOS v3.0

                                ;Equates for COUNTRY INFORMATION.
                                SetCountryInfo          EQU     1       ;country info
                                SetUcase        EQU     2       ;uppercase table
                                SetLcase        EQU     3       ;lowercase table (Reserved)
                                SetUcaseFile            EQU     4       ;uppercase file spec table
                                SetFileList     EQU     5       ;valid file character list
                                SetCollate      EQU     6       ;collating sequence
                                SetDBCS         EQU     7       ;double byte character set
                                SetALL                  EQU     -1      ;all the entries

                                ;DOS country and code page information table structure.
                                ;Internally, IBMDOS gives a pointer to this table.
                                ;IBMBIO, MODE and NLSFUNC modules communicate with IBMDOS through
                                ;this structure.

                                struc country_cdpg_info ; DOS_country_cdpg_info
00000000 ????????????????       .ccInfo_reserved : resb    8       ;reserved for internal use
00000008 <res 40h>              .ccPath_CountrySys:        resb    64      ;path and filename for country info
00000048 ????                   .ccSysCodePage:            resw    1       ;system code page id
0000004A ????                   .ccNumber_of_entries:      resw    1 ; dw 5
0000004C ??                     .ccSetUcase:               resb    1 ; db SetUcase ; = 2
0000004D ????????               .ccUcase_ptr:              resd    1       ;pointer to Ucase table

00000051 ??                     .ccSetUcaseFile:   resb    1 ; db SetUcaseFile ; = 4
00000052 ????????               .ccFileUcase_ptr:  resd    1       ;pointer to File Ucase table

00000056 ??                     .ccSetFileList:            resb    1 ; db SetFileList ; = 5
00000057 ????????               .ccFileChar_ptr:   resd    1       ;pointer to File char list table

0000005B ??                     .ccSetCollate:             resb    1 ; db SetCollate ; = 6
0000005C ????????               .ccCollate_ptr:            resd    1       ;pointer to collate table

00000060 ??                     .ccSetCountryInfo: resb    1 ; db SetCountryInfo ; = 1
00000061 ????                   .ccCountryInfoLen: resw    1       ;length of country info
00000063 ????                   .ccDosCountry:             resw    1       ;system country code id
00000065 ????                   .ccDosCodePage:            resw    1       ;system code page id
00000067 ????                   .ccDFormat:        resw    1       ;date format
00000069 ??????????             .ccCurSymbol:              resb    5 ; db "    ",0
                                                           ;5 byte of (currency symbol+0)
0000006E ????                   .cc1000Sep:        resb    2 ; db " ",0 ;2 byte of (1000 sep. + 0)
00000070 ????                   .ccDecSep:         resb    2 ; db " ",0 ;2 byte of (Decimal sep. + 0)
00000072 ????                   .ccDateSep:        resb    2 ; db " ",0 ;2 byte of (date sep. + 0)
00000074 ????                   .ccTimeSep:        resb    2 ; db " ",0 ;2 byte of (time sep. + 0)
00000076 ??                     .cccFormat:        resb    1       ;currency format flags
00000077 ??                     .cccSigDigits:             resb    1       ;# of digits in currency
00000078 ??                     .ccTFormat:        resb    1       ;time format
00000079 ????????               .ccMono_Ptr:               resd    1       ;monocase routine entry point
0000007D ????                   .ccListSep:        resb    2 ; db " ",0 ;data list separator
0000007F <res Ah>               .ccReserved_area:  resw    5 ; dw 5 dup(?) ;reserved
                                .size:
                                endstruc

                                NEW_COUNTRY_SIZE    equ  country_cdpg_info.size - country_cdpg_info.ccDosCountry

                                ; ==========================================================================
                                ; retrodos4.s (offset addresses in MSDOS.SYS or RETRODOS.SYS)
                                ; ==========================================================================
                                ; 21/03/2019 - Retro DOS v4.0
                                ; 21/10/2022 - Retro DOS v4.0 (MOdified MSDOS 5.0 IO.SYS)

                                ;KERNEL_SEGMENT    equ 0070h  ; (IO.SYS loading segment, BIOS_DATA segment)
                                ; 21/10/2022
                                DOSBIODATASEG equ 0070h    ; (IO.SYS loading segment, BIOS_DATA segment)
                                ; 22/10/2022
                                ;DOSBIOCODESEG equ 02C7h ; (MSDOS 5.0 IO.SYS, BIOS_CODE segment)
                                ; 09/12/2022
                                DOSBIOCODESEG equ IOSYSCODESEG

                                ; Note: These offset addresses must be changed when the code
                                ;   in retrodos4.s (MSDOS.SYS) file will be changed.

                                ; (following addresses can be verified by searching them in retrodos4.lst)

                                ; 09/12/2022
                                %if 0

                                ; 13/05/2019
```

```
20832                           ;IsWin386         equ 08CFh
20833                           ;V86_Crit_SetFocus equ 08D0h
20834                           ; 21/10/2022
20835                           IsWin386          equ 08D0h
20836                           V86_Crit_SetFocus equ 08D1h
20837                           
20838                           ;seg_reinit   equ 0772h ; not used in Retro DOS v4.0
20839                           ; 21/10/2022 - Retro DOS v4.0 (MOdified MSDOS 5.0 IO.SYS)
20840                           seg_reinit    equ 0032h ; DOSBIOCODESEG:0032h
20841                           
20842                           ;SysinitPresent      equ 08FCh
20843                           ; 21/10/2022
20844                           SysinitPresent       equ 08FDh
20845                           
20846                           inHMA                equ 000Dh
20847                           xms           equ 000Eh
20848                           ;FreeHMAPtr   equ 08F6h
20849                           ;multrk_flag         equ 0533h
20850                           ;ec35_flag    equ 0535h
20851                           ;EOT          equ 012Eh
20852                           ; 21/10/2022
20853                           FreeHMAPtr    equ 08F7h
20854                           multrk_flag   equ 052Fh
20855                           ec35_flag     equ 0531h
20856                           EOT           equ 012Ch
20857                           
20858                           ;NextStack    equ 08BFh
20859                           ;IT_StackLoc         equ 08C5h
20860                           ;IT_StackSize        equ 08C9h
20861                           ; 21/10/2022
20862                           NextStack     equ 08C0h
20863                           IT_StackLoc   equ 08C6h
20864                           IT_StackSize         equ 08CAh
20865                           
20866                           ;MoveDOSIntoHMA      equ 08F8h
20867                           ; 21/10/2022
20868                           MoveDOSIntoHMA       equ 08F9h
20869                           
20870                           ;INT19SEM equ 0644h ; 01/05/2019 - retrodos4.lst
20871                           ;I19_LST  equ 0645h ; 27/03/2019 - retrodos4.lst
20872                           ; 21/10/2022
20873                           INT19SEM equ 0640h ; (iosys5.txt)
20874                           I19_LST  equ 0641h ; (iosys5.txt)
20875                           
20876                           %endif
20877                           
20878                           ; 09/12/2022
20879                           seg_reinit equ _seg_reinit
20880                           ec35_flag  equ ec35flag
20881                           INT19SEM   equ int19sem
20882                           I19_LST    equ i19_lst
20883                           
20884                           INT19OLD02 equ I19_LST+1 ; 0642h ; 21/10/2022
20885                           INT19OLD08 equ I19_LST+6
20886                           INT19OLD09 equ I19_LST+11
20887                           INT19OLD0A equ I19_LST+16
20888                           INT19OLD0B equ I19_LST+21
20889                           INT19OLD0C equ I19_LST+26
20890                           INT19OLD0D equ I19_LST+31
20891                           INT19OLD0E equ I19_LST+36
20892                           INT19OLD70 equ I19_LST+41
20893                           INT19OLD72 equ I19_LST+46
20894                           INT19OLD73 equ I19_LST+51
20895                           INT19OLD74 equ I19_LST+56
20896                           INT19OLD76 equ I19_LST+61
20897                           INT19OLD77 equ I19_LST+66 ; 0683h ; 21/10/2022
20898                           
20899                           ; 09/12/2022
20900                           %if 0
20901                           
20902                           ;keyrd_func equ 04E9h
20903                           ;keysts_func       equ 04EAh
20904                           ;t_switch   equ 04F6h
20905                           ; 21/10/2022
20906                           keyrd_func  equ 04E5h
20907                           keysts_func equ 04E6h
20908                           t_switch    equ 04F2h
20909                           
20910                           ; 22/10/2022
20911                           SYSINITSEG  equ 046Dh  ; SYSINIT segment
20912                           BCODE_END   equ (SYSINITSEG-DOSBIOCODESEG)*16 ; = 1A60h
20913                           BCODE_START equ 30h  ; (offset BiosDataWord in DOSBIOCODESEG)
20914                           RE_INIT            equ 089Bh ; (re_init offset in DOSBIODATASEG)
20915                           
20916                           %endif
20917                           
20918                           ; 09/12/2022
20919                           BCODESTART  equ BIOSDATAWORD
20920                           RE_INIT            equ re_init
20921                           
20922                           ; ----------------------------------------------------------------------------
20923                           ; CONFIG.INC (MSDOS 6.0 - 1991)
20924                           ; ----------------------------------------------------------------------------
20925                           ; 15/04/2019 - Retro DOS v4.0
20926                           
20927                           CONFIG_BEGIN        equ  '['
20928                           CONFIG_BREAK        equ  'C'
20929                           CONFIG_BUFFERS      equ  'B'
20930                           CONFIG_COMMENT      equ  'Y'
20931                           CONFIG_COUNTRY      equ  'Q'
20932                           CONFIG_DEVICE       equ  'D'
20933                           CONFIG_DEVICEHIGH   equ  'U'
20934                           CONFIG_DOS          equ  'H'
20935                           CONFIG_DRIVPARM     equ  'P'
20936                           CONFIG_FCBS         equ  'X'
20937                           CONFIG_FILES        equ  'F'
20938                           CONFIG_INCLUDE      equ  'J'
20939                           CONFIG_INSTALL      equ  'I'
20940                           CONFIG_INSTALLHIGH  equ  'W'
20941                           CONFIG_LASTDRIVE    equ  'L'
20942                           CONFIG_MENUCOLOR    equ  'R'
20943                           CONFIG_MENUDEFAULT  equ  'A'
20944                           CONFIG_MENUITEM     equ  'E'
20945                           CONFIG_MULTITRACK   equ  'M'
20946                           CONFIG_NUMLOCK      equ  'N'
20947                           CONFIG_REM          equ  'O'
20948                           CONFIG_SEMICOLON    equ  ';'
20949                           CONFIG_SET          equ  'V'
20950                           CONFIG_SHELL        equ  'S'
20951                           CONFIG_STACKS       equ  'K'
20952                           CONFIG_SUBMENU      equ  'O'
20953                           CONFIG_SWITCHES     equ  '1'
20954                           
20955                           CONFIG_UNKNOWN      equ  'Z'
```

```
20956
20957                           ; 13/05/2024 - Retro DOS v5.0 (PCDOS 71 IBMBIO.COM)
20958                           CONFIG_DOSDATA      equ  'T'
20959
20960                           CONFIG_OPTION_QUERY equ 80h
20961
20962                           ; ----------------------------------------------------------------------
20963                           ; SYSINIT1.ASM (MSDOS 6.0 - 1991)
20964                           ; ----------------------------------------------------------------------
20965                           ; 21/03/2019 - Retro DOS v4.0
20966
20967                           trueequ    0FFFFh
20968                           false      equ    0
20969                           cr  equ    13
20970                           lf  equ    10
20971                           tab equ    9
20972
20973                           multMULT      equ  4Ah
20974                           multMULTGETHMAPTR  equ    1
20975                           multMULTALLOCHMA   equ    2
20976
20977                           ;NOEXEC    equ     FALSE
20978
20979                           stacksw    equ     true   ;include switchable hardware stacks
20980                           mycds_size equ     88     ;size of curdir_list. if it is not
20981                                              ;the same, then will generate compile error.
20982
20983                           entrysize   equ     8
20984
20985                           mincount    equ     8
20986                           defaultcount equ    9
20987                           maxcount    equ     64
20988
20989                           minsize     equ     32
20990                           defaultsize equ     128
20991                           maxsize     equ     512
20992
20993                           ;%define allocbyte  byte [es:bp+0]
20994                           ;%define intlevel   byte [es:bp+1]
20995                           ;%define savedsp    word [es:bp+2]
20996                           ;%define savedss    word [es:bp+4]
20997                           ;%define newsp         word [es:bp+6]
20998
20999                           allocbyte   equ     0
21000                           intlevel    equ     1
21001                           savedsp     equ     2
21002                           savedss     equ     4
21003                           newsp       equ     6
21004
21005                           free    equ     0
21006                           allocated   equ     1
21007                           overflowed  equ     2
21008                           clobbered   equ     3
21009
21010                           ;---------------------------------------
21011                           ; external variable defined in ibmbio module for multi-track
21012
21013                           multrk_on equ      10000000b ;user specified mutitrack=on,or system turns
21014                                               ; it on after handling config.sys file as a
21015                                               ; default value,if multrk_flag = multrk_off1.
21016                           multrk_off1 equ 00000000b ;initial value. no "multitrack=" command entered.
21017                           multrk_off2 equ 00000001b ;user specified multitrack=off.
21018
21019                           ; SYSINITSEG       SEGMENT PUBLIC 'SYSTEM_INIT'
21020
21021                           SYSINIT$:
21022                               ;IF    STACKSW
21023                               ; include MSSTACK.INC ;Main stack program and data definitions
21024                               ; include STKMES.INC  ;Fatal stack error message
21025                               ;   public Endstackcode
21026                           ;Endstackcode      label byte
21027                               ;ENDIF
21028
21029                           ; 05/07/2018
21030                           ; ----------------------------------------------------------------------
21031                           ; 04/06/2018 - Retro DOS v3.0
21032
21033                           ; ----------------------------------------------------------------------
21034                           ; 21/10/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 IO.SYS - SYSINIT)
21035                           ; ----------------------------------------------------------------------
21036
21037                           ;   MSStack.inc
21038                           ;
21039                           ;   Interrupt level 2, 3, 4, 5, 6, 7,(10, 11, 12, 14, 15 - AT level)
21040                           ;   should follow the standard Interrupt Sharing Scheme which has
21041                           ;   a standard header structure.
21042                           ;   Fyi, the following shows the relations between
21043                           ;   the interrupt vector and interrupt level.
21044                           ; VEC(Hex)    2  8  9  A  B  C    D  E  70  72  73  74  76  77
21045                           ; LVL(Deci)   9  0  1  2  3  4    5  6   8  10  11  12  14  15
21046                           ;   MSSTACK module modifies the following interrupt vectors
21047                           ;   to meet the standard Interrupt Sharing standard;
21048                           ;     A, B, C, D, E, 72, 73, 74, 76, 77.
21049                           ;   Also, for interrupt level 7 and 15, the FirstFlag in a standard header
21050                           ;   should be initialized to indicat whether this interrupt handler is
21051                           ;   the first (= 80h) or not.  The FirstFlag entry of INT77h's
21052                           ;   program header is initialized in STKINIT.INC module.
21053                           ;   FirstFlag is only meaningful for interrupt level 7 and 15.
21054                           ;
21055
21056                           ;  User specifies the number of stack elements - default = 9
21057                           ;                                         minimum = 8
21058                           ;                                         maximum = 64
21059                           ;
21060                           ;  Intercepts Asynchronous Hardware Interrupts only
21061                           ;
21062                           ;  Picks a stack from pool of stacks and switches to it
21063                           ;
21064                           ;  Calls the previously saved interrupt vector after pushing flags
21065                           ;
21066                           ;  On return, returns the stack to the stack pool
21067                           ;
21068
21069                           ; This is a modification of STACKS:
21070                           ; 1. To fix a bug which was causing the program to take up too much space.
21071                           ; 2. To dispense stack space from hi-mem first rather than low-mem first.
21072                           ;    . Clobbers the stack that got too big instead of innocent stack
21073                           ;    . Allows system to work if the only stack that got too big was the most
21074                           ;      deeply nested one
21075                           ; 3. Disables NMI interrupts while setting the NMI vector.
21076                           ; 4. Does not intercept any interrupts on a PCjr.
21077                           ; 5. Double checks that a nested interrupt didn't get the same stack.
21078                           ; 6. Intercepts Ints 70, 72-77 for PC-ATs and other future products
21079
```

```
                                        ;EVEN
                                ;align 2
                                        ; 21/10/2022

00000000 0000                           dw      0       ; spare field but leave these in order
00000002 0000                   stackcount: dw  0
00000004 0000                   stackat:    dw  0
00000006 0000                   stacksize:  dw  0
00000008 0000                   stacks:             dw      0
0000000A 0000                           dw      0

0000000C [0800]                 firstentry: dw  stacks
0000000E [4800]                 lastentry:  dw  stacks+(defaultcount*entrysize)-entrysize
00000010 [4800]                 nextentry:  dw  stacks+(defaultcount*entrysize)-entrysize

                                ;********************************************************************
                                ; THESE ARE THE INDIVIDUAL INTERRUPT HANDLERS

                                ; ------------------------------------------------------------------

00000012 00000000               old02:      dd      0

                                int02:

                                ; ********************************************************************
                                ;
                                ; this is special support for the pc convertible / nmi handler
                                ;
                                ;   on the pc convertible, there is a situation where an nmi can be
                                ;   caused by using the "out" instructions to certain ports. when this
                                ;   occurs, the pc convertible hardware *guarantees* that **nothing**
                                ;   can stop the nmi or interfere with getting to the nmi handler. this
                                ;   includes other type of interrupts (hardware and software), and
                                ;   also includes other type of nmi's. when any nmi has occured,
                                ;   no other interrtupt (hardware, software or nmi) can occur until
                                ;   the software takes specific steps to allow further interrupting.
                                ;
                                ;   for pc convertible, the situation where the nmi is generated by the
                                ;   "out" to a control port requires "fixing-up" and re-attempting. in
                                ;   otherwords, it is actually a "restartable exception". in this
                                ;   case, the software handler must be able to get to the stack in
                                ;   order to figure out what instruction caused the problem, where
                                ;   it was "out"ing to and what value it was "out"ing.  therefore,
                                ;   we will not switch stacks in this situation. this situation is
                                ;   detected by interrogating port 62h, and checking for a bit value
                                ;   of 80h. if set, *****do not switch stacks*****.
                                ;
                                ; ********************************************************************

00000016 50                             push    ax
00000017 06                             push    es
00000018 B800F0                         mov     ax,0F000h
0000001B 8EC0                           mov     es,ax
                                        ; 02/11/2022
0000001D 26803EFEFFF9                   cmp     byte [es:0FFFEh],0F9h ; mdl_convert ; check if convertible
00000023 07                             pop     es
00000024 750C                           jne     short normal02

00000026 E462                           in      al,62h          ; PC/XT PPI port C. Bits:
                                                                ; 0-3: values of DIP switches
                                                                ; 5: 1=Timer 2 channel out
                                                                ; 6: 1=I/O channel check
                                                                ; 7: 1=RAM parity check error occurred.
00000028 A880                           test    al,80h
0000002A 7406                           jz      short normal02
                                special02:
0000002C 58                             pop     ax
0000002D 2EFF2E[1200]                   jmp     far [cs:old02]
                                normal02:
00000032 58                             pop     ax
00000033 E81101                         call    do_int_stacks
00000036 [1200]                         dw      old02

                                ; ------------------------------------------------------------------

00000038 00000000               old08:      dd      0

                                int08:
0000003C E80801                         call    do_int_stacks
0000003F [3800]                         dw      old08

                                ; ------------------------------------------------------------------

00000041 00000000               old09:      dd      0

                                int09:

                                ; keyboard interrupt must have a three byte jump, a nop and a zero byte
                                ; as its first instruction for compatibility reasons

00000045 EB02                           jmp     short keyboard_lbl
00000047 90                             nop
00000048 00                             db      0

                                keyboard_lbl:
00000049 E8FB00                         call    do_int_stacks
0000004C [4100]                         dw      old09

                                ; ------------------------------------------------------------------

0000004E 00000000               old70:      dd      0

                                int70:
00000052 E8F200                         call    do_int_stacks
00000055 [4E00]                         dw      old70

                                ; ------------------------------------------------------------------

                                ;   irp     a,<0a,0b,0c,0d,0e,72,73,74,76,77>
                                ;public     int&a
                                ;public     old&a
                                ;public     firstflag&a
                                ;int&a      proc    far
                                ;   jmp     short entry_int&a&_stk
                                ;old&a      dd      0               ;forward pointer
                                ;   dw        424bh         ;compatible signature for int. sharing
                                ;firstflag&a db   0               ;the firstly hooked.
                                ;   jmp     short intret_&a       ;reset routine. we don't care this.
                                ;   db      7 dup (0)       ;reserved for future.
                                ;entry_int&a&_stk:
                                ;   call    do_int_stacks
                                ;   dw      old&a
                                ;intret_&a:
                                ;   iret
```

```
21204                                   ;int&a      endp
21205                                   ;   endm
21206
21207                                   ; --------------------------------------------------------------------------
21208
21209                                   int0A:
21210 00000057 EB10                         jmp     short entry_int0A_stk
21211 00000059 00000000                 old0A:      dd      0
21212 0000005D 4B42                         dw      424Bh
21213                                   firstflag0A:
21214 0000005F 00                           db      0
21215 00000060 EB0C                         jmp     short intret_0A
21216 00000062 00<rep 7h>                   times   7 db 0
21217
21218                                   entry_int0A_stk:
21219 00000069 E8DB00                       call    do_int_stacks
21220 0000006C [5900]                       dw      old0A
21221                                   intret_0A:
21222 0000006E CF                           iret
21223
21224                                   ; --------------------------------------------------------------------------
21225
21226                                   int0B:
21227 0000006F EB10                         jmp     short entry_int0B_stk
21228 00000071 00000000                 old0B:      dd      0
21229 00000075 4B42                         dw      424Bh
21230                                   firstflag0B:
21231 00000077 00                           db      0
21232 00000078 EB0C                         jmp     short intret_0B
21233 0000007A 00<rep 7h>                   times   7 db 0
21234
21235                                   entry_int0B_stk:
21236 00000081 E8C300                       call    do_int_stacks
21237 00000084 [7100]                       dw      old0B
21238                                   intret_0B:
21239 00000086 CF                           iret
21240
21241                                   ; --------------------------------------------------------------------------
21242
21243                                   int0C:
21244 00000087 EB10                         jmp     short entry_int0C_stk
21245 00000089 00000000                 old0C:      dd      0
21246 0000008D 4B42                         dw      424Bh
21247                                   firstflag0C:
21248 0000008F 00                           db      0
21249 00000090 EB0C                         jmp     short intret_0C
21250 00000092 00<rep 7h>                   times   7 db 0
21251
21252                                   entry_int0C_stk:
21253 00000099 E8AB00                       call    do_int_stacks
21254 0000009C [8900]                       dw      old0C
21255                                   intret_0C:
21256 0000009E CF                           iret
21257
21258                                   ; --------------------------------------------------------------------------
21259
21260                                   int0D:
21261 0000009F EB10                         jmp     short entry_int0D_stk
21262 000000A1 00000000                 old0D:      dd      0
21263 000000A5 4B42                         dw      424Bh
21264                                   firstflag0D:
21265 000000A7 00                           db      0
21266 000000A8 EB0C                         jmp     short intret_0D
21267 000000AA 00<rep 7h>                   times   7 db 0
21268
21269                                   entry_int0D_stk:
21270 000000B1 E89300                       call    do_int_stacks
21271 000000B4 [A100]                       dw      old0D
21272                                   intret_0D:
21273 000000B6 CF                           iret
21274
21275                                   ; --------------------------------------------------------------------------
21276
21277                                   int0E:
21278 000000B7 EB10                         jmp     short entry_int0E_stk
21279 000000B9 00000000                 old0E:      dd      0
21280 000000BD 4B42                         dw      424Bh
21281                                   firstflag0E:
21282 000000BF 00                           db      0
21283 000000C0 EB0C                         jmp     short intret_0E
21284 000000C2 00<rep 7h>                   times   7 db 0
21285
21286                                   entry_int0E_stk:
21287 000000C9 E87B00                       call    do_int_stacks
21288 000000CC [B900]                       dw      old0E
21289                                   intret_0E:
21290 000000CE CF                           iret
21291
21292                                   ; --------------------------------------------------------------------------
21293
21294                                   int72:
21295 000000CF EB10                         jmp     short entry_int72_stk
21296 000000D1 00000000                 old72:      dd      0
21297 000000D5 4B42                         dw      424Bh
21298                                   firstflag72:
21299 000000D7 00                           db      0
21300 000000D8 EB0C                         jmp     short intret_72
21301 000000DA 00<rep 7h>                   times   7 db 0
21302
21303                                   entry_int72_stk:
21304 000000E1 E86300                       call    do_int_stacks
21305 000000E4 [D100]                       dw      old72
21306                                   intret_72:
21307 000000E6 CF                           iret
21308
21309                                   ; --------------------------------------------------------------------------
21310
21311                                   int73:
21312 000000E7 EB10                         jmp     short entry_int73_stk
21313 000000E9 00000000                 old73:      dd      0
21314 000000ED 4B42                         dw      424Bh
21315                                   firstflag73:
21316 000000EF 00                           db      0
21317 000000F0 EB0C                         jmp     short intret_73
21318 000000F2 00<rep 7h>                   times   7 db 0
21319
21320                                   entry_int73_stk:
21321 000000F9 E84B00                       call    do_int_stacks
21322 000000FC [E900]                       dw      old73
21323                                   intret_73:
21324 000000FE CF                           iret
21325
21326                                   ; --------------------------------------------------------------------------
21327
```

```
                                int74:
000000FF EB10                       jmp     short entry_int74_stk
00000101 00000000               old74:      dd      0
00000105 4B42                       dw      424Bh
                                firstflag74:
00000107 00                         db      0
00000108 EB0C                       jmp     short intret_74
0000010A 00<rep 7h>                 times   7 db 0

                                entry_int74_stk:
00000111 E83300                     call    do_int_stacks
00000114 [0101]                     dw      old74
                                intret_74:
00000116 CF                         iret

                                ; --------------------------------------------------------------------------

                                int76:
00000117 EB10                       jmp     short entry_int76_stk
00000119 00000000               old76:      dd      0
0000011D 4B42                       dw      424Bh
                                firstflag76:
0000011F 00                         db      0
00000120 EB0C                       jmp     short intret_76
00000122 00<rep 7h>                 times   7 db 0

                                entry_int76_stk:
00000129 E81B00                     call    do_int_stacks
0000012C [1901]                     dw      old76
                                intret_76:
0000012E CF                         iret

                                ; --------------------------------------------------------------------------

                                int77:
0000012F EB10                       jmp     short entry_int77_stk
00000131 00000000               old77:      dd      0
00000135 4B42                       dw      424Bh
                                firstflag77:
00000137 00                         db      0
00000138 EB0C                       jmp     short intret_77
0000013A 00<rep 7h>                 times   7 db 0

                                entry_int77_stk:
00000141 E80300                     call    do_int_stacks
00000144 [3101]                     dw      old77
                                intret_77:
00000146 CF                         iret

                                ; --------------------------------------------------------------------------

                                ;****************************************************************************
                                ;common routines
                                ;****************************************************************************

                                ; do interrupt stack switching. the fake return address holds
                                ; a pointer to the far-pointer of the actual interrupt
                                ; service routine

                                ; 21/10/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 SYSINIT)
                                ; 21/03/2019 - Retro DOS v4.0

                                ;allocbyte   equ 0
                                ;intlevel    equ 1
                                ;savedsp     equ 2
                                ;savedss     equ 4
                                ;newsp       equ 6

                                ; 30/12/2022 - Retro DOS v4.2 (Modified MSDOS 6.21 SYSINIT)
                                ; (MSDOS 6.21 IO.SYS, SYSINIT:0147h)

                                do_int_stacks:
00000147 50                         push    ax
00000148 55                         push    bp
00000149 06                         push    es
0000014A 2E8E06[0A00]               mov     es,[cs:stacks+2]      ; Get segment of stacks
0000014F 2E8B2E[1000]               mov     bp,[cs:nextentry]     ; get most likely candidate
00000154 B001                       mov     al,allocated ; 1
                                    ; 21/10/2022
                                    ;xchg   [es:bp+allocbyte],al
                                    ; 11/12/2022
00000156 26864600                   xchg    [es:bp],al            ; grab the entry
0000015A 3C00                       cmp     al,free ; 0           ; still avail?
0000015C 7551                       jne     short notfree02

0000015E 2E832E[1000]08             sub     word [cs:nextentry],entrysize ; set for next interrupt

                                found02:
00000164 26896602                   mov     [es:bp+savedsp],sp    ; save sp value
00000168 268C5604                   mov     [es:bp+savedss],ss    ; save ss also

0000016C 89E8                       mov     ax,bp                 ; temp save of table offset

0000016E 268B6E06                   mov     bp,[es:bp+newsp]      ; get new SP value
                                    ; 21/10/2022
                                    ;mov    bp,[es:bp+6]
                                    ; 11/12/2022
                                    ;cmp    [es:bp+0],ax
00000172 26394600                   cmp     [es:bp],ax            ; check for offset into table
00000176 7544                       jne     short foundbad02

                                    ; 02/07/2023 (MSDOS 6.21 SYSINIT code)
00000178 8CC0                       mov     ax,es                 ; point ss,sp to the new stack
0000017A 8EC5                       mov     es,bp
0000017C 89E5                       mov     bp,sp
0000017E 8B6E06                     mov     bp,[bp+6]
00000181 8ED0                       mov     ss,ax
00000183 8CC4                       mov     sp,es
00000185 8EC0                       mov     es,ax
00000187 2E8B6E00                   mov     bp,[cs:bp]

                                    ; 21/10/2022 (MSDOS 5.0 SYSINIT code)
                                    ;push    bp
                                    ;mov     bp,sp
                                    ;mov     ax,[bp+8]
                                    ;pop     bp
                                    ;push    es
                                    ;pop     ss
                                    ;mov     sp,bp
                                    ;mov     bp,ax
                                    ; 11/12/2022
                                    ;;mov   bp,[cs:bp+0]
                                    ;mov    bp,[cs:bp]
```

```
21452 0000018B 9C                          pushf                           ; go execute the real interrupt handler
21453                                      ; 11/12/2022
21454 0000018C 2EFF5E00                    call    far [cs:bp]             ;  which will iret back to here
21455                                      ; 21/10/2022
21456                                      ;call  far [cs:bp+0]
21457
21458 00000190 89E5                        mov     bp,sp                   ; retrieve the table offset for us
21459                                      ; 11/12/2022
21460 00000192 268B6E00                    mov     bp,[es:bp]              ;  but leave it on the stack
21461                                      ; 21/10/2022
21462                                      ;mov   bp,[es:bp+0]
21463 00000196 268E5604                    mov     ss,[es:bp+savedss]      ; get old stack back
21464 0000019A 268B6602                    mov     sp,[es:bp+savedsp]
21465
21466                                      ; 11/12/2022
21467                                      ;mov   byte [es:bp+allocbyte],free ; free the entry
21468                                      ; 21/10/2022
21469 0000019E 26C6460000                  mov     byte [es:bp],free ; 0
21470 000001A3 2E892E[1000]                mov     [cs:nextentry],bp       ; setup to use next time
21471
21472 000001A8 07                          pop     es
21473 000001A9 5D                          pop     bp                      ; saved on entry
21474 000001AA 58                          pop     ax                      ; saved on entry
21475 000001AB 83C402                      add     sp,2
21476 000001AE CF                          iret                            ; done with this interrupt
21477
21478                                  notfree02:
21479 000001AF 3C01                        cmp     al,allocated            ; error flag
21480 000001B1 7404                        je      short findnext02        ;  no, continue
21481                                      ; 11/12/2022
21482                                      ;xchg  [es:bp+allocbyte],al    ;  yes, restore error value
21483                                      ; 21/10/2022
21484 000001B3 26864600                    xchg    [es:bp],al
21485
21486                                  findnext02:
21487 000001B7 E81200                      call    longpath
21488 000001BA EBA8                        jmp     short found02
21489
21490                                  foundbad02:
21491 000001BC 2E3B2E[0C00]                cmp     bp,[cs:firstentry]
21492 000001C1 72F4                        jc      short findnext02
21493 000001C3 89C5                        mov     bp,ax                   ; flag this entry
21494                                      ; 11/12/2022
21495                                      ;mov   byte [es:bp+allocbyte],clobbered
21496                                      ; 21/10/2022
21497 000001C5 26C6460003                  mov     byte [es:bp],clobbered ; 3
21498 000001CA EBEB                        jmp     short findnext02        ; keep looking
21499
21500                                  ; ---------------------------------------------------------------------------
21501
21502                                  ; Common routines
21503
21504                                  longpath:
21505                                      ; 21/03/2019
21506 000001CC 2E8B2E[0E00]                mov     bp,[cs:lastentry]       ; start with last entry in table
21507                                  lploopp:
21508                                      ; 11/12/2022
21509                                      ;cmp   byte [es:bp+allocbyte],free ; is entry free?
21510                                      ; 21/10/2022
21511 000001D1 26807E0000                  cmp     byte [es:bp],free
21512 000001D6 7512                        jne     short inuse             ;  no, try next one
21513
21514 000001D8 B001                        mov     al,allocated
21515                                      ; 11/12/2022
21516                                      ;xchg  [es:bp+allocbyte],al    ; allocate entry
21517                                      ; 21/10/2022
21518 000001DA 26864600                    xchg    [es:bp],al
21519 000001DE 3C00                        cmp     al,free                 ; is it still free?
21520 000001E0 7414                        je      short found             ;  yes, go use it
21521
21522 000001E2 3C01                        cmp     al,allocated            ; is it other than Allocated or Free?
21523 000001E4 7404                        je      short inuse             ;  no, check the next one
21524
21525                                      ; 11/12/2022
21526                                      ;mov   [es:bp+allocbyte],al    ;  yes, put back the error state
21527                                      ; 21/10/2022
21528 000001E6 26884600                    mov     [es:bp],al
21529                                  inuse:
21530 000001EA 2E3B2E[0C00]                cmp     bp,[cs:firstentry]
21531 000001EF 7406                        je      short fatal
21532 000001F1 83ED08                      sub     bp,entrysize
21533 000001F4 EBDB                        jmp     short lploopp
21534                                  found:
21535 000001F6 C3                          retn
21536                                  fatal:
21537 000001F7 1E                          push    ds
21538 000001F8 B800F0                      mov     ax,0F000h               ;look at the model byte
21539 000001FB 8ED8                        mov     ds,ax
21540 000001FD 803EFEFFF9                  cmp     byte [0FFFEh],0F9h ; mdl_convert ; convertible?
21541 00000202 1F                          pop     ds
21542 00000203 7504                        jne     short skip_nmis
21543
21544 00000205 B007                        mov     al,07h                  ; disable pc convertible nmis
21545 00000207 E672                        out     72h,al
21546
21547                                  skip_nmis:
21548 00000209 FA                          cli                             ; disable and mask
21549 0000020A B0FF                        mov     al,0FFh                 ;   all other ints
21550 0000020C E621                        out     021h,al
21551 0000020E E6A1                        out     0A1h,al
21552
21553 00000210 8CCE                        mov     si,cs
21554 00000212 8EDE                        mov     ds,si
21555 00000214 BE[3B02]                    mov     si,fatal_msg
21556                                  ;SR;
21557                                  ;   We set all foci to this VM to issue the stack failure message
21558                                  ;
21559 00000217 50                          push    ax
21560 00000218 1E                          push    ds
21561                                      ;;mov  ax,Bios_Data ; 0070h
21562                                      ;mov   ax,KERNEL_SEGMENT ; 0070h
21563                                      ; 21/10/2022
21564 00000219 B87000                      mov     ax,DOSBIODATASEG
21565 0000021C 8ED8                        mov     ds,ax
21566
21567                                      ;test  byte [08D0h],1          ; (MSDOS 6.21, IO.SYS - SYSINIT:021Eh)
21568 0000021E F606[1208]01                test    byte [IsWin386],1 ; (retrodos4.sys, offset: ****h)
21569 00000223 1F                          pop     ds
21570 00000224 58                          pop     ax
21571 00000225 7405                        jz      short fatal_loop        ; win386 not present, continue
21572
21573                                      ;;call  far ptr 0070h:08D1h ; (MSDOS 621, IO.SYS - SYSINIT:0227h)
21574                                      ;call  KERNEL_SEGMENT:V86_Crit_SetFocus ; set focus to this VM
21575                                      ; 21/10/2022
```

```
21576 00000227 9A[1308]7000              call   DOSBIODATASEG:V86_Crit_SetFocus ; 0070h:08D1h
21577                                    ;
21578                                    ;SR; We do not bother about the returned status of this call.
21579                                    ;
21580                                    fatal_loop:
21581 0000022C AC                            lodsb
21582 0000022D 3C24                          cmp     al,'$'
21583 0000022F 7408                          je      short fatal_done
21584
21585 00000231 B307                          mov     bl,7
21586 00000233 B40E                          mov     ah,14
21587 00000235 CD10                          int     10h                     ; whoops, this enables ints
21588 00000237 EBF3                          jmp     short fatal_loop
21589
21590                                    fatal_done:
21591 00000239 EBFE                          jmp     short fatal_done
21592
21593
21594                                    ; 21/03/2019 - Retro DOS v4.0 (MSDOS 6.0, SYSINIT1.ASM, 1991)
21595                                    ; ----------------------------------------------------------------------
21596                                    ;   include msbio.cl5             ; fatal stack error message
21597
21598                                    ; MSDOS 6.21, IO.SYS, SYSINIT:023Bh
21599
21600                                    ; STKMES.INC - MSDOS 3.3 (24/07/1987)
21601                                    ; ----------------------------------------------------------------------
21602                                    ; 04/06/2018 - Retro DOS v3.0
21603
21604                                    fatal_msg:
21605 0000023B 0D0A                          db      0Dh,0Ah
21606 0000023D 070D0A                        db      7,0Dh,0Ah
21607 00000240 496E7465726E616C20-           db      "Internal stack overflow",0Dh,0Ah
21607 00000249 737461636B206F7665-
21607 00000252 72666C6F770D0A
21608 00000259 53797374656D206861-           db      "System halted",0Dh,0Ah,"$"
21608 00000262 6C7465640D0A24
21609
21610                                    endstackcode:
21611
21612                                    ; ----------------------------------------------------------------------
21613                                    ; SYINIT1.ASM (MSDOS 6.0, 1991) 'SYSINIT' jump addr from 'MSINIT.ASM'
21614                                    ; ----------------------------------------------------------------------
21615                                    ; 04/06/2018 - Retro DOS v3.0 (MSDOS 3.3, SYSINIT1.ASM, 24/07/1987)
21616
21617                                    ; 22/03/2019 - Retro DOS v4.0
21618
21619                                    ; SYSINIT:0269h (MSDOS 6.21 IO.SYS, SYSINIT segment, offset: 0269h)
21620
21621                                    ; ('SYSINIT:' location/address is used in 'retrodos4.s'. If following
21622                                    ; address will be changed, it must also be changed in 'retrodos4.s'.)
21623
21624                                    ; 21/10/2022- Retro DOS v4.0 (Modified MSDOS 5.0 IO.SYS)
21625                                    ; ----------------------------------------------------------------------
21626                                    ; SYSINITSEG:0267h (MSDOS 5.0 IO.SYS, SYSINIT segment, offset: 0267h)
21627
21628                                    ; SYSINIT:0269h (MSDOS 6.22 IO.SYS, SYSINIT segment, offset: 0269h)
21629
21630                                    ; 29/12/2023- Retro DOS v5.0 (Modified PCDOS 7.1 IBMBIO.COM)
21631                                    ; ----------------------------------------------------------------------
21632                                    ; SYSINITSEG:0269h (PCDOS 7.1 IBMBIO.COM, SYSINIT segment, offset: 0269h)
21633
21634                                    SYSINIT:
21635 00000269 E9AD01                          JMP GOINIT
21636                                        ;JMP    SYSIN ; 25/02/2018 - Retro DOS 2.0 modification
21637
21638                                    ; ----------------------------------------------------------------------
21639
21640                                    struc DDHighInfo
21641 00000000 ????????                  .ddhigh_CSegPtr resd 1    ; pointer to code segment to be relocated
21642 00000004 ????                      .ddhigh_CSegLen resw 1    ; length of code segment to be relocated
21643 00000006 ????????                  .ddhigh_CallBak resd 1    ; pointer to the call back routine
21644                                    endstruc
21645
21646                                    ; 22/03/2019 - Retro DOS v4.0
21647
21648 0000026C 00                       runhigh: db 0
21649
21650                                    ; 02/11/2022
21651                                    ;align 4
21652
21653                                    DOSINFO:
21654 0000026D 00000000                      dd      0       ; address of the DOS Sysini Variables
21655                                    ;MSDOS:
21656                                    dos_temp_location: ; dword ; MSDOS 6.0
21657                                    dosinit:           ; MSDOS 6.0
21658 00000271 0000                          dw      0
21659
21660                                    ; 02/11/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
21661                                    ;FINAL_DOS_LOCATION: ; 20/04/2019 - Retro DOS v4.0
21662                                    ;   dw      0
21663                                    ;MSDOS 5.0 IO.SYS - SYSINIT:0271h
21664
21665                                    CURRENT_DOS_LOCATION:
21666 00000273 0000                          dw      0
21667
21668                                    ;DOSSIZE: ; Retro DOS 2.0 feature - 25/02/2018
21669                                    ;   dw      0   ; 'MSDOS.BIN' kernel size in words
21670
21671                                    ; 22/10/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 IO.SYS)
21672                                    ; (MSDOS 5.0 MSDOS.SYS size is 37394 bytes)
21673                                    ;DOSSIZE equ       0A000h  ; (MSDOS 6.0 - SYSINIT1.ASM - 1991)
21674                                    ; 30/12/2023 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMBIO.COM)
21675                                    ; 03/09/2023 (PCDOS 7.1 IBMDOS.COM size is 42566 bytes, 04/12/2003)
21676                                    DOSSIZE equ 0B000h  ; (PCDOS 7.1 - SYSINIT)
21677
21678                                    DEVICE_LIST:
21679 00000275 00000000                      dd      0
21680
21681                                    ; 04/06/2018 - Retro DOS v3.0
21682                                    ; 28/03/2018
21683                                    ;; MSDOS 3.3 - SYSINIT1.ASM - 24/07/1987
21684                                    ;
21685                                    sysi_country:
21686 00000279 00000000                      dd      0 ; 5/29/86 Pointer to country table in DOS
21687
21688                                    ; MSDOS 6.0
21689 0000027D 00000000                  dos_segreinit:     dw      0,0     ; room for dword
21690
21691 00000281 0000                       lo_doscod_size:    dw      0       ; dos code size when in low mem
21692 00000283 0000                       hi_doscod_size:    dw      0       ; dos code size when in HMA
21693
21694 00000285 0000                       def_php:    dw      0
21695
21696                                    ; M022--
```

```
                                ; pointer for calling into Bios_Code for re-initializing segment values.
                                ;  call with ax = new segment for Bios_Code. Notice that we'll
                                ;  call it in its temporary home, cuz seg_reinit won't get moved to
                                ;  the new home.

                                ;Bios_Code  equ     KERNEL_SEGMENT  ; 0070h
                                ; 21/10/2022
                                ;DOSBIOCODESEG     equ     02C7h ; (MSDOS 5.0 IO.SYS)

                                ; 22/10/2022
                                seg_reinit_ptr:     ; label dword
00000287 [3200]                             dw      seg_reinit ; Bios_Code:0032h for MSDOS 6.21 IO.SYS
                                temp_bcode_seg:
                                            ;dw     Bios_Code  ; 02CCh for MSDOS 6.21 IO.SYS
                                            ; 22/10/2022
00000289 0203                               dw      DOSBIOCODESEG ; 02C7h for MSDOS 5.0 IO.SYS
                                                            ; 364h for PCDOS 7.1 IBMBIO.COM - 29/12/2023
                                fake_floppy_drv:
0000028B 00                                 db      0       ; set to 1 if this machine
                                                            ; does not have any floppies!!!

                                ; Internal Stack Parameters

0000028C 0900                   stack_count:        dW      defaultcount ; 9
0000028E 8000                   stack_size: dw      defaultsize  ; 128
00000290 00000000               stack_addr: dd      0

                                ; 05/06/2018 - Retro DOS v3.0

                                ; various default values

00000294 0100                   MEMORY_SIZE:        dw      1

                                ; 22/03/2019 - Retro DOS v4.0 (MSDOS 6.0 source, MSDOS 6.21 disassembled src.)

00000296 0000                   RPLMemTop:  dw      0  ; 22/10/2022 (MSDOS 5.0 IO.SYS SYSINIT:0294h)
00000298 00                     DEFAULT_DRIVE:      db      0       ; initialized by ibminit.
00000299 FFFF                   buffers:    dw      0FFFFh  ; initialized during buffer allocation
0000029B 0000                   h_buffers:  dw      0       ; # of the heuristic buffers. initially 0.
0000029D 0000                   singlebuffersize: dw        0       ; maximum sector size + buffer head

0000029F 08                     FILES:      db      8       ; enough files for pipe
000002A0 04                     FCBS:       db      4       ; performance for recycling
000002A1 00                     KEEP:       db      0       ; keep original set
000002A2 05                     NUM_CDS: db 5       ; 5 net drives

                                ; 22/10/2022 (MSDOS 5.0 SYSINIT)
                                ;;CONFBOT: dw       0
                                ;;ALLOCLIM: dw      0
                                ;CONFBOT: ; 02/11/2022
                                ;top_of_cdss: dw 0

                                ; 30/12/2022 - Retrodos v4.2 (MSDOS 6.21 SYSINIT)
                                ; (SYSINIT:02A3h)
000002A3 0000                   CONFBOT: dw 0
000002A5 0000                   ALLOCLIM: dw        0
000002A7 0000                   top_of_cdss: dw 0

                                ; 02/11/2022 (MSDOS 5.0 SYSINIT)
                                ; 30/12/2022 (MSDOS 6.21 SYSINIT)
                                ;ALLOCLIM: dw       0       ; (SYSINIT:02A3h)

000002A9 413A5C00               DirStrng: db        "A:\",0 ; string for the root directory of a drive

                                ; 30/12/2022 - Retro DOS v4.2 (MSDOS 6.21 SYSINIT)
                                %if 0
                                ; 30/10/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
                                ; (SYSINIT:02A9h)

                                command_line:
                                    db      2,0
                                    db      'P'
                                    db      0
                                    times   124 db 0 ; db 124 dup(0)

                                %endif

                                ; (SYSINIT:0329h)
000002AD 00                     ZERO:       db      0
000002AE 00                     sepchr:     db      0
000002AF 0000                   linecount: dw       0                   ; line count in config.sys
000002B1 20202020200D0A24       showcount: db       '     ',cr,lf,'$'     ; used to convert linecount to ascii.
000002B9 0000                   buffer_linenum: dw 0                ; line count for "buffers=" command if entered.

000002BB FF                     sys_model_byte:     db      0FFh            ; model byte used in sysinit
000002BC 00                     sys_scnd_model_byte: db 0           ; secondary model byte used in sysinit

000002BD 0000                   buf_prev_off:       dw      0

                                        ;IF      NOT NOEXEC
                                ;COMEXE EXEC0 <0,COMMAND_LINE,DEFAULT_DRIVE,ZERO>
                                        ;ENDIF

                                ; 29/12/2023
                                ; 01/05/2018
                                COMEXE:
000002BF 0000                   EXEC0.ENVIRON:      dw      0       ; seg addr of environment
000002C1 [BB4B]                 EXEC0.COM_LINE:     dw      command_line ; pointer to asciz command line
000002C3 0000                               dw      0       ; SYSINIT segment (0473h for MSDOS 6.21 IO.SYS)
                                                            ; SYSINIT segment (0544h for PCDOS 7.1 IBMBIO.COM)
000002C5 [9802]                 EXEC0.5C_FCB:       dw      DEFAULT_DRIVE ; default fcb at 5C
000002C7 0000                               dw      0       ; SYSINIT segment (0473h for MSDOS 6.21 IO.SYS)
000002C9 [AD02]                 EXEC0.6C_FCB:       dw      ZERO    ; default fcb at 6C
000002CB 0000                               dw      0

                                ; variables for install= command.

000002CD 00                     multi_pass_id:      db      0               ; parameter passed to multi_pass
                                                                    ;  indicating the pass number
                                                                    ; 0 - do scan for DOS=HIGH/LOW
                                                                    ; 1 - load device drivers
                                                                    ; 2 - was to load IFS
                                                                    ;       now it is unused
                                                                    ; 3 - do install=
                                                                    ; >3 - nop
000002CE 0000                   install_flag:       dw      0

                                have_install_cmd equ        00000001b       ; config.sys has install= commands
                                has_installed       equ     00000010b       ; sysinit_base installed.

000002D0 0000                   config_size:        dw      0               ; size of config.sys file. set by sysconf.asm
000002D2 00000000               sysinit_base_ptr: dd        0               ; pointer to sysinit_base
000002D6 00000000               sysinit_ptr:        dd      0               ; returning addr. from sysinit_base
000002DA 0000                   checksum:   dw      0               ; used by sum_up
```

```
21821
21822 000002DC 20<rep 14h>             ldexec_fcb: times 20 db 20h ; db 20 dup (' ') ;big enough
21823 000002F0 00                      ldexec_line:       db     0            ;# of parm characters
21824 000002F1 20                      ldexec_start:      db     ' '
21825 000002F2 00<rep 50h>             ldexec_parm:       times 80 db 0  ; db 80 dup (0)
21826
21827                                  ;instexe exec0     <0,ldexec_line,ldexec_fcb,ldexec_fcb>
21828
21829                                  instexe:
21830 00000342 0000                    iexec.environ:     dw     0             ; seg addr of environment
21831 00000344 [F002]                  iexec.ldexec_line: dw     ldexec_line ; pointer to asciz command line
21832 00000346 0000                              dw     0      ; SYSINIT segment (0473h for MSDOS 6.21 IO.SYS)
21833                                                           ; SYSINIT segment (0544h for PCDOS 7.1 IBMBIO.COM)
21834 00000348 [DC02]                  iexec.ldexec_5c_fcb: dw   ldexec_fcb     ; default fcb at 5C
21835 0000034A 0000                              dw     0      ; SYSINIT segment (0473h for MSDOS 6.22 IO.SYS)
21836 0000034C [DC02]                  iexec.ldexec_6c_fcb: dw   ldexec_fcb     ; default fcb at 6C
21837 0000034E 0000                              dw     0
21838
21839                                  ; variables for comment=
21840
21841 00000350 00                      com_level:  db     0              ; level of " " in command line
21842 00000351 00                      cmmt:              db     0              ; length of comment string token
21843 00000352 00                      cmmt1:             db     0              ; token
21844 00000353 00                      cmmt2:             db     0              ; token
21845 00000354 00                      cmd_indicator:     db     0
21846 00000355 00                      donotshownum:      db     0
21847
21848 00000356 0000                    count:             dw     0
21849 00000358 0000                    org_count:  dw     0
21850 0000035A 0000                    chrptr:            dw     0
21851 0000035C 0000                    cntryfilehandle: dw       0
21852 0000035E 0000                    old_area:   dw     0
21853 00000360 0000                    impossible_owner_size: dw 0              ; paragraph
21854
21855                                  bucketptr: ; label dword
21856                                  bufptr:        ; label dword             ; leave this stuff in order!
21857 00000362 0000                    memlo:     dw     0
21858                                  prmblk:        ; label word
21859 00000364 0000                    memhi:     dw     0
21860 00000366 0000                    ldoff:     dw     0
21861 00000368 0000                    area:      dw     0
21862
21863                                  ; 29/12/2023 - PCDOS 7.1 IBMBIO.COM - SYSINIT:036Ah
21864 0000036A 0000                    prev_memhi: dw 0
21865 0000036C 0000                    prev_alloclim:     dw 0
21866 0000036E 00                      dosdata_umb:       db 0
21867
21868                                  ; Following is the request packet used to call INIT routines for
21869                                  ; all device drivers. Some fields may be accessed individually in
21870                                  ; the code, and hence have individual labels, but they should not
21871                                  ; be separated.
21872
21873 0000036F 19                      packet:    db     25                    ; PCDOS 7.1 IBMBIO.COM
21874                                      ;db    24                    ; was 22
21875 00000370 00                          db     0
21876 00000371 00                          db     0                     ; initialize code
21877 00000372 0000                        dw     0
21878 00000374 00<rep 8h>                  times  8 db 0  ; db 8 dup (?)
21879
21880 0000037C 00                      unitcount:  db     0
21881 0000037D 00000000                break_addr: dd     0
21882 00000381 00000000                bpb_addr:   dd     0
21883                                  drivenumber:       ; 22/10/2022
21884 00000385 00                      devdrivenum:       db     0
21885 00000386 0000                    configmsgflag:     dw     0  ; used to control "error in config.sys line #" message
21886
21887                                  ; end of request packet
21888
21889                                  ;drivenumber:      db     0  ; 22/03/2019
21890
21891                                  toomanydrivesflag:
21892 00000388 00                              db     0  ; >24 fixed disk partitions flag ; M029
21893 00000389 90                      align 2
21894
21895                                  BCodeSeg:  ; 21/10/2022
21896 0000038A 0203                        dw     DOSBIOCODESEG ; (02C7h for MSDOS 5.0 IO.SYS)
21897                                                  ; 0364h for PCDOS 7.1 IBMBIO.COM - 29/12/2023
21898                                      ;dw    Bios_Code  ; = KERNEL_SEGMENT = 0070h (for Retro DOS v4.0)
21899                                                     ; BCodeSeg = 2CCh (for MSDOS 6.21 IO.SYS)
21900
21901                                  ; 30/12/2022
21902                                  ; MSDOS 6.21 IO.SYS, SYSINIT:0387h
21903                                  ;
21904                                  ; Magicbackdoor: dd 0
21905                                  ; NullBackdoor:
21906                                  ;          retf
21907
21908                                  ; 02/11/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 IO.SYS)
21909                                  ; 01/01/2023 - Retro DOS v4.2 (Modified MSDOS 6.21 IO.SYS)
21910                                  ; 19/04/2019
21911                                  _timer_lw_:
21912 0000038C 0000                        dw     0  ; MSDOS 6.21 IO.SYS - SYSINIT:038Ch
21913
21914                                  ; 29/12/2023 - Retro DOS v5.0
21915                                  ; PCDOS 7.1 IBMBIO.COM - SYSINIT:038Eh
21916
21917 0000038E 00                      F5_key:    db 0
21918 0000038F 00                      F8_key:    db 0
21919 00000390 00000000                MagicBackdoor:     dd 0
21920                                  NullBackdoor:
21921 00000394 CB                              retf
21922
21923                                  ;SR;
21924                                  ; This is the communication block between the DOS and the BIOS. It starts at
21925                                  ;the SysinitPresent flag. Any other data that needs to be communicated
21926                                  ;to the DOS should be added after SysinitPresent. The pointer to this block
21927                                  ;is passed to DOS as part of the DOSINIT call.
21928                                  ;
21929
21930                                  BiosComBlock:
21931                                      ;dd    Bios_Data:SysinitPresent
21932                                             ; 0070h:08FDh for MSDOS 6.21 IO.SYS
21933 00000395 [DD07]                    dw     SysinitPresent  ; (retrodos4.sys, offset: ****h)
21934                                      ;dw    KERNEL_SEGMENT ; 0070h
21935                                      ; 21/10/2022
21936 00000397 7000                      dw     DOSBIODATASEG ; 0070h
21937
21938                                  ;align 2
21939
21940                                      ; 22/10/2022 - (MSDOS 5.0 IO.SYS, SYSINIT:0406h)
21941                                      ; 30/12/2022 - (MSDOS 6.21 IO.SYS, SYSINIT:0392h)
21942                                  tempstack:
21943 00000399 00<rep 80h>               times  128 db 0  ; db 80h dup (?)
21944
```

```
                                ; ----------------------------------------------------------------------------

                                    ; 29/12/2023 - Retro DOS v5.0
                                    ; 22/10/2022 - Retro DOS v4.0
                                    ;       ; (MSDOS 5.0 IO.SYS, SYSINIT:0486h)
                                GOINIT:             ; (MSDOS 6.22 IO.SYS, SYSINIT:0412h)
                                    ; (PCDOS 7.1 IBMBIO.COM, SYSINIT:0419h)
                                    ; 12/12/2023
00000419 0E                         push    cs
0000041A 1F                         pop     ds

                                    ; 12/12/2022
                                    ; 22/03/2019 - Retro DOS v4.0
                                    ; 06/07/2018
                                    ; 04/06/2018 - Retro DOS v3.0
                                ; before doing anything else, let's set the model byte
0000041B B4C0                       mov     ah,0C0h                 ; get system configuration
0000041D CD15                       int     15h                     ;
0000041F 7214                       jc      short no_rom_config

                                    ;cmp    ah,0                    ; double check
                                    ;jne    short no_rom_config
                                    ; 03/09/2023
00000421 08E4                       or      ah,ah
00000423 7510                       jnz     short no_rom_config

                                    ; 12/12/2023 ; *
                                    ; ds = cs

00000425 268A4702                   mov     al,[es:bx+ROMBIOS_DESC.bios_sd_modelbyte]
                                    ;mov    [cs:sys_model_byte],al
00000429 A2[BB02]                   mov     [sys_model_byte],al ; *
0000042C 268A4703                   mov     al,[es:bx+ROMBIOS_DESC.bios_sd_scnd_modelbyte]
                                    ;mov    [cs:sys_scnd_model_byte],al
00000430 A2[BC02]                   mov     [sys_scnd_model_byte],al ; *
                                    ;jmp    short SYSIN
                                    ; 02/11/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
00000433 EB29                       jmp     short move_myself

                                no_rom_config:                          ; Old ROM
                                    ; 12/12/2023
                                    ;mov    ax,0F000h
                                    ;mov    ds,ax
                                    ;mov    al,[0FFFEh]
                                    ;mov    [cs:sys_model_byte],al; set the model byte.
                                    ; 12/12/2023
                                    ; ds = cs
00000435 B800F0                     mov     ax,0F000h
00000438 8EC0                       mov     es,ax
0000043A 26A0FEFF                   mov     al,[es:0FFFEh]
0000043E A2[BB02]                   mov     [sys_model_byte],al   ; set the model byte.

                                ; set fake_floppy_drv if there is no diskette drives in this machine.
                                ; execute the equipment determination interrupt and then
                                ; check the returned value to see if we have any floppy drives
                                ; if we have no floppy drive we set cs:fake_floppy_drv to 1
                                ; see the at tech ref bios listings for help on the equipment
                                ; flag interrupt (11h)

                                    ; 22/10/2022
                                ;check_for_fake_floppy:                 ; entry point for rom_config above
00000441 CD11                       int     11h                     ; check equipment flag

                                    ; 29/12/2023 - Retro DOS v5.0
                                    ;jmp    short check_for_fake_floppy
                                    ; PCDOS 7.1 IBMBIO.COM - SYSINIT:0446h
                                    ;db     52h     ; 'RPS' sign
                                    ;db     50h
                                    ;db     53h

                                check_for_fake_floppy:
                                    ; 29/12/2023
                                    ; PCDOS 7.1 IBMBIO.COM - SYSINIT:0449h
                                    ;or     ax, 1           ; (nonsense! this may be overwritten/disabled
                                    ;                       ; by using 'RPS' sign position)
                                    ;                       ;               03/07/2023 - Erdogan Tan
                                    ;test   ax, 1           ; have any floppies?

                                    ; 12/12/2022
00000443 A801                       test    al,1
                                    ;test   ax,1                    ; have any floppies?
00000445 7517                       jnz     short move_myself       ; yes,normal system

                                ; Some ROM BIOSs lie that there are no floppy drives. Lets find out
                                ; whether it is an old ROM BIOS or a new one
                                ;
                                ; WARNING !!!
                                ;
                                ; This sequence of code is present in MSINIT.ASM also. Any modification
                                ; here will require an equivalent modification in MSINIT.ASM also

                                    ; 12/12/2023
                                    ;push   es  ; not necessary

00000447 30C9                       xor     cl,cl
00000449 B408                       mov     ah,8                    ; get disk parameters
0000044B B200                       mov     dl,0                    ; of drive 0
0000044D CD13                       int     13h

                                    ;pop    es  ; 12/12/2023

0000044F 720D                       jc      short move_myself       ; if error lets assume that the
                                                                    ;  ROM BIOS lied
                                    ;cmp    cl,0                    ; double check (max sec no cannot be 0)
                                    ;je     short move_myself
                                    ; 03/09/2023
00000451 08C9                       or      cl,cl
00000453 7409                       jz      short move_myself

00000455 08D2                       or      dl,dl                   ; number of flp drvs == 0?
00000457 7505                       jnz     short move_myself       ; no

                                    ;mov    byte [cs:fake_floppy_drv],1 ; set fake flag.
                                    ; 12/12/2023
                                    ; ds = cs
00000459 C606[8B02]01               mov     byte [fake_floppy_drv],1 ; set fake flag.

                                move_myself:
                                    ; 12/12/2023
                                    ;cld    ; not necessary             ; set up move
                                    ;xor    si,si
                                    ;mov    di,si

                                    ; 12/12/2023
```

```
                                    ; ds = cs
                                    ; 12/12/2022
                                    ;push   cs
                                    ;pop    ds
                                    ;mov    cx,[cs:MEMORY_SIZE]
0000045E 8B0E[9402]                 mov     cx,[MEMORY_SIZE] ; 12/12/2022
                                    ; (MSDOS 6.0 - SYSINIT1.ASM - 1991)
                                ;;; if      msver
                                ;   cmp     cx,1            ; 1 means do scan
                                ;   jnz     short noscan
                                ;   mov     cx,2048         ; start scanning at 32k boundary
                                ;   xor     bx,bx
                                ;
                                ;memscan:inc       cx
                                ;   jz      short setend
                                ;   mov     ds,cx
                                ;   mov     al,[bx]
                                ;   not     al
                                ;   mov     [bx],al
                                ;   cmp     al,[bx]
                                ;   not     al
                                ;   mov     [bx],al
                                ;   jz      short memscan
                                ;setend:
                                ;   mov     cs:[memory_size],cx
                                ;;; endif

                                ;noscan:                         ; cx is mem size in para
                                ;;
                                ;;  cas -- a) if we got our memory size from the ROM, we should test it
                                ;;           before we try to run.
                                ;;         b) in any case, we should check for sufficient memory and give
                                ;;           an appropriate error diagnostic if there isn't enough
                                ;
                                ;   push    cs
                                ;   pop     ds
                                ;
                                ;;  cas note:  It would be better to put dos + bios_code BELOW sysinit
                                ;;    that way it would be easier to slide them down home in a minimal
                                ;;    memory system after sysinit.  As it is, you need room to keep
                                ;;    two full non-overlapping copies, since sysinit sits between the
                                ;;    temporary home and the final one.  the problem with doing that
                                ;;    is that sys*.asm are filled with "mov ax,cs, sub ax,11h" type stuff.
                                ;
                                ;   dec     cx                  ; one para for an arena at end of mem
                                ;                               ; in case of UMBs

                                    ; 22/10/2022
                                    ; (MSDOS 5.0 IO.SYS SYSINIT:04DBh)

                                    ; 12/12/2022
                                    ;push   cs
                                    ;pop    ds

00000462 49                         dec     cx

                                ;------ Check if an RPL program is present at TOM and do not tromp over it

00000463 31DB                       xor     bx,bx
00000465 8EC3                       mov     es,bx
                                    ;mov    bx,[es:(2Fh*4)] ; INT 2Fh address (0:0BCh)
                                    ;mov    es,[es:((2Fh*4)+2)] ; INT 2Fh segment (0:0BEh)
                                    ; 29/09/2023
00000467 26C41EBC00                 les     bx,[es:(2Fh*4)]
0000046C 26817F035250               cmp     word [es:bx+3],'RP'
00000472 751B                       jne     short NoRPL
00000474 26807F054C                 cmp     byte [es:bx+5],'L'
00000479 7514                       jne     short NoRPL

0000047B 89CA                       mov     dx,cx               ; get TOM into DX
0000047D 52                         push    dx
0000047E B8064A                     mov     ax,4A06h
                                    ;mov    ax,(multMULT<<8)+multMULTRPLTOM
00000481 CD2F                       int     2Fh                 ; Get new TOM from any RPL
00000483 58                         pop     ax
00000484 89D1                       mov     cx,dx
00000486 39C2                       cmp     dx,ax
00000488 7405                       je      short NoRPL

                                    ; 11/12/2022
                                    ; ds = cs
0000048A 8916[9602]                 mov     [RPLMemTop],dx
                                    ; 02/11/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
                                    ;mov    [cs:RPLMemTop],dx

0000048E 49                         dec     cx
                                NoRPL:
0000048F B8[1054]                   mov     ax,SI_end           ; need this much room for sysinit
                                                                ; (SI_end == sysinit code size)
                                                                ; 03/09/2023
                                                                ; (58A0h for MSDOS 6.21 IO.SYS)
                                                                ; (5B40h for PCDOS 7.1 IBMBIO.COM)
00000492 E80509                     call    off_to_para
00000495 29C1                       sub     cx,ax

                                ; we need to leave room for the DOS and (if not ROMDOS) for the BIOS
                                ; code above sysinit in memory
                                ;
00000497 81E9000B                   sub     cx,DOSSIZE/16 ; (0A00h)     ; leave this much room for DOS
                                                        ; (0B00h) ; (PCDOS 7.1 IBMBIO.COM) -03/09/2023-

0000049B B8701D                     mov     ax,BCODE_END        ; (1A60h for MSDOS 5.0 IO.SYS)
                                                                ; (1A70h for MSDOS 6.21 IO.SYS)
                                                                ; 03/09/2023
                                                                ; (1E00h for PCDOS 7.1 IBMBIO.COM)
0000049E E8F908                     call    off_to_para         ; leave this much room for BIOS code
000004A1 29C1                       sub     cx,ax
000004A3 8EC1                       mov     es,cx               ; segment where sysinit will be located

                                    ; 12/12/2023
000004A5 FC                         cld     ; not necessary             ; set up move
000004A6 31F6                       xor     si,si
000004A8 89F7                       mov     di,si

000004AA B9[1054]                   mov     cx,SI_end           ; (sysinit code size)
000004AD D1E9                       shr     cx,1                ; divide by 2 to get words
000004AF F3A5                       rep     movsw               ; relocate sysinit

000004B1 06                         push    es                  ; push relocated segment
000004B2 B8[B704]                   mov     ax,SYSIN
000004B5 50                         push    ax                  ; push relocated entry point
```

```
                                    retf                            ; far jump to relocated sysinit

                                ; =============== S U B R O U T I N E =======================================

                                ; 30/12/2023
                                ; PCDOS 7.1 IBMBIO.COM - SYSINIT:04CEh
                                %if 0
                                get_cpu_type:
                                    pushf
                                    push    bx
                                    xor     bx,bx
                                    xor     ax,ax
                                    push    ax
                                    popf
                                    pushf
                                    pop     ax
                                    and     ax,0F000h
                                    cmp     ax,0F000h
                                    jz      short cpu_8086
                                    mov     ax,0F000h
                                    push    ax
                                    popf
                                    pushf
                                    pop     ax
                                    and     ax,0F000h
                                    jz      short cpu_286
                                cpu_386:
                                    inc     bx
                                cpu_286:
                                    inc     bx
                                cpu_8086:
                                    mov     ax,bx
                                    pop     bx
                                    popf
                                    retn
                                %endif

                                ; ---------------------------------------------------------------------------

                                ;   MOVE THE DOS TO ITS PROPER LOCATION

                                    ; 22/10/2022 - Retro DOS 4.0 (Modified MSDOS 5.0 IO.SYS)
                                    ; (SYSINIT:0533h)
                                    ; 30/12/2022 - Retro DOS 4.2 (Modified MSDOS 6.21 IO.SYS)
                                    ; (SYSINIT:04BFh)
                                    ; 03/09/2023 - Retro DOS 4.2 (5.0 - Modified PCDOS 7.1 IBMBIO.COM)
                                    ; (SYSINIT:04F3h)
                                SYSIN:
                                    ; Retro DOS 5.0 - 30/12/2023
                                    ; Retro DOS 4.0 - 22/03/2019
                                    ; Retro DOS 2.0 - 25/02/2018

                                    ; 23/04/2019
                                    ;;mov   ax,Bios_Data
                                    ;mov    ax,KERNEL_SEGMENT ; 0070h
                                    ; 21/10/2022
000004B7 B87000                     mov     ax,DOSBIODATASEG ; 0070h
000004BA 8ED8                       mov     ds,ax

                                    ; 30/12/2023 - Retro DOS v5.0
                                    ;;;
                                    ;push   es
                                    ;push   ax              ; not needed (*) E.TAN - 03/07/2023
                                    ;push   di

                                    ;call   get_cpu_type    ; determine if 386 system
                                    ;
                                get_cpu_type:
000004BC 9C                         pushf
000004BD 31C0                       xor     ax,ax
000004BF 50                         push    ax
000004C0 9D                         popf
000004C1 9C                         pushf
000004C2 58                         pop     ax
000004C3 2500F0                     and     ax,0F000h
000004C6 3D00F0                     cmp     ax,0F000h
000004C9 740F                       jz      short cpu_8086
000004CB B800F0                     mov     ax,0F000h
000004CE 50                         push    ax
000004CF 9D                         popf
000004D0 9C                         pushf
000004D1 58                         pop     ax
000004D2 2500F0                     and     ax,0F000h
000004D5 7402                       jz      short cpu_286
                                cpu_386:
000004D7 29C0                       sub     ax,ax
                                cpu_286:
000004D9 40                         inc     ax
                                cpu_8086:   ; ax = 0
                                    ; 30/12/2023 - Retro DOS v5.0
000004DA 2EA2[B606]                 mov     [cs:cpu_type],al ; 07/04/2024
000004DE 9D                         popf
                                    ;
                                    ;cmp    ax,2            ; 0 = 8086, 1 = 286, 2 = 386
000004DF 3C02                       cmp     al,2
000004E1 7512                       jnz     short not_386_system
000004E3 FC                         cld                     ; 80386
000004E4 1E                         push    ds
000004E5 07                         pop     es              ; change A20 line on/off check code
000004E6 BF[4D07]                   mov     di,cpu386_cmpsd
000004E9 B8B904                     mov     ax,04B9h        ; mov cx,4 ; B90400
000004EC AB                         stosw
000004ED B800F3                     mov     ax,0F300h       ; repz  ; F3
000004F0 AB                         stosw
000004F1 B866A7                     mov     ax,0A766h       ; cmpsd ; 66A7
000004F4 AB                         stosw
                                not_386_system:
                                    ;pop    di
                                    ;pop    ax
                                    ;pop    es
                                    ;;;

000004F5 8C0E[DB07]                 mov     [MoveDOSIntoHMA+2],cs ; set seg of routine to move DOS
000004F9 C606[DD07]01               mov     byte [SysinitPresent],1       ; flag that MoveDOSIntoHMA can be called

                                ; first move the MSDOS.SYS image up to a harmless place
                                ; on top of our new sysinitseg

                                    ; 22/10/2022
000004FE B8[1054]                   mov     ax,SI_end               ; how big is sysinitseg?
00000501 E89608                     call    off_to_para
00000504 8CC9                       mov     cx,cs                   ; pick a buffer for msdos above us
00000506 01C8                       add     ax,cx
00000508 8EC0                       mov     es,ax
```

```
22317
22318 0000050A 31F6                      xor     si,si
22319 0000050C 89F7                      mov     di,si
22320
22321 0000050E 2E8E1E[7302]              mov     ds,[cs:CURRENT_DOS_LOCATION] ; where it is (set by msinit)
22322
22323                                    ;mov    ax,cs
22324                                    ;mov    ds,ax
22325
22326                                    ;;;mov  cx,20480  ; MSDOS 6.21 IO.SYS - SYSINIT:04E2h
22327                                    ;;mov   cx,dossize/2 ; MSDOS 6.0
22328                                    ;mov    cx,[DOSSIZE] ; words (not bytes!)  ; Retro DOS v4.0 (3.0, 2.0)
22329                                    ;mov    es,[FINAL_DOS_LOCATION] ; on top of SYSINIT code
22330                                    ;mov    ds,[CURRENT_DOS_LOCATION]
22331
22332                                    ; 22/10/2022
22333 00000513 B90058                   mov     cx,DOSSIZE/2 ; 5000h
22334                                                         ; 03/09/2023
22335                                                         ; 5800h (PCDOS 7.1)
22336 00000516 F3A5                      rep     movsw
22337 00000518 2E8C06[7302]              mov     [cs:CURRENT_DOS_LOCATION],es
22338
22339                                  ; The DOS code is ORGed at a non-zero value to allow it to be located in
22340                                  ; HIMEM. Thus, the DOS segment location must be adjusted accordingly.
22341                                  ; If this is ROMDOS, however, only the init code is loaded into RAM, so
22342                                  ; this ORG is not done. The entry point is at offset zero in the segment.
22343
22344                                      ; 22/04/2019 (MSDOS 6.0 & MSDOS 6.21 kernel address modification)
22345                                      ;mov    ax,cs
22346                                      ; 02/11/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 IO.SYS, SYSINIT)
22347                                      ;mov    ds,ax
22348
22349                                  ; 29/04/2019 - Retro DOS v4.0 ! important MODIFICATION !
22350
22351                                  ;    ; 24/04/2019
22352                                  ;;ifndef ROMDOS
22353                                  ;    mov     ax,[es:3]              ; get offset of dos
22354                                  ;            ; ax = 3DE0h for MSDOS 6.21 kernel (MSDOS.SYS, offset 3)
22355                                  ;    mov     [dosinit],ax           ; that's the entry point offset
22356                                  ;    call    off_to_para            ; subtract this much from segment
22357                                  ;    ; 23/04/2019
22358                                  ;    ;sub    [CURRENT_DOS_LOCATION],ax
22359                                  ;    sub     [FINAL_DOS_LOCATION],ax
22360                                  ;;else
22361                                  ;;   mov     word [dosinit],0       ; entry to init is at zero
22362                                  ;;
22363                                  ;;endif ; ROMDOS
22364
22365                                      ; 29/04/2019 - Retro DOS v4.0 ! important MODIFICATION !
22366                                      ; (! MSDOS6.BIN starts with DOSDATA ! - Retro DOS v4.0 modification)
22367
22368                                      ;mov    ax,[es:0] ; DOSCODE start address = DOSDATA size (= 136Ah)
22369                                      ;                 ; (Valid for Retro DOS v4.0 only!)
22370
22371                                      ; 22/10/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 IO.SYS, SYSINIT)
22372                                      ; (SYSINIT:0563h for MSDOS 5.0 IO.SYS SYSINIT)
22373                                      ; 03/09/2023 - Retro DOS 4.2 (Modified MSDOS 6.21 IO.SYS)
22374                                      ; (SYSINIT:04ECh for MSDOS 6.21 IO.SYS SYSINIT)
22375                                      ; (SYSINIT:0540h for PCDOS 7.1 IBMBIO.COM SYSINIT)
22376 0000051D A10300                    mov     ax,[3]          ; mov ax, word ptr ds:3
22377                                                            ; 30/12/2023
22378                                                            ; ax = 3F10h for IBMDOS 7.1 kernel
22379                                                            ;  (IBMDOS.SYS, offset 3)
22380
22381 00000520 2EA3[7102]                mov     [cs:dosinit],ax ; (SYSINIT:0563h for MSDOS 5.0 IO.SYS SYSINIT)
22382                                    ; 02/11/2022
22383 00000524 E87308                    call    off_to_para            ; subtract this much from segment
22384 00000527 2E2906[7302]              sub     [cs:CURRENT_DOS_LOCATION],ax
22385
22386                                      ; Current DOSCODE start address = dword [dosinit]
22387
22388                                  ;; If this is not ROMDOS, then the BIOS code is moved to the top of memory
22389                                  ;; until it is determined whether it will be running in HIMEM or not.
22390
22391                                  ;ifndef ROMDOS
22392
22393                                  ; now put Bios_Code up on top of that. Assume Bios_Code + dossize < 64k
22394
22395                                      ; 22/10/2022
22396 0000052C 8CC0                      mov     ax,es
22397 0000052E 05000B                    add     ax,DOSSIZE/16          ; get paragraph of end of dos
22398 00000531 8EC0                      mov     es,ax
22399 00000533 2E8706[8902]              xchg    ax,[cs:temp_bcode_seg]; swap with original home of Bios_Code
22400 00000538 8ED8                      mov     ds,ax                  ; point to loaded image of Bios_Code
22401
22402                                    ;mov    si,BCODE_START ; mov si,30h
22403                                    ; 09/12/2022
22404 0000053A BE[3000]                  mov     si,BCODESTART
22405                                    ; 02/11/2022
22406 0000053D 89F7                      mov     di,si
22407                                    ; 30/12/2023
22408                                    ;mov    cx,1E00h       ; BCODE_END = (SYSINITSEG-DOSBIOCODESEG)*16
22409                                    ;                      ; (544h-364h)*10h = 1E00h (for PCDOS 7.1 IBMBIO.COM)
22410                                    ;mov    cx,BCODE_END   ; mov cx,1A60h ; mov cx,1A70h ; 30/12/2022
22411                                    ;sub    cx,si
22412                                    ; 31/03/2024
22413                                    BCODESIZE equ BCODEEND-BCODESTART
22414 0000053F B9401D                    mov     cx,BCODESIZE
22415 00000542 D1E9                      shr     cx,1
22416 00000544 F3A5                      rep     movsw                  ; move Bios_Code into place
22417
22418 00000546 8CC0                      mov     ax,es                  ; tell it what segment it's in
22419 00000548 2EFF1E[8702]              call    far [cs:seg_reinit_ptr]      ; far call to seg_reinit in Bios_Code (M022)
22420
22421                                  ;endif      ; not ROMDOS
22422
22423                                  ; now call dosinit while it's in its temporary home
22424
22425                                      ;mov    ax,cs
22426                                      ;mov    ds,ax
22427
22428                                      ;mov    dx,[MEMORY_SIZE]       ; set for call to dosinit
22429
22430                                      ; 22/10/2022
22431
22432 0000054D 2EC43E[9503]               les     di,[cs:BiosComBlock]  ; ptr to BIOS communication block
22433                                                                  ; es = KERNEL_SEGMENT (70h), di = 'SysInitPresent' address
22434 00000552 2EC536[7502]              lds     si,[cs:DEVICE_LIST]   ; set for call to dosinit
22435                                                                  ; ds = KERNEL_SEGMENT (70h), si = 'res_dev_list' address
22436
22437 00000557 2E8B16[9402]              mov     dx,[cs:MEMORY_SIZE]   ; set for call to dosinit
22438
22439 0000055C FA                        cli
22440 0000055D 8CC8                      mov     ax,cs
```

```
                        mov     ss,ax

                    ; 30/12/2023 - Retro DOS v5.0 (PCDOS 7.1 IBMBIO.COM)
                    ; 22/03/2019 - Retro DOS v4.0 (MSDOS 6.0, SYSINIT1.ASM)
                    %define locstack ($ - SYSINIT$) & 0FFFEh  ; 532h in MSDOS 6.21 IO.SYS
                                                               ; 5A6h in MSDOS 5.0 IO.SYS SYSINIT
                                                               ; 586h in PCDOS 7.1 IBMBIO.COM SYSINIT
                    ;SYSINIT:0532h:

                    ; 22/10/2022
                    ; ---------------------------------------------------------------------------
                    ;SYSINIT:05A6h:
                    ;locstack:  ; (at SYSINIT:05A6h for MSDOS 5.0 IO.SYS)

                    ; 03/09/2023
                    ; (locstack at SYSINIT:0586h in PCDOS 7.1 IBMBIO.COM SYSINIT)

                        ;mov    sp,05A6h
00000561 BC6005         mov     sp,locstack           ; set stack

00000564 FB             sti

                    ;align 2
                        ; 30/03/2018
                    ;LOCSTACK:
                            ;CALL      FAR [CS:MSDOS] ; FINAL_DOS_LOCATION:0
                                       ;('jmp DOSINIT' in 'MSHEAD.ASM')
                                       ;('DOSINIT:' is in 'MSINIT.ASM')

                        ; 22/10/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 IO.SYS)
                        ; 22/03/2019 - Retro DOS v4.0 (MSDOS 6.0, 6.21)

                    ; This call to DOSINIT will relocate the DOS data from its present location
                    ; at the top of memory, to its final location in low memory just above the
                    ; BIOS data. It will then build important DOS data structures in low
                    ; memory following the DOS data. It returns (among many other things) the
                    ; new starting address of free memory.

00000565 2EFF1E[7102]   call    far [cs:dosinit]      ; call dosinit
                                ; es:di -> sysinitvars_ext

0000056A 2E8C1E[8502]   mov     [cs:def_php],ds               ; save pointer to PSP

                        ; 11/12/2022
                        ; 22/03/2019
0000056F 0E             push    cs
00000570 1F             pop     ds
                        ; 22/10/2022
00000571 A3[8302]       mov     [hi_doscod_size],ax
00000574 890E[8102]     mov     [lo_doscod_size],cx
00000578 8916[7D02]     mov     [dos_segreinit],dx

                        ; 11/12/2022
                        ; ds = cs
                        ;mov    [cs:hi_doscod_size],ax; size of doscode (including exepatch)
                        ;mov    [cs:lo_doscod_size],cx; (not including exepatch)
                        ;mov    [cs:dos_segreinit],dx ; save offset of segreinit

                        ; 05/06/2018 - Retro DOS v3.0
                        ; ES:DI = Address of pointer to SYSINITVARS structure (MSDOS 3.3)

                        ; 11/12/2022
                        ; ds = cs
                        ; 22/10/2022
                        ;mov    ax,[es:di+SysInitVars_Ext.SYSI_InitVars] ; 5/29/86
0000057C 268B05         mov     ax,[es:di] ; 22/03/2019
                        ;mov    [cs:DOSINFO],ax
0000057F A3[6D02]       mov     [DOSINFO],ax
                        ;mov    ax,[es:di+SysInitVars_Ext.SYSI_InitVars+2]
00000582 268B4502       mov     ax,[es:di+2]
                        ;mov    [cs:DOSINFO+2],ax
00000586 A3[6F02]       mov     [DOSINFO+2],ax ; set the sysvar pointer

                        ;mov    ax,[es:di+SysInitVars_Ext.SYSI_Country_Tab]
00000589 268B4504       mov     ax,[es:di+4]
                        ;mov    [cs:sysi_country],ax
0000058D A3[7902]       mov     [sysi_country],ax
                        ;mov    ax,[es:di+SysInitVars_Ext.SYSI_Country_Tab+2]
00000590 268B4506       mov     ax,[es:di+6]
                        ;mov    [cs:sysi_country+2],ax
00000594 A3[7B02]       mov     [sysi_country+2],ax   ; set the SYSI_Country pointer

                        ; 20/04/2019
                        ;mov    ax,[CURRENT_DOS_LOCATION]
                        ;;mov   es,[CURRENT_DOS_LOCATION]
                        ;mov    ax,[FINAL_DOS_LOCATION] ; give dos its temporary location
                        ; 22/10/2022
                        ;mov    ax,[cs:CURRENT_DOS_LOCATION]
                        ;;;mov  [dos_segreinit+2],es
                        ;;mov   [dos_segreinit+2],ax
                        ;mov    [cs:dos_segreinit+2],ax
                        ; 11/12/2022
                        ; ds = cs
00000597 8E06[7302]     mov     es,[CURRENT_DOS_LOCATION]
0000059B 8C06[7F02]     mov     [dos_segreinit+2],es
                        ; 02/11/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
                        ;mov    es,[cs:CURRENT_DOS_LOCATION]
                        ;mov    [cs:dos_segreinit+2],es

                    ; ---------------------------------------------------------------------------

                    ;SYSINIT:0577h:
                        ; ... RPLArena ... MSDOS 6.21 IO.SYS (SYSINIT:0577h to SYSINIT:05D1h)
                    ;SYSINIT:05D1h:    ; NoRPLArena

                        ; 22/10/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 IO.SYS SYSINIT)
                    ;------ Cover up RPL code with an arena
                    ;SYSINIT:05EBh:
                        ; 11/12/2022
                        ; ds = cs
0000059F 31DB           xor     bx,bx
000005A1 391E[9602]     cmp     [RPLMemTop],bx ; 0
                        ;cmp    word [RPLMemTop],0
                        ;;cmp   word [cs:RPLMemTop],0
000005A5 7450           je      short NoRPLArena

                    ;------ alloc all memory

                        ; 11/12/2022
                        ;mov    bx,0FFFFh
000005A7 4B             dec     bx
                        ; bx = 0FFFFh
000005A8 B448           mov     ah,48h
000005AA CD21           int     21h
```

```
                                                ; DOS - 2+ - ALLOCATE MEMORY
                                                ; BX = number of 16-byte paragraphs desired
000005AC B448                           mov     ah,48h
000005AE CD21                           int     21h

000005B0 8EC0                           mov     es,ax                   ; get it into ES and save it
000005B2 06                             push    es

                                ;------ resize upto RPL mem

                                        ; 11/12/2022
                                        ; ds = cs
                                        ;sub    ax,[cs:RPLMemTop]
000005B3 2B06[9602]                     sub     ax,[RPLMemTop]
000005B7 F7D8                           neg     ax
000005B9 48                             dec     ax
000005BA 89C3                           mov     bx,ax
000005BC B44A                           mov     ah,4Ah
000005BE CD21                           int     21h
                                                ; DOS - 2+ - ADJUST MEMORY BLOCK SIZE (SETBLOCK)
                                                ; ES = segment address of block to change
                                                ; BX = new size in paragraphs

                                ;------ allocate the free (RPL MEM)

000005C0 BBFFFF                         mov     bx,0FFFFh
000005C3 B448                           mov     ah,48h
000005C5 CD21                           int     21h
000005C7 B448                           mov     ah,48h
000005C9 CD21                           int     21h

                                ;----- mark that it belongs to RPL

000005CB 48                             dec     ax
000005CC 8EC0                           mov     es,ax
                                        ;mov    word [es:arena_owner],8
000005CE 26C70601000800                 mov     word [es:1],8
                                        ;mov    word [es:arena_name],'RP'
000005D5 26C70608005250                 mov     word [es:8],'RP'
                                        ;mov    word [es:arena_name+2],'L'
000005DC 26C7060A004C00                 mov     word [es:10],'L'
                                        ;mov    word [es:arena_name+4],0
000005E3 26C7060C000000                 mov     word [es:12],0
                                        ;mov    word [es:arena_name+6],0
000005EA 26C7060E000000                 mov     word [es:14],0

000005F1 07                                 pop     es                      ; get back ptr to first block
000005F2 B449                               mov     ah,49h    ; Dealloc     ; and free it
000005F4 CD21                           int     21h
                                                ; DOS - 2+ - FREE MEMORY
                                                ; ES = segment address of area to be freed
                                        ; 11/12/2022
000005F6 F8                             clc

                                ; ------------------------------------------------------------------------

                                NoRPLArena:
                                        ; 11/12/2022
                                        ; ds = cs
                                        ; 22/03/2019 - Retro DOS v4.0 (MSDOS 6.0, 6.21, IO.SYS)
000005F7 C43E[6D02]                     les     di,[DOSINFO]    ; es:di -> dosinfo
                                        ; 22/10/2022 - Retro DOS v4.0 (MSDOS 5.0 IO.SYS SYSINIT)
                                        ;les    di,[cs:DOSINFO]         ; es:di -> dosinfo

                                        ; 11/12/2022
                                        ;clc                            ; get the extended memory size

                                ;   execute the get extended memory size subfunction in the bios int 15h
                                ;   if the function reports an error do nothing else store the extended
                                ;   memory size reported at the appropriate location in the dosinfo buffer
                                ;   currently pointed to by es:di. use the offsets specified in the
                                ;   definition of the sysinitvars struct in inc\sysvar.inc

000005FB B488                           mov     ah,88h
000005FD CD15                           int     15h                     ; check extended memory size
000005FF 720B                           jc      short no_ext_memory
                                                ; Get Extended Memory Size
                                                ; Return: CF clear on success
                                                ; AX = size of memory above 1M in K
                                        ;mov    [es:di+SYSI_EXT_MEM],ax ; save extended memory size
                                        ; 22/10/2022
00000601 26894545                       mov     [es:di+45h],ax ; save extended memory size
00000605 09C0                           or      ax,ax
00000607 7403                           jz      short no_ext_memory
00000609 E8F006                         call    ClrVDISKHeader
                                no_ext_memory:
                                        ;mov    ax,[es:di+SYSI_MAXSEC]; get the sector size
0000060C 268B4510                       mov     ax,[es:di+10h]
                                        ;add    ax,bufinsiz
                                        ; 30/12/2023 - Retro DOS v5.0
                                        ;add    ax,20           ; size of buffer header
00000610 83C018                         add     ax,24           ; bufinsiz
                                                ; size of buffer header = 24 (PCDOS v7.1 IBMBIO.COM)
                                                ; (it was 20 in MSDOS 6.22 IO.SYS)
                                        ; 11/12/2022
                                        ; ds = cs
00000613 A3[9D02]                       mov     [singlebuffersize],ax ; total size for a buffer
                                        ;mov    [cs:singlebuffersize],ax
                                        ; 11/12/2022
00000616 A0[9802]                       mov     al,[DEFAULT_DRIVE]      ; get the 1 based boot drive number set by msinit
                                        ;mov    al,[cs:DEFAULT_DRIVE]
                                        ;mov    [es:di+SYSI_BOOT_DRIVE],al ; set sysi_boot_drive
00000619 26884543                       mov     [es:di+43h],al

                                ; determine if 386 system...

                                ; 30/12/2023
                                %if 0
                                        ;get_cpu_type                  ; macro to determine cpu type

                                get_cpu_type:
                                        ; 11/12/2022
                                        pushf
                                        ;push   bx
                                        ;xor    bx,bx
                                        ; 11/12/2022
                                        ;xor    cx,cx
                                        ;
                                        xor     ax,ax
                                        ; ax = 0
                                        push    ax
                                        popf
                                        pushf
                                        pop     ax
```

```
                                    and     ax,0F000h
                                    ;cmp    ax,0F000h
                                    cmp     ah,0F0h
                                    je      short cpu_8086
                                    ;mov    ax,0F000h
                                    mov     ah,0F0h
                                    ; ax = 0F000h
                                    push    ax
                                    popf
                                    pushf
                                    pop     ax
                                    ;and    ax,0F000h
                                    and     ah,0F0h
                                    jz      short cpu_286
                                cpu_386:
                                    ; 11/12/2022
                                    ;;inc   bx
                                    ;inc    cx
                                    ; 11/12/2022
                                    ;mov    byte [es:di+SYSI_DWMOVE],1
                                    mov     byte [es:di+44h],1

                                    ; 03/09/2023 - Retro DOS v5.0 (PCDOS 7.1 Modified SYSINIT)
                                    ; change A20 line on/off check code to the faster (for 32 bit cpu)
                                    ;push   es
                                    ;push   di
                                    ;mov    ax,DOSBIODATASEG ; 0070h
                                    ;mov    es,ax
                                    ;cld
                                    ;mov    di,cpu386_cmpsd ; (IsA20Off)
                                    ;mov    ax,4B9h         ; mov cx,4 ; B90400
                                    ;stosw
                                    ;mov    ax,0F300h       ; repz  ; F3
                                    ;stosw
                                    ;mov    ax,0A766h       ; cmpsd ; 66A7
                                    ;stosw
                                    ;pop    di
                                    ;pop    es

                                cpu_286:
                                    ;;;inc  bx
                                    ;;inc   cx
                                cpu_8086:
                                    ; 11/12/2022
                                    ;;mov   ax,bx
                                    ;pop    bx
                                    popf
                                %endif
                                    ;...

                                    ; 11/12/2022
                                    ;or     cl,cl
                                    ;jz     short not_386_system
                                    ; 11/12/202
                                    ;cmp    cl,2
                                    ;;cmp   ax,2                    ; is it a 386?
                                    ;jne    short not_386_system   ; no: don't mess with flag

                                    ; 30/12/2023 - Retro DOS v5.0
0000061D 803E[B606]02               cmp     byte [cpu_type], 2     ; is it a 386?
00000622 7505                       jne     short _not_386_cpu     ; no: don't mess with flag

                                    ;mov    byte [es:di+SYSI_DWMOVE],1
                                    ; 11/12/2022
                                    ; 22/10/2022
00000624 26C6454401                 mov     byte [es:di+44h],1
                                _not_386_cpu:
                                    ;mov    al,[es:di+SYSI_NUMIO]
00000629 268A4520                   mov     al,[es:di+20h]
                                    ; 11/12/2022
                                    ; ds = cs
0000062D A2[8503]                   mov     [drivenumber],al       ; save start of installable block drvs
                                    ;mov    [cs:drivenumber],al

00000630 8CC8                       mov     ax,cs
00000632 83E811                     sub     ax,11h                 ; room for PSP we will copy shortly
                                    ; 11/12/2022
                                    ;mov    cx,[singlebuffersize] ; temporary single buffer area
                                    ;;mov   cx,[cs:singlebuffersize]
                                    ;shr    cx,1
                                    ;shr    cx,1                   ; divide size by 16...
                                    ;shr    cx,1
                                    ;shr    cx,1                   ; ...to get paragraphs...
                                    ;inc    cx                     ; ... and round up
                                    ; 11/12/2022
00000635 8B1E[9D02]                 mov     bx,[singlebuffersize]
00000639 B104                       mov     cl,4
0000063B D3EB                       shr     bx,cl
0000063D 43                         inc     bx

                                ;   cas note: this unorthodox paragraph rounding scheme wastes a byte
                                ;     if [singlebuffersize] ever happens to be zero mod 16. Could this
                                ;     ever happen? Only if the buffer overhead was zero mod 16, since
                                ;     it is probably safe to assume that the sector size always will be.
                                ;
                                ;    mohans also found a bug in CONFIG.SYS processing where it replaces
                                ;     EOF's with cr,lf's, without checking for collision with [confbot].
                                ;     perhaps the extra byte this code guarantees is what has kept that
                                ;     other code from ever causing a problem???

                                    ; 11/12/2022
0000063E 29D8                       sub     ax,bx
                                    ;sub    ax,cx
00000640 A3[A702]                   mov     [top_of_cdss],ax       ; temp "unsafe" location
                                    ; 22/10/2022
                                    ;mov    [cs:top_of_cdss],ax

                                ;   chuckst -- 25 Jul 92 -- added code here to pre-allocate space
                                ;   for 26 temporary CDSs, which makes it easier to use alloclim
                                ;   for allocating memory for MagicDrv.

                                    ; 30/12/2023
                                    ;push   es ; not necessary (!*)       ; preserve pointer to dosinfo
                                    ;push   di

                                    ; 22/10/2022
                                ;   mov     cx,ax                  ; save pointer for buffer
                                ;
                                ;;  now allocate space for 26 CDSs
                                ;
                                ;   sub     ax,((26 *(curdirlen))+15)/16
                                ;   mov     [ALLOCLIM],ax          ; init top of free memory pointer
                                ;   mov     [CONFBOT],ax           ; init this in case no CONFIG.SYS
```

```
                                    ; 30/12/2022 - Retro DOS v4.2 (MSDOS 6.21 IO.SYS)
                                    ; (SYSINIT:064Ch)
00000643 89C1                       mov     cx,ax ; (*)
00000645 2D8F00                     sub     ax,((26*(curdirlen))+15)/16 ; sub ax,143
00000648 A3[A502]                   mov     [ALLOCLIM],ax           ; init top of free memory pointer
0000064B A3[A302]                   mov     [CONFBOT],ax            ; init this in case no CONFIG.SYS

                                ; setup and initialize the temporary buffer at cx

                                    ;les    di,[es:di+SYSI_BUF]     ; get the buffer chain entry pointer
0000064E 26C47D12                   les     di,[es:di+12h]
                                    ; 11/12/2022
00000652 31DB                       xor     bx,bx
                                    ;xor    ax,ax
                                    ;mov    [es:di+BUFFINF.Dirty_Buff_Count],ax ; 0
                                    ;mov    word [es:di+4],0
00000654 26895D04                   mov     [es:di+4],bx ; 0
                                    ;mov    [es:di+BUFFINF.Buff_Queue],ax ; 0
                                    ;mov    word [es:di],0
00000658 26891D                     mov     [es:di],bx ; 0
                                    ;mov    [es:di+BUFFINF.Buff_Queue+2],cx ; cx = [top_of_cdss] ; 6.21
                                    ;;mov   [es:di+BUFFINF.Buff_Queue+2],ax ; ax = [top_of_cdss] ; 5.0
                                    ;mov    [es:di+2],ax
                                    ;mov    es,ax   ; [top_of_cdss] = [CONFBOT]
                                    ; 30/12/2022 - Retro DOS v4.2 (MSDOS 6.21 IO.SYS, SYSINIT)
0000065B 26894D02                   mov     [es:di+2],cx ; [top_of_cdss] ; (*)
0000065F 8EC1                       mov     es,cx

                                    ; 11/12/2022
                                    ;xor    ax,ax
                                    ;mov    di,ax                   ; es:di -> single buffer
00000661 89DF                       mov     di,bx
                                    ; di = 0

                                    ;mov    [es:di+buffinfo.buf_next],ax ; points to itself
                                    ; 11/12/2022
                                    ;mov    [es:di],ax ; 0
00000663 26891D                     mov     [es:di],bx ; 0
                                    ;mov    [es:di+buffinfo.buf_prev],ax ; points to itself
                                    ; 11/12/2022
                                    ;mov    [es:di+2],ax ; 0
00000666 26895D02                   mov     [es:di+2],bx ; 0

                                    ; 22/10/2022 - Retro DOS 4.0 (Modified MSDOS 5.0 IO.SYS SYINIT)
                                    ; MSDOS 5.0 IO.SYS - SYSINIT:06E0h

                                    ;mov    word [es:di+buffinfo.buf_ID],00FFh ; free buffer,clear flag
0000066A 26C74504FF00               mov     word [es:di+4],00FFh
                                ;SYSINIT:06E6h
                                    ;;mov   [es:di+buffinfo.buf_sector],ax ; 0
                                    ;mov    word [es:di+6],0
                                    ; 11/12/2022
                                    ;mov    [es:di+buffinfo.buf_sector],bx ; 0
00000670 26895D06                   mov     [es:di+6],bx ; 0
                                    ;;mov   [es:di+buffinfo.buf_sector+2],ax ; 0
                                    ;mov    word [es:di+8],0
                                    ; 11/12/2022
                                    ;mov    [es:di+buffinfo.buf_sector+2],bx ; 0
00000674 26895D08                   mov     [es:di+8],bx ; 0

                                    ; 30/12/2023 (!*)
                                    ;pop    di                      ; restore pointer to DOSINFO data
                                    ;pop    es

                                    ; 11/12/2022
                                    ; ds = cs
                                    ; 22/10/2022
                                    ;push   cs
                                    ;pop    ds

00000678 E82807                     call    TempCDS                 ; set up cdss so re_init and sysinit
                                                                    ;  can make disk system calls
                                                                    ; tempcds trashes ds
                                    ; 10/05/2019
0000067B 2E8E1E[8502]               mov     ds,[cs:def_php]                 ; retrieve pointer to PSP returned by DOSINIT

                                    ;if not ibmjapver
                                    ;call   far KERNEL_SEGMENT:re_init ; re-call the bios
                                    ;endif

                                    ; 22/10/2022
                                ;SYSINIT:06FEh:    ; (MSDOS 5.0 IO.SYS, SYSINIT)
                                    ; 30/12/2022
                                ;SYSINIT:0697h:    ; (MSDOS 6.21 IO.SYS, SYSINIT)
                                    ;call   far ptr 70h:89Bh
00000680 9A[2807]7000               call    DOSBIODATASEG:RE_INIT

00000685 FB                         sti                             ; ints ok
00000686 FC                         cld                             ; make sure

                                ; 23/03/2019

                                ;SYSINIT:069Eh     ; 30/12/2022

                                ; dosinit has set up a default "process" (php) at ds:0. we will move it out
                                ; of the way by putting it just below sysinit at end of memory.

00000687 8CCB                       mov     bx,cs
00000689 83EB10                     sub     bx,10h
0000068C 8EC3                       mov     es,bx
0000068E 31F6                       xor     si,si
00000690 89F7                       mov     di,si
00000692 B98000                     mov     cx,128
00000695 F3A5                       rep     movsw

                                    ;mov    [es:PDB.JFN_POINTER+2],es ; Relocate
                                    ; 22/10/2022
00000697 268C063600                 mov     [es:36h],es

                                    ; Set Process Data Block - Program Segment Prefix address
                                    ; BX = PDB/PSP segment
0000069C B450                           mov ah,50h  ; SET_CURRENT_PDB
0000069E CD21                       int     21h                     ; tell DOS we moved it
                                                    ; DOS - 2+ internal - SET PSP SEGMENT
                                                    ; BX = segment address of new PSP
                                    ; 22/10/2022
                                    ; 27/03/2019
                                    ; 30/12/2023
                                    ;push   ds ; */                 ; preserve DS returned by DOSINIT

000006A0 0E                         push    cs
000006A1 1F                         pop     ds

                                    ; set up temp. critical error handler
```

```
000006A2 BA[794A]                   mov     dx,int24                ; set up int 24 handler
                                    ;;mov   ax,(SET_INTERRUPT_VECTOR*256)+24h
                                    ;mov    ax,(SET_INTERRUPT_VECTOR<<8)|24h
000006A5 B82425                     mov     ax,2524h
000006A8 CD21                       int     21h

000006AA 803E[8803]00                   cmp     byte [toomanydrivesflag],0 ; Q: >24 partitions?      M029
000006AF 7406                           je      short no_err                ;  N: continue            M029
000006B1 BA[9E53]                       mov     dx,TooManyDrivesMsg         ;  Y: print error message M029
                                        ; 22/10/2022
                                    ;call   print                   ;                       M029
                                    ; 12/12/2022
000006B4 EB04                       jmp     short p_dosinit_msg ; 23/03/2019 - Retro DOS v4.0

                                    ; 30/12/2023 - Retro DOS v5.0
                                cpu_type:
000006B6 FF                         db 0FFh ; db 0

                                no_err:
                                    ; 12/05/2019
                                    ;-----------------------------------------------
                                    ; 27/06/2018 - Retro DOS v3.0 ; 23/03/2019 - Retro DOS v4.0
                                    ; 22/10/2022 - Retro DOS v4.0
                                    ; 12/12/2022
                                    ; 30/12/2023 - Retro DOS v5.0
000006B7 BA[7D4A]                   mov     dx,BOOTMES              ; Display (fake) MSDOS version message
                                p_dosinit_msg:
000006BA E89743                     call    print                   ; Print message
                                    ;-----------------------------------------------

                                    ; 11/12/2022
                                    ; 22/10/2022
                                    ; 23/03/2019 - Retro DOS v4.0
                                    ;pop    ds                      ; start of free memory
                                    ;mov    dl,[cs:DEFAULT_DRIVE]

                                    ; 11/12/2022
                                    ; 27/03/2019
000006BD 8A16[9802]                 mov     dl,[DEFAULT_DRIVE]
                                    ; 30/12/2023
                                    ;pop    ds ; */

000006C1 08D2                       or      dl,dl
                                    ; 30/12/2023
000006C3 7405                       jz      short nodrvset          ; bios didn't say
                                    ;jz     short ProcessConfig  ; (Retro DOS v4.0 does not contain DBLSPACE code)
                                    ;dec    dl                      ; A = 0
                                    ; 18/12/2022
000006C5 4A                         dec     dx
000006C6 B40E                       mov     ah,0Eh  ; SET_DEFAULT_DRIVE
000006C8 CD21                       int     21h                     ; select the disk
                                                    ; DOS - SELECT DISK
                                                    ; DL = new default drive number (0 = A, 1 = B, etc.)
                                                    ; Return: AL = number of logical drives
                                nodrvset:
                                    ; 04/01/2023 - Retro DOS 4.2 (Modified MSDOS 6.21 IO.SYS SYINIT)
                                    ; (SYSINIT:06DFh)

                                    ; 30/12/2023 - Retro DOS 5.0
                                    ; (PCDOS 7.1 IBMBIO.COM - SYSINIT:0733h)
000006CA 1E                         push    ds
000006CB 29C0                       sub     ax,ax
000006CD 8ED8                       mov     ds,ax ; 0 ; ROM BIOS Data Area
000006CF A16C04                     mov     ax,[46Ch] ; timer tick count (18.2 ticks per second)
                                    ;mov    [cs:_timer_lw_],ax
000006D2 1F                         pop     ds
                                    ; ds = cs
000006D3 A3[8C03]                   mov     [_timer_lw_],ax

                                    ; ---------------------

                                    ;ifdef  dblspace_hooks
                                    ;       ....
                                    ;       ....
                                    ;endif

                                    ; ---------------------

                                    ; 30/12/2023 - Retro DOS 5.0 (Modified MSDOS 7.1 IBMBIO.COM SYS SYINIT)
                                ; ---------------------------------------------------------------------------
                                    ; PCDOS 7.1 IBMBIO.COM - SYSINIT:0740h

000006D6 0E                         push    cs
000006D7 07                         pop     es

                                    ; 07/04/2024
                                    ;mov    word [cs:MagicBackdoor+2],cs
                                    ;mov    word [cs:MagicBackdoor], NullBackdoor
000006D8 8C0E[9203]                 mov     word [MagicBackdoor+2],cs
000006DC C706[9003][9403]           mov     word [MagicBackdoor], NullBackdoor

                                    ; ds = es = cs = SYSINIT segment
                                set_drvspc_size:
000006E2 BE[AB16]                   mov     si,MagicDDName          ; "\DBLSPACE.BIN"
                                set_dblspc_size:
000006E5 E8792F                     call    SizeDevice
000006E8 732B                       jnc     short wait_for_key_2s
                                    ;cmp    byte [cs:si], 'C'
000006EA 803C43                     cmp     byte [si],'C'           ; "C:\STACKER.BIN"
000006ED 740C                       je      short set_drvspc_name
                                    ;cmp    byte [cs:DEFAULT_DRIVE],3
000006EF 803E[9802]03               cmp     byte [DEFAULT_DRIVE],3
000006F4 7405                       je      short set_drvspc_name
000006F6 83EE02                     sub     si,2                    ; "C:\DBLSPACE.BIN"
000006F9 EBEA                       jmp     short set_dblspc_size

                                set_drvspc_name:
                                    ;cmp    byte [cs:MagicDDName+2],'R'   ; "BLSPACE.BIN"
000006FB 803E[AD16]52               cmp     byte [MagicDDName+2],'R'
00000700 7408                       je      short set_stacker_name
                                    ;mov    word [cs:MagicDDName+2],'RV'  ; "DRVSPACE.BIN"
00000702 C706[AD16]5256             mov     word [MagicDDName+2],'RV'
00000708 EBD8                       jmp     short set_drvspc_size

                                set_stacker_name:
0000070A 81FE[B916]                 cmp     si,StackerName          ; "C:\STACKER.BIN"
0000070E 734B                       jnb     short wfk2s_4
00000710 BE[BB16]                   mov     si,StackerName+2        ; "\STACKER.BIN"
00000713 EBD0                       jmp     short set_dblspc_size

                                wait_for_key_2s:
                                    ;mov    [cs:MagicDDNamePtr],si
00000715 8936[A716]                 mov     [MagicDDNamePtr],si
00000719 1E                         push    ds
```

```
23061 0000071A 29C0                     sub     ax,ax
23062 0000071C 8ED8                     mov     ds,ax   ; 0             ; ROMBIOS data area
23063 0000071E 8B166C04                 mov     dx,[46Ch]               ; Counter for Interrupt 1Ah
23064                                   wfk2s_1:
23065 00000722 B401                     mov     ah,1
23066 00000724 CD16                     int     16h             ; KEYBOARD - CHECK BUFFER, DO NOT CLEAR
23067                                                           ; Return: ZF clear if character in buffer
23068                                                           ; AH = scan code, AL = character
23069                                                           ; ZF set if no character in buffer
23070 00000726 7511                     jnz     short wfk2s_2
23071 00000728 B402                     mov     ah,2
23072 0000072A CD16                     int     16h                     ; KEYBOARD - GET SHIFT STATUS
23073                                                                   ; AL = shift status bits
23074 0000072C A803                     test    al,3
23075 0000072E 7509                     jnz     short wfk2s_2
23076 00000730 A16C04                   mov     ax,[46Ch]               ; tick count
23077 00000733 29D0                     sub     ax,dx
23078 00000735 3C25                         cmp al,37                   ; 2 seconds
23079 00000737 72E9                     jb      short wfk2s_1           ; wait for user's key press
23080                                   wfk2s_2:
23081 00000739 1F                       pop     ds                      ; read/check the pressed key
23082 0000073A 29DB                     sub     bx,bx   ; bx = 0
23083 0000073C B402                     mov     ah,2
23084 0000073E CD16                     int     16h                     ; KEYBOARD - GET SHIFT STATUS
23085                                                                   ; AL = shift status bits
23086 00000740 A803                     test    al,3                    ; Left or Right SHIFT key pressed ?
23087 00000742 7402                     jz      short wfk2s_3           ; no
23088 00000744 43                       inc     bx
23089 00000745 43                       inc     bx      ; bx = 2
23090                                   wfk2s_3:
23091 00000746 B401                     mov     ah,1
23092 00000748 CD16                     int     16h             ; KEYBOARD - CHECK BUFFER, DO NOT CLEAR
23093                                                           ; Return: ZF clear if character in buffer
23094                                                           ; AH = scan code, AL = character
23095                                                           ; ZF set if no character in buffer
23096 0000074A 7418                     jz      short wfk2s_6
23097 0000074C 80FC65                   cmp     ah,65h                  ; F8 key pressed ?
23098 0000074F 740C                     jz      short wfk2s_5
23099 00000751 80FC62                   cmp     ah,62h                  ; F5 key pressed ?
23100 00000754 750E                     jnz     short wfk2s_6
23101                                   ;mov    byte [cs:F5_key],1
23102 00000756 C606[8E03]01             mov     byte [F5_key],1
23103                                   wfk2s_4:
23104 0000075B EB49                     jmp     short ProcessConfig   ; continue (as normal/default state)
23105
23106                                   wfk2s_5:
23107                                   ;mov    byte [cs:F8_key],1
23108 0000075D C606[8F03]01             mov     byte [F8_key],1
23109 00000762 EB42                     jmp     short ProcessConfig
23110
23111                                   wfk2s_6:
23112 00000764 E8AA02                   call    AllocFreeMem            ; get the largest free block from DOS
23113 00000767 E8700F                   call    MagicPreload            ; **** PRE-LOAD MAGICDRV!!! ****
23114
23115                                   ; 07/04/2024 - Retro DOS v5.0
23116                                   ; (DS may not be same with CS here!)
23117 0000076A 0E                       push    cs
23118 0000076B 1F                       pop     ds ; *
23119 0000076C 8E06[6803]               mov     es,[area]
23120
23121 00000770 09C0                     or      ax,ax                   ; error?
23122 00000772 7406                     jz      short wfk2s_7
23123                                   PreloadFailed:
23124 00000774 B449                     mov     ah,49h                  ; Dealloc ; free the block if no load
23125                                   ;;mov   es,[cs:area]
23126                                   ;mov    es,[area]
23127 00000776 CD21                     int     21h             ; DOS - 2+ - FREE MEMORY
23128                                                           ; ES = segment address of area to be freed
23129 00000778 EB2C                     jmp     short ProcessConfig
23130
23131                                   wfk2s_7:
23132                                   ;mov    bx,[cs:memhi]
23133                                   ;mov    es,[cs:area]
23134                                   ;sub    bx,[cs:area]            ; get desired block size in paras
23135                                   ; 07/04/2024 - Retro DOS v5.0
23136                                   ; ds = cs ; *
23137 0000077A 8CC3                     mov     bx,es
23138 0000077C F7DB                     neg     bx              ; bx = - [cs:area]
23139 0000077E 031E[6403]               add     bx,[memhi]      ; bx = [cs:memhi] - [cs:area]
23140
23141 00000782 B44A                     mov     ah,4Ah
23142 00000784 CD21                     int     21h             ; DOS - 2+ - ADJUST MEMORY BLOCK SIZE (SETBLOCK)
23143                                                           ; ES = segment address of block to change
23144                                                           ; BX = new size in paragraphs
23145 00000786 8CC0                     mov     ax,es
23146 00000788 48                       dec     ax
23147 00000789 8EC0                     mov     es,ax                   ; get Magicdrv arena
23148
23149 0000078B 26C70601000800               mov     word [es:1],8  ; [es:arena_owner]
23150                                       ;mov    word [es:ARENA.OWNER],8       ; set impossible owner
23151 00000792 26C70608005344               mov     word [es:8],4453h ; [es:arena_name],'SD' ; System Data
23152                                       ;mov    word [es:ARENA.NAME],'SD' ; 4453h
23153 00000799 2603060300                       add ax,[es:3]               ; get MCB length
23154                                       ;add    ax,[es:ARENA.SIZE]
23155
23156                                       ;lds    si,[cs:DOSINFO]                 ; get to arena header
23157 0000079E C536[6D02]                   lds     si,[DOSINFO]
23158 000007A2 40                               inc     ax                  ; get addr of next MCB
23159 000007A3 8944FE                           mov     [si-2], ax          ; store that
23160
23161                                   ; ----------------------------------------------------------------------------
23162
23163                                   ; MSDOS 6.21 IO.SYS, SYSINIT:0744h
23164
23165                                   ; 23/03/2019 - Retro DOS v4.0 (MSDOS 6.0, SYSINIT1.ASM, 1991)
23166                                   ; ----------------------------------------------------------------------------
23167                                   ; 22/10/2022 - Retro DOS v4.0 (MSDOS 5.0 IO.SYS SYSINIT)
23168                                   ; ----------------------------------------------------------------------------
23169                                   ; 30/12/2022 - Retro DOS v4.2 (MSDOS 6.21 IO.SYS SYSINIT)
23170                                   ; ----------------------------------------------------------------------------
23171                                   ; 07/04/2024 - Retro DOS v5.0 (PCDOS 7.1 IBMBIO.COM SYSINIT)
23172
23173                                   ; (MSDOS 6.22 IO.SYS - SYSINIT:0744h)
23174
23175                                   ; PCDOS 7.1 IBMBIO.COM - SYSINIT:0820h
23176
23177                                   ProcessConfig:
23178                                       ;; ds = cs ; 27/03/2019
23179                                       ; 11/12/2022
23180                                       ; ds <> cs
23181
23182                                   ; (MSDOS 5.0 IO.SYS - SYSINIT:0746h)
23183
23184 000007A6 E8BF1C                     call    doconf                  ; do pre-scan for dos=high/low
```

```
                                    ; 11/12/2022
                                    ; 27/03/2019
                                    ; ds = cs (at return from doconf)

                                ; Now, if this is not romdos, we decide what to do with the DOS code.
                                ; It will either be relocated to low memory, above the DOS data structures,
                                ; or else it will be located in HiMem, in which case a stub with the DOS
                                ; code entry points will be located in low memory. Dos_segreinit is used
                                ; to tell the DOS data where the code has been placed, and to install the
                                ; low memory stub if necessary. If the DOS is going to go into HiMem, we
                                ; must first initialize it in its present location and load the installable
                                ; device drivers. Then, if a HiMem driver has been located, we can actually
                                ; relocate the DOS code into HiMem.
                                ;
                                ; For ROMDOS, if DOS=HIGH is indicated, then we need to call dos_segreinit
                                ; to install the low memory stub (this must be done before allowing any
                                ; device drivers to hook interrupt vectors). Otherwise, we don't need to
                                ; call dos_segreinit at all, since the interrupt vector table has already
                                ; been patched.

                                    ; 22/10/2022 - Retro DOS v4.0
                                    ; (MSDOS 5.0 IO.SYS - SYSINIT:0749h)
                                    ;cmp    byte [cs:runhigh],0   ; Did user choose to run low ?
                                    ; 11/12/2022
000007A9 803E[6C02]00               cmp     byte [runhigh],0
000007AE 7404                       je      short dont_install_stub       ; yes, don't install dos low mem stub

                                ;------ user chose to load high

                                    ; 22/10/2022
                                    ;mov    es,[cs:CURRENT_DOS_LOCATION] ; MSDOS 6.21 (& MSDOS 6.0)
                                    ; 11/12/2022
                                    ; ds = cs
                                ; 13/04/2024
                                %if 0
                                    mov     es,[CURRENT_DOS_LOCATION]
                                %endif
                                    ;mov    es,[cs:FINAL_DOS_LOCATION] ; Retro DOS v4.0
                                    ; 27/03/2019
                                    ;;mov   es,[FINAL_DOS_LOCATION]

000007B0 31C0                       xor     ax,ax                 ; ax = 0 ---> install stub

                                ; 13/04/2024
                                %if 0
                                    ; 11/12/2022
                                    ; ds = cs
                                    ;call   far [cs:dos_segreinit]; call dos segreinit
                                    call    far [dos_segreinit]
                                %endif
000007B2 EB08                       jmp     short do_multi_pass

                                ;------ User chose to load dos low

                                dont_install_stub:
                                    ; 22/10/2022
000007B4 31DB                       xor     bx,bx                 ; M012
                                                                  ; don't use int 21 call to alloc mem
000007B6 E80E03                     call    MovDOSLo              ; move it !

000007B9 B80100                     mov     ax,1                  ; dont install stub

                                ; 13/04/2024
                                %if 1
                                do_multi_pass:
                                %endif
                                    ; 11/12/2022
                                    ; ds = cs
000007BC 8E06[7302]                 mov     es,[CURRENT_DOS_LOCATION]
                                    ;mov    es,[cs:CURRENT_DOS_LOCATION] ; set_dos_final_position set it up
                                    ;;mov   es,[cs:FINAL_DOS_LOCATION]   ; Retro DOS v4.0
                                    ; 27/03/2019
                                ;do_multi_pass:
                                    ;;mov   es,[FINAL_DOS_LOCATION]

                                    ; 11/12/2022
                                    ; ds =cs
                                    ;call   far [cs:dos_segreinit]; inform dos about new seg
000007C0 FF1E[7D02]                 call    far [dos_segreinit]

                                ; 13/04/2024
                                %if 0
                                do_multi_pass:
                                %endif

000007C4 E84A02                     call    AllocFreeMem          ; allocate all the free mem
                                                                  ; & update [memhi] & [area]
                                                                  ; start of free memory.
                                    ;ifdef dblspace_hooks
                                    ;mov    bx,0                  ; magic backdoor to place int hooks
                                    ;call   cs:MagicBackdoor
                                    ;endif

                                    ; 07/04/2024 - Retro DOS v5.0
                                    ; (PCDOS 7.1 IBMBIO.COM)
                                    ;cmp    byte [cs:F5_key],1
000007C7 803E[8E03]01               cmp     byte [F5_key],1
000007CC 740D                       je      short skip_magicbackdoor
                                    ;cmp    byte [cs:F8_key],1
000007CE 803E[8F03]01               cmp     byte [F8_key],1
000007D3 7406                       je      short skip_magicbackdoor
000007D5 31DB                       xor     bx,bx   ; bx = 0      ; magic backdoor to place int hooks
                                    ;call   far [cs:MagicBackdoor]
000007D7 FF1E[9003]                 call    far [MagicBackdoor]

                                skip_magicbackdoor:

                                ; Now, process config.sys some more.
                                ; Load the device drivers and install programs

                                    ; 22/10/2022
                                    ;inc    byte [cs:multi_pass_id]      ; multi_pass_id = 1
                                    ; 11/12/2022
                                    ; ds = cs
000007DB FE06[CD02]                 inc     byte [multi_pass_id]
000007DF E8221D                     call    multi_pass            ; load device drivers
000007E2 E8EA31                     call    ShrinkUMB
000007E5 E80E32                     call    UnlinkUMB             ; unlink all UMBs     ;M002
                                    ; 02/11/2022
                                    ;inc    byte [cs:multi_pass_id]      ; multi_pass_id = 2
                                    ; 11/12/2022
                                    ; ds = cs
000007E8 FE06[CD02]                 inc     byte [multi_pass_id]
```

```
                                    call   multi_pass            ; was load ifs (now does nothing)

                                    ;ifdef dblspace_hooks
                                    ;call  MagicPostload         ; make sure Magicdrv is final placed
                                    ;endif

                                    ; ds = cs

                                    ; 07/04/2024
                                    ;call  endfile               ; setup fcbs, files, buffers etc

                                    ;ifdef dblspace_hooks
                                    ;call  MagicSetCdss          ; disable CDSs of reserved drives
                                    ;endif

                                    ; 07/04/2024 - Retro DOS v5.0
                                    ; (PCDOS 7.1 IBMBIO.COM)
                                    ;cmp    byte [cs:F5_key],1
000007EF 803E[8E03]01               cmp    byte [F5_key],1
000007F4 7412                       je     short skip_magicpostload
                                    ;cmp    byte [cs:F8_key],1
000007F6 803E[8F03]01               cmp    byte [F8_key],1
000007FB 740B                       je     short skip_magicpostload
000007FD E8B710                     call   MagicPostload         ; make sure Magicdrv is final placed
                                    ; 13/04/2024
                                    ; ds = cs
00000800 E83E06                     call   endfile               ; setup fcbs, files, buffers etc
00000803 E81011                     call   MagicSetCdss          ; disable CDSs of reserved drives
                                    ; ds = cs
00000806 EB03                       jmp    short _@_

                                skip_magicpostload:
                                    ; 13/04/2024
                                    ; ds = cs
00000808 E83606                     call   endfile               ; setup fcbs, files, buffers etc
                                _@_:

                                ;Reset SysinitPresent flag here. This is needed for the special fix for lying
                                ;to device drivers. This has been moved up to this point to avoid problems
                                ;with overlays called from installed programs

                                    ; 11/12/2022
                                    ; ds = cs

                                    ;;mov  ax,Bios_Data ; 0070h
                                    ;mov   ax,KERNEL_SEGMENT
                                    ; 21/10/2022
0000080B B87000                     mov    ax,DOSBIODATASEG ; 0070h
0000080E 8EC0                       mov    es,ax                 ; point ES to bios data

00000810 26C606[DD07]00             mov    byte [es:SysinitPresent],0 ; clear SysinitPresent flag

                                    ; 02/11/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
                                    ;test  word [cs:install_flag],have_install_cmd ; 1
                                    ;test  byte [cs:install_flag],1
                                    ; 11/12/2022
                                    ; ds = cs
00000816 F606[CE02]01               test   byte [install_flag],1
                                    ;test  byte [cs:install_flag],have_install_cmd
                                                                  ; are there install commands?
0000081B 7407                       jz     short dolast          ; no, no need for further processing
                                    ;inc   byte [cs:multi_pass_id]       ; mult_pass_id = 3
                                    ; 11/12/2022
                                    ; ds =cs
0000081D FE06[CD02]                 inc    byte [multi_pass_id]
00000821 E8E01C                     call   multi_pass            ; execute install= commands

                                dolast:

                                ; [area] has the segment address for the allocated memory of sysinit, confbot.
                                ;  free the confbot area used for config.sys and sysinit itself.

                                ; Now if DOS is supposed to run high, we actually move it into high memory
                                ; (if HiMem manager is available). For ROMDOS, we don't actually move
                                ; anything, but just set up the ROM area for suballocation (or print
                                ; a message if HiMem is not available).
                                ;
                                ; There is also this little hack for CPM style DOS calls that needs to
                                ; be done when A20 is set...

                                    ; 11/12/2022
                                    ; ds = cs

                                    ; 22/10/2022
                                    ;cmp    byte [cs:runhigh],0FFh; are we still waiting to be moved?
                                    ; 11/12/2022
00000824 803E[6C02]FF               cmp    byte [runhigh],0FFh
00000829 7503                       jne    short _@@_ ; 09/12/2022 ; no, our job is over
0000082B E84802                     call   LoadDOSHiOrLo
                                _@@_:
                                    ;cmp    byte [cs:runhigh],0   ; are we running low
                                    ; 11/12/2022
                                    ; ds = cs
0000082E 803E[6C02]00               cmp    byte [runhigh],0
                                    ;je    short _@@@
00000833 7403                       je     short ConfigDone      ; yes, no CPM hack needed
00000835 E84C05                     call   CPMHack               ; make ffff:d0 same as 0:c0
                                _@@@:

                                ; We are now done with CONFIG.SYS processing

                                ConfigDone:
                                    ; 12/12/2022
                                    ; 22/10/2022
                                    ;mov    byte [cs:donotshownum],1
                                                                  ; done with config.sys.
                                                                  ; do not show line number message.
                                    ;mov    es,[cs:area]
                                    ; 12/12/2022
                                    ; ds = cs
                                    ; 27/03/2019
00000838 C606[5503]01               mov    byte [donotshownum],1
0000083D 8E06[6803]                 mov    es,[area]

00000841 B449                           mov     ah,49h ; DEALLOC  ; free allocated memory for command.com
00000843 CD21                       int    21h
                                                   ; DOS - 2+ - FREE MEMORY
                                                   ; ES = segment address of area to be freed

                                    ; 22/10/2022
                                    ;test  word [cs:install_flag],2
                                    ;test  word [cs:install_flag],has_installed ; sysinit_base installed?
                                    ;test  byte [cs:install_flag],has_installed
                                    ; 11/12/2022
```

```
                                    ; ds = cs
00000845 F606[CE02]02               test    byte [install_flag],2 ; has_installed
                                    ;test   byte [install_flag],has_installed
0000084A 741F                       jz      short skip_free_sysinitbase ; no.

                                ; set block from the old_area with impossible_owner_size.
                                ; this will free the unnecessary sysinit_base that had been put in memory to
                                ; handle install= command.

                                    ; 12/12/2022
                                        ;push      es             ; BUGBUG 3-30-92 JeffPar: no reason to save ES
                                    ;push   bx

                                    ; 22/10/2022
                                    ;mov    es,[cs:old_area]
                                    ;mov    bx,[cs:impossible_owner_size]
                                    ; 12/12/2022
                                    ; ds = cs
0000084C 8E06[5E03]                 mov     es,[old_area]
00000850 8B1E[6003]                 mov     bx,[impossible_owner_size]

00000854 B44A                       mov     ah,4Ah ; SETBLOCK
00000856 CD21                       int     21h
                                            ; DOS - 2+ - ADJUST MEMORY BLOCK SIZE (SETBLOCK)
                                            ; ES = segment address of block to change
                                            ; BX = new size in paragraphs
00000858 8CC0                       mov     ax,es
0000085A 48                         dec     ax
0000085B 8EC0                       mov     es,ax               ; point to arena
                                    ;mov    word [es:ARENA.OWNER],8      ; set impossible owner
0000085D 26C70601000800             mov     word [es:1],8
                                    ;mov    word [es:ARENA.NAME],'SD' ; 4453h ; System Data
00000864 26C70608005344             mov     word [es:8],'SD'

                                    ; 12/12/2022
                                    ;pop    bx
                                        ;pop       es             ; BUGBUG 3-30-92 JeffPar: no reason to save ES

                                skip_free_sysinitbase:
                                    ; 22/10/2022
                                    ;cmp    byte [cs:runhigh],0
                                    ; 12/12/2022
                                    ; ds = cs
0000086B 803E[6C02]00               cmp     byte [runhigh],0
00000870 7403                       je      short _@@@_ ; 04/07/2023

00000872 E8DF03                     call    InstVDiskHeader       ; Install VDISK header (allocates some mem from DOS)

                                ; ----------------------------------------------------------------------------

                                _@@@_:
                                    ; 12/12/2022
                                    ; ds = cs
                                    ; 22/10/2022
                                    ; 27/03/2019
                                    ;push   cs
                                    ;pop    ds                  ; point DS to sysinitseg

                                ; set up the parameters for command

                                ;   ; 22/10/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 IO.SYS SYSINIT)
                                ;;ifdef     MULTI_CONFIG
                                ;   mov     byte [config_cmd],0   ; set special code for query_user
                                ;       call    query_user        ; to issue the AUTOEXEC prompt
                                ;   jnc     short process_autoexec; we should process autoexec normally
                                ;   ; !!!
                                ;   or      byte [bQueryOpt],4 ; MSDOS 6.21 IO.SYS - SYSINIT:081Fh
                                ;       ; !!!
                                ;   call    disable_autoexec        ; no, we should disable it
                                ;process_autoexec:
                                ;;endif     ; !!!
                                ;   call    CheckQueryOpt  ; MSDOS 6.21 IO.SYS - SYSINIT:0827h
                                ;   ; !!!

                                    ; 22/10/2022
                                    ;mov     cl,[command_line]
                                        ;mov     ch,0
                                        ;inc     cx
                                        ;mov     si,command_line
                                    ;add     si,cx
                                        ;mov     byte [si],cr     ; cr-terminate command line

                                    ; 22/10/2022 - Retro DOS 4.0 (Modified MSDOS 5.0 IO.SYS)
                                    ; (SYSINIT:0809h)

                                    ;;;;
                                    ; 30/12/2022 - Retro DOS 4.2 (Modified MSDOS 6.21 IO.SYS)
                                    ; (SYSINIT:0813h)
                                    ; ds = cs
                                    ; push  cs
                                    ; pop   ds

00000875 C606[6419]00               mov     byte [config_cmd],0   ; set special code for query_user
0000087A E89F3D                     call    query_user            ; to issue the AUTOEXEC prompt
                                    ; 07/04/2024
                                    ;jnc    short process_autoexec; we should process autoexec normally

                                    ; 07/04/2024 - Retro DOS v5.0 (PCDOS 7.1 IBMBIO.COM)
                                    ;;;
0000087D 9C                         pushf
0000087E F606[814C]01               test    byte [bDisableUI],1
00000883 7507                       jnz     short _@@@@_   ; F5 clean/interactive boot option (has been) disabled
00000885 803E[8E03]01               cmp     byte [F5_key],1
0000088A 7405                       je      short _@@@@@_  ; F5 key pressed, bypass AUTOEXEC.BAT (clean boot)
                                _@@@@_:
0000088C 9D                         popf
0000088D 730B                       jnc     short process_autoexec; we should process autoexec normally
0000088F EB01                       jmp     short bypass_autoexec
                                _@@@@@_:
00000891 9D                         popf                    ; cf status at the return from 'query_user' call
                                bypass_autoexec:
                                    ;;;

                                    ; !!!
00000892 800E[854C]04               or      byte [bQueryOpt],4 ; MSDOS 6.21 IO.SYS - SYSINIT:081Fh
                                    ; !!!
00000897 E87D3E                     call    disable_autoexec        ; no, we should disable it
                                process_autoexec:
                                    ; !!!
0000089A E8C53E                     call    CheckQueryOpt  ; MSDOS 6.21 IO.SYS - SYSINIT:0827h

                                    ;mov     cl,[command_line]
                                    ; 30/12/2022
```

```
23557 0000089D BE[BB4B]                     mov     si,command_line
23558 000008A0 8A0C                         mov     cl,[si]
23559 000008A2 B500                         mov     ch,0
23560 000008A4 41                           inc     cx
23561                                       ;mov    si,command_line
23562 000008A5 01CE                         add     si,cx
23563 000008A7 C6040D                       mov     byte [si],cr ; 0Dh    ; cr-terminate command line
23564
23565                                       ;;;;
23566
23567                                   ; 30/12/2022 - Retro DOS v4.2
23568                                   %if 0
23569                                       ;mov    si,(offset command_line+1)
23570                                       mov     si,command_line+1
23571                                       push    ds
23572                                       pop     es
23573                                       mov     di,si
23574                                       mov     cl,0FFh ; -1
23575                                   _@_loop:
23576                                       inc     cl ; +1
23577                                       lodsb
23578                                       stosb
23579                                       or      al,al
23580                                       jnz     short _@_loop
23581                                       dec     di
23582                                       mov     al,0Dh
23583                                       stosb                   ; cr-terminate command line
23584                                       mov     [command_line],cl ; command line length (except CR)
23585                                   %endif
23586
23587                                   ; ----------------------------------------------------------------------------
23588
23589                                   ;   Once we get to this point, the above code, which is below "retry"
23590                                   ;   in memory, can be trashed (and in fact is -- see references to retry
23591                                   ;   which follow....)
23592
23593                                   retry:              ; PCDOS 7.1 IBMBIO.COM - SYSINIT:094Ch ; 07/04/2024
23594 000008AA BA[2D4B]                     mov     dx,commnd       ; now pointing to file description
23595
23596                                   ; we are going to open the command interpreter and size it as is done in
23597                                   ; ldfil. the reason we must do this is that sysinit is in free memory. if
23598                                   ; there is not enough room for the command interpreter,exec will probably
23599                                   ; overlay our stack and code so when it returns with an error sysinit won't be
23600                                   ; here to catch it. this code is not perfect (for instance .exe command
23601                                   ; interpreters are possible) because it does its sizing based on the
23602                                   ; assumption that the file being loaded is a .com file. it is close enough to
23603                                   ; correctness to be usable.
23604
23605                                   ; first, find out where the command interpreter is going to go.
23606
23607 000008AD 52                         push    dx              ; save pointer to name
23608 000008AE BBFFFF                       mov     bx,0FFFFh
23609 000008B1 B448                         mov     ah,48h  ; ALLOC
23610 000008B3 CD21                             int     21h             ; get biggest piece
23611 000008B5 B448                         mov     ah,48h  ; ALLOC
23612 000008B7 CD21                         int     21h             ; second time gets it
23613 000008B9 726B                         jc      short memerrjx ; oooops
23614
23615 000008BB 8EC0                         mov     es,ax
23616 000008BD B449                         mov     ah,49h  ; DEALLOC
23617 000008BF CD21                         int     21h             ; give it right back
23618 000008C1 89DD                         mov     bp,bx
23619
23620                                   ; es:0 points to block,and bp is the size of the block in para.
23621
23622                                   ; we will now adjust the size in bp down by the size of sysinit.
23623                                   ; we need to do this because exec might get upset if some of the exec
23624                                   ; data in sysinit is overlayed during the exec.
23625
23626                                       ; 22/10/2022
23627                                       ; (MSDOS 5.0 IO.SYS SYSINIT:083Bh)
23628 000008C3 8B1E[9402]                      mov     bx,[MEMORY_SIZE] ; get location of end of memory
23629 000008C7 8CC8                         mov     ax,cs           ; get location of beginning of sysinit
23630
23631                                   ; Note that the "config_wrkseg" environment data is a segment in
23632                                   ; unallocated memory (as of the Dealloc of [area], above). This is ideal
23633                                   ; in one sense, because Exec is going to make a copy of it for COMMAND.COM
23634                                   ; anyway, and no one has responsibility for freeing "config_wrkseg". But
23635                                   ; we need to make sure that there's no way Exec will stomp on that data
23636                                   ; before it can copy it, and one way to do that is to make the available
23637                                   ; memory calculation even more "paranoid", by subtracting "config_wrkseg"
23638                                   ; from the "memory_size" segment value (which is typically A000h) instead
23639                                   ; of the current sysinit CS....
23640                                   ;
23641                                   ; The reason I use the term "paranoid" is because this code should have
23642                                   ; slid the data required by Exec up to the very top of memory, because as
23643                                   ; it stands, you have to have sizeof(COMMAND.COM) PLUS 64K to load just
23644                                   ; COMMAND.COM (64k is about what sysinit, and all the goop above sysinit,
23645                                   ; consumes). Now it's just a little worse (65K or more, depending on
23646                                   ; the size of your CONFIG.SYS, since the size of the environment workspace
23647                                   ; is determined by the size of CONFIG.SYS.... -JTP
23648
23649                                       ; 30/12/2022 - Retro DOS v4.2 (Modified MSDOS 6.21, IO.SYS)
23650                                       ; (SYSINIT:0858h)
23651 000008C9 8B0E[6019]                   mov     cx,[config_envlen]
23652 000008CD E303                             jcxz        no_env          ; use config_wrkseg only if there's env data
23653 000008CF A1[6219]                         mov ax,[config_wrkseg]
23654
23655                                       ; 22/10/2022
23656                                       ;mov    cx,[config_envlen]
23657                                           ;jcxz       no_env          ; use config_wrkseg only if there's env data
23658                                           ;mov        ax,[config_wrkseg]
23659                                   ;no_env:
23660                                       ; 22/10/2022
23661                                       ; (MSDOS 5.0 IO.SYS SYSINIT:0841h)
23662                                   no_env:
23663                                       ; 30/12/2022
23664                                       ; (MSDOS 6.21 IO.SYS SYSINIT:0861h)
23665 000008D2 29C3                         sub     bx,ax           ; bx is size of sysinit in para
23666 000008D4 83C311                       add     bx,11h          ; add the sysinit php
23667 000008D7 29DD                         sub     bp,bx           ; sub sysinit size from amount of free memory
23668 000008D9 724B                         jc      short memerrjx ; if there isn't even this much memory, give up
23669
23670                                           ;mov        ax,(OPEN<<8)   ; open the file being execed
23671 000008DB B8003D                           mov ax,3D00h
23672 000008DE F9                           stc                     ; in case of int 24
23673 000008DF CD21                         int     21h
23674 000008E1 7271                         jc      short comerr   ; ooops
23675                                                   ; DOS - 2+ - OPEN DISK FILE WITH HANDLE
23676                                                   ; DS:DX -> ASCIZ filename
23677                                                   ; AL = access mode
23678                                                   ; 0 - read
23679                                       ; 22/10/2022
23680                                       ; (MSDOS 5.0 IO.SYS SYSINIT:0852h)
```

```
23681 000008E3 89C3                         mov     bx,ax           ; handle in bx
23682
23683                                ;   If the standard command interpreter is being used, verify it is correct
23684
23685                                    ; 30/12/2022 - Retro DOS v4.2
23686                                    ; (MSDOS 6.21 IO.SYS, SYSINIT:0874h)
23687 000008E5 803E[2A4B]00              cmp     byte [newcmd],0       ; was a new shell selected?
23688 000008EA 7518                      jne     short skip_validation ; yes
23689                                    ; 07/04/2024 - Retro DOS v5.0
23690                                    ; (PCDOS 7.1 IBMBIO.COM, SYSINIT:098Eh)
23691 000008EC BA[A608]                  mov     dx,retry-4      ; PCDOS 7.1 IBMBIO.COM - SYSINIT:0948h
23692 000008EF B90400                    mov     cx,4            ;
23693 000008F2 B43F                      mov     ah,READ         ;
23694 000008F4 CD21                      int     21h             ;
23695 000008F6 803E[A608]E9              cmp     byte [retry-4],0E9h
23696 000008FB 7557                      jne     short comerr
23697                                    ; 20/04/2019 - Retro DOS v4.0
23698                                    ; 30/12/2022
23699                                    ;cmp    byte [retry-1],64h ; MSDOS 6.21 IO.SYS - SYSINIT:088Ch
23700                                                               ; .. COMMAND.COM Version 6.20 (14h&0Fh)
23701                                    ; 07/04/2024 - Retro DOS v5.0
23702                                    ;;cmp   byte [retry-1],66h ; .. COMMAND.COM Version 6.22 (16h&0Fh)
23703                                    ;cmp    byte [retry-1],7Ah ; PCDOS 7.1 IBMBIO.COM - SYSINIT:099Fh
23704                                                               ; .. COMMAND.COM Version 7.10 (0Ah&0Fh)
23705 000008FD 803E[A908]7A             cmp     byte [retry-1],((MAJOR_VERSION&0Fh)<<4)|(MINOR_VERSION&0Fh)
23706 00000902 7550                      jne     short comerr    ;
23707
23708                                    ; 22/10/2022
23709                                    ;cmp    byte [newcmd],0       ; was a new shell selected?
23710                                    ;jne    short skip_validation ; yes
23711                                    ;mov    dx,retry-4
23712                                    ;mov    cx,4            ;
23713                                    ;mov    ah,READ         ;
23714                                    ;int    21h             ;
23715                                    ;cmp    byte [retry-4],0E9h
23716                                    ;jne    short comerr
23717                                    ;; 20/04/2019 - Retro DOS v4.0
23718                                    ;cmp    byte [retry-1],64h ; MSDOS 6.21 IO.SYS - SYSINIT:088Ch
23719                                    ;;cmp   byte [retry-1],((MAJOR_VERSION&0Fh)<<4)|(MINOR_VERSION&0Fh)
23720                                    ;jne    short comerr    ;
23721
23722                                ;skip_validation:
23723                                    ; 22/10/2022
23724                                    ; (MSDOS 5.0 IO.SYS SYSINIT:0854h)
23725                                skip_validation:
23726                                    ; 30/12/2022
23727                                    ; (MSDOS 6.21 IO.SYS SYSINIT:0893h)
23728 00000904 31C9                      xor     cx,cx
23729 00000906 31D2                      xor     dx,dx
23730                                    ;mov    ax,(LSEEK<<8)|2
23731 00000908 B80242                    mov     ax,4202h
23732 0000090B F9                        stc                     ; in case of int 24
23733 0000090C CD21                      int     21h             ; get file size in dx:ax
23734 0000090E 7244                      jc      short comerr
23735                                                            ; convert size in dx:ax to para in ax
23736 00000910 83C00F                    add     ax,15           ; round up size for conversion to para
23737 00000913 83D200                    adc     dx,0
23738 00000916 E88104                    call    off_to_para
23739 00000919 B10C                      mov     cl,12
23740 0000091B D3E2                      shl     dx,cl           ; low nibble of dx to high nibble
23741 0000091D 09D0                      or      ax,dx           ; ax is now # of para for file
23742 0000091F 83C010                    add     ax,10h          ; 100h byte php
23743 00000922 39E8                      cmp     ax,bp           ; will command fit in available mem?
23744 00000924 7208                      jb      short okld      ; jump if yes.
23745
23746                                ; 30/12/2022
23747                                %if 0
23748                                    ; 22/10/2022
23749                                memerrjx:   ; (MSDOS 5.0 IO.SYS SYSINIT:0876h)
23750                                    ;jmp    memerr  ; (MSDOS 5.0 IO.SYS SYSINIT:34D5h)
23751                                    ; 02/11/2022
23752                                    ;jmp    mem_err
23753                                    ; 11/12/2022
23754                                    ; ds = cs
23755                                    jmp     mem_err2
23756                                %endif
23757                                    ; 30/12/2022
23758                                    ; (MSDOS 6.21, IO.SYS, SYSINIT:08B5h)
23759                                memerrjx:
23760 00000926 BA[4951]                  mov     dx,badmem       ; "Configuration too large for memory"
23761 00000929 E82841                    call    print
23762 0000092C EB3A                      jmp     short continue
23763
23764                                okld:
23765 0000092E B43E                      mov     ah,3Eh ; CLOSE
23766 00000930 CD21                      int     21h             ; close file
23767
23768                                    ; 22/10/2022
23769 00000932 5A                        pop     dx      ; (MSDOS 5.0 IO.SYS SYSINIT:087Dh)
23770
23771                                    ; 24/03/2019
23772
23773 00000933 0E                        push    cs              ; point es to sysinitseg
23774 00000934 07                        pop     es
23775 00000935 BB[BF02]                      mov     bx,COMEXE ; point to exec block
23776                                    ; 22/10/2022
23777                                    ;pop    dx               ; recover pointer to name
23778
23779                                ;;ifdef     MULTI_CONFIG
23780
23781                                ;   If there's any environment data in "config_wrkseg", pass it to shell;
23782                                ;   there will be data if there were any valid SET commands and/or if a menu
23783                                ;   selection was made (in which case the CONFIG environment variable will be
23784                                ;   set to that selection).
23785
23786                                    ; 23/10/2022
23787                                    ;mov    cx,[config_envlen]
23788                                    ;jcxz   no_envdata
23789                                        ;mov        cx,[config_wrkseg]
23790                                ;no_envdata:
23791                                    ;;mov   [bx+EXEC0.ENVIRON],cx
23792                                    ;mov    [bx],cx
23793
23794                                ;;endif     ;MULTI_CONFIG
23795
23796                                    ; 30/12/2022 - Retro DOS v4.2
23797                                    ; (MSDOS 6.21 IO.SYS SYSINIT:08C7h)
23798 00000938 8B0E[6019]                mov     cx,[config_envlen]
23799 0000093C E304                      jcxz    no_envdata
23800 0000093E 8B0E[6219]                    mov cx,[config_wrkseg]
23801                                no_envdata:
23802                                    ;mov    [bx+EXEC0.ENVIRON],cx
23803 00000942 890F                      mov     [bx],cx
23804
```

```
                                    ; 23/10/2022
                                    ; (MSDOS 5.0 IO.SYS SYSINIT:0883h)

                                    ;mov    [bx+EXEC0.COM_LINE+2],cs ; set segments
00000944 8C4F04                     mov     [bx+4],cs
                                    ;mov    [bx+EXEC0.5C_FCB+2],cs
00000947 8C4F08                     mov     [bx+8],cs
                                    ;mov    [bx+EXEC0.6C_FCB+2],cs
0000094A 8C4F0C                     mov     [bx+12],cs

                                    ;mov    ax,(EXEC<<8) + 0
                                    ; 23/10/2022
                                    ;xor    ax,ax
                                    ;mov    ah,4Bh
                                    ; 04/07/2023
                                    ;mov    ax,4B00h
0000094D B8004B                     mov     ax,(EXEC<<8)

00000950 F9                         stc                     ; in case of int 24
00000951 CD21                           int     21h             ; go start up command
                                                ; DOS - 2+ - LOAD OR EXECUTE (EXEC)
                                                ; DS:DX -> ASCIZ filename
                                                ; ES:BX -> parameter block
                                                ; AL = subfunc: load & execute program
                                    ;push   cs
                                    ;pop    ds

                                    ; 13/04/2024
                                    ; 23/10/2022
00000953 52                         push    dx              ; push to balance fall-through pop

                                ; note fall through if exec returns (an error)
                                comerr:
                                    ; 23/10/2022
                                ;;ifdef     MULTI_CONFIG
                                    ;cmp    byte [commnd4],0
                                    ;je     short comerr2  ; all defaults exhausted, print err msg
                                    ;cmp    byte [newcmd],0
                                    ;je     short continue ; don't print err msg for defaults just yet
                                ;comerr2:
                                ;;endif

                                    ; 30/12/2022 - Retro DOS v4.2
                                    ;push   cs
                                    ;pop    ds
                                    ; 07/04/2024
                                    ; ds = cs

00000954 803E[9E4B]00               cmp     byte [commnd4],0
00000959 7407                       je      short comerr2  ; all defaults exhausted, print err msg
0000095B 803E[2A4B]00               cmp     byte [newcmd],0
00000960 7406                       je      short continue ; don't print err msg for defaults just yet
                                comerr2:
                                    ; 07/04/2024
                                    ;push   dx ; 30/12/2022

                                    ; 23/10/2022
00000962 BA[C550]                       mov     dx,badcom ; want to print command error
00000965 E8C040                     call    badfil

                                    ; 07/04/2024
                                    ;pop    dx  ; 30/12/2022
                                continue:
                                    ; 13/04/2024
                                    ; 23/10/2022
00000968 5A                         pop     dx

                                ; 30/12/2022
                                %if 0

                                ;;ifndef MULTI_CONFIG
                                    ;jmp    stall
                                    ; 24/10/2022
                                stall:              ; (MSDOS 5.0 IO.SYS, SYSINIT:0899h)
                                    jmp     short stall
                                ;;else

                                %endif

                                ; 30/12/2022 (MSDOS 6.21 SYSINIT, Retro DOS v4.2)
                                ;%if 1
                                ; 23/10/2022 (MSDOS 5.0 SYSINIT, Retrodos v4.0)
                                ;%if 0
00000969 B419                       mov     ah,GET_DEFAULT_DRIVE ; 19h
0000096B CD21                       int     21h             ;
0000096D 0441                       add     al,'A'          ;
0000096F 88C2                       mov     dl,al           ; DL == default drive letter
00000971 BE[6D4B]                   mov     si,commnd2
00000974 803E[2A4B]00               cmp     byte [newcmd],0 ; if a SHELL= was given
00000979 7505                       jne     short do_def2  ; then try the 2nd alternate;
0000097B C60400                     mov     byte [si],0    ; otherwise, the default SHELL= was tried,
0000097E EB05                       jmp     short do_def3   ; which is the same as our 2nd alt, so skip it
                                do_def2:
00000980 803C00                     cmp     byte [si],0    ; has 2nd alternate been tried?
00000983 7554                           jne short do_alt    ; no
                                do_def3:
00000985 BE[7E4B]                   mov     si,commnd3
00000988 803C00                     cmp     byte [si],0    ; has 3rd alternate been tried?
0000098B 754C                       jne     short do_alt   ; no
0000098D BE[9E4B]                   mov     si,commnd4
00000990 803C00                     cmp     byte [si],0    ; has 4th alternate been tried?
00000993 7544                       jne     short do_alt   ; no
00000995 52                         push    dx              ;
00000996 BA[3853]                   mov     dx,badcomprmpt
00000999 E8B840                     call    print          ;
0000099C 5A                         pop     dx              ; recover default drive letter in DL
                                request_input:                  ;
0000099D B402                       mov     ah,STD_CON_OUTPUT
0000099F CD21                       int     21h             ;
000009A1 52                         push    dx              ;
000009A2 B23E                       mov     dl,'>'          ;
000009A4 CD21                       int     21h             ;
000009A6 8A1E[2C4B]                 mov     bl,[tmplate+1] ; [tmplate+1] = 12
000009AA B700                       mov     bh,0            ;
000009AC C687[2D4B]0D               mov     byte [commnd+bx],0Dh
000009B1 BA[2B4B]                   mov     dx,tmplate
000009B4 B40A                       mov     ah,STD_CON_STRING_INPUT
000009B6 CD21                       int     21h             ; read a line of input
000009B8 BA[7050]                   mov     dx,crlfm       ;
000009BB E89640                     call    print           ;
000009BE 5A                         pop     dx              ;
000009BF 8A1E[2C4B]                 mov     bl,[tmplate+1] ;
000009C3 08DB                       or      bl,bl           ; was anything typed?
000009C5 74D6                       jz      short request_input ;
```

```
23929 000009C7 C606[2A4B]01              mov     byte [newcmd],1 ; disable validation for user-specified binaries
23930 000009CC C687[2D4B]00              mov     byte [commnd+bx],0 ; NULL-terminate it before execing it
23931 000009D1 C706[BB4B]000D            mov     word [command_line],0D00h
23932 000009D7 EB35                      jmp     short do_exec   ;
23933                                 do_alt:
23934 000009D9 1E                        push    ds
23935 000009DA 07                        pop     es
23936 000009DB C606[2A4B]00              mov     byte [newcmd],0 ; force validation for alternate binaries
23937 000009E0 BF[2D4B]                  mov     di,commnd       ;
23938                                 do_alt1:
23939 000009E3 AC                        lodsb                   ; copy the alternate, zapping it as we go,
23940 000009E4 C644FF00                  mov     byte [si-1],0   ; so that we know it's been tried
23941 000009E8 AA                        stosb                   ;
23942 000009E9 08C0                      or      al,al           ;
23943 000009EB 75F6                      jnz     short do_alt1   ;
23944 000009ED BF[BB4B]                  mov     di,command_line
23945 000009F0 807C023A                  cmp     byte [si+2],':'
23946 000009F4 7503                      jne     short do_alt2   ;
23947 000009F6 885401                    mov     [si+1],dl       ; stuff default drive into alt. command line
23948                                 do_alt2:                    ;
23949 000009F9 AC                        lodsb                   ;
23950 000009FA AA                        stosb                   ;
23951 000009FB 08C0                      or      al,al           ;
23952 000009FD 75FA                      jnz     short do_alt2   ;
23953 000009FF C645FF0D                  mov     byte [di-1],cr
23954
23955                                 ;;   Last but not least, see if we need to call disable_autoexec
23956
23957                                     ; MSDOS 6.0 (SYSINIT1.ASM)
23958                                     ;cmp    [command_line-1],0
23959                                         ;jne        short do_exec   ;
23960                                         ;mov        [command_line-1],'/'
23961                                     ;call   disable_autoexec ;
23962
23963                                     ; MSDOS 6.21 IO.SYS (SYSINIT:0994h)
23964 00000A03 C606[7B4C]00              mov     byte [dae_flag],0 ; 24/03/2019 - Retro DOS v4.0
23965 00000A08 E80C3D                    call    disable_autoexec
23966 00000A0B E8543D                    call    CheckQueryOpt  ; 24/03/2019 - Retro DOS v4.0
23967                                 do_exec:
23968 00000A0E E999FE                    jmp     retry          ;
23969
23970                                 ;;endif    ;MULTI_CONFIG
23971
23972                                 ;%endif ; 23/10/2022 (MSDOS 5.0 SYSINIT)
23973                                 ;%endif ; 30/12/2022 (MSDOS 6.21 SYSINIT)
23974
23975                                 ; 24/03/2019 - Retro DOS v4.0
23976
23977                                 ; ----------------------------------------------------------------------
23978                                 ; procedure : AllocFreeMem
23979                                 ;
23980                                 ; Allocate Max memory from DOS to find out where to load DOS.
23981                                 ; DOS is at temporary location when this call is being made
23982                                 ;
23983                                 ; Inputs : None
23984                                 ; Outputs: The biggest chunk of memory is allocated (all mem at init time)
23985                                 ;      [area] & [memhi] set to the para value of the start of the
23986                                 ;      free memory.
23987                                 ;
23988                                 ; Uses   : AX, BX
23989                                 ;
23990                                 ; ----------------------------------------------------------------------
23991
23992                                     ; 30/12/2022 - Retro DOS v4.2
23993                                     ; (MSDOS 6.21 IO.SYS, SYSINIT:09A2h)
23994
23995                                     ; 08/04/2024 - Retro DOS v5.0
23996                                     ; (PCDOS 7.1 IBMBIO.COM, SYSINIT:0AB5h)
23997
23998                                     ; 23/10/2022
23999                                 AllocFreeMem:
24000 00000A11 BBFFFF                    mov     bx,0FFFFh
24001 00000A14 B448                      mov     ah,48h ; ALLOC
24002 00000A16 CD21                      int     21h                     ; first time fails
24003 00000A18 B448                      mov     ah,48h ; ALLOC
24004 00000A1A CD21                      int     21h                     ; second time gets it
24005                                    ; 11/12/2022
24006                                    ; ds = cs
24007                                    ;mov    [cs:area],ax
24008                                    ;mov    [cs:memhi],ax           ; memhi:memlo now points to
24009 00000A1C A3[6803]                  mov     [area],ax
24010 00000A1F A3[6403]                  mov     [memhi],ax              ; memhi:memlo now points to
24011 00000A22 C3                        retn                            ; start of free memory
24012
24013                                    ; include msbio.cl6
24014                                 ; ----------------------------------------------------------------------
24015                                 DOSLOMSG:
24016 00000A23 484D41206E6F742061-       db      'HMA not available: Loading DOS low',0Dh,0Ah,'$'
24016 00000A2C 7661696C61626C653A-
24016 00000A35 204C6F6164696E6720-
24016 00000A3E 444F53206C6F770D0A-
24016 00000A47 24
24017                                 FEmsg:
24018 00000A48 466174616C20457272-       db      'Fatal Error: Cannot allocate Memory for DOS',0Dh,0Ah,'$'
24018 00000A51 6F723A2043616E6E6F-
24018 00000A5A 7420616C6C6F636174-
24018 00000A63 65204D656D6F727920-
24018 00000A6C 666F7220444F530D0A-
24018 00000A75 24
24019
24020                                 ; ----------------------------------------------------------------------
24021                                 ;
24022                                 ; procedure : LoadDOSHiOrLo
24023                                 ;
24024                                 ;          Tries to move DOS into HMA. If it fails then loads
24025                                 ;          DOS into Low memory. For ROMDOS, nothing is actually
24026                                 ;          moved; this just tries to allocate the HMA, and prints
24027                                 ;          a message if this is not possible.
24028                                 ;
24029                                 ; ----------------------------------------------------------------------
24030
24031                                     ; 23/10/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 IO.SYS)
24032                                 LoadDOSHiOrLo:
24033                                     ; 27/03/2019 - Retro DOS v4.0
24034                                     ; ds = cs
24035 00000A76 E81F00                    call    TryToMovDOSHi           ; Try moving it into HMA (M024)
24036                                     ;jc     short LdngLo            ; If that don't work...
24037                                     ;retn
24038                                     ; 18/12/2022
24039 00000A79 731C                      jnc     short LoadDosHi_ok
24040                                 LdngLo:
24041                                     ; 23/10/2022
24042                                     ;push   cs
24043                                     ;pop    ds
```

```
                                    ; 11/12/2022
                                    ; ds = cs
00000A7B B409                       mov     ah,9
00000A7D BA[230A]                   mov     dx,DOSLOMSG             ; inform user that we are
00000A80 CD21                       int     21h                     ; loading low

                                ;ifndef ROMDOS
                                    ; actually move the dos, and reinitialize it.

00000A82 BB0100                     mov     bx,1                            ; M012
                                                                            ;  use int 21 alloc for mem
00000A85 E83F00                     call    MovDOSLo
                                    ; 11/12/2022
                                    ; ds = cs
                                    ;mov    es,[cs:CURRENT_DOS_LOCATION]  ; give dos its temporary loc.
                                    ; 23/10/2022
00000A88 8E06[7302]                 mov     es,[CURRENT_DOS_LOCATION]
                                    ;;mov   es,[cs:FINAL_DOS_LOCATION]  ; 24/03/2019 - Retro DOS v4.0
                                    ;mov    es,[FINAL_DOS_LOCATION] ; 27/03/2019
00000A8C 31C0                       xor     ax,ax                           ; ax = 00 ---> install stub
                                    ; 11/12/2022
                                    ; ds = cs
                                    ;call   far [cs:dos_segreinit]          ; call dos segreinit
00000A8E FF1E[7D02]                 call    far [dos_segreinit] ; 27/03/2019

                                ;endif ; ROMDOS
                                    ; 23/10/2022
                                    ;mov    byte [cs:runhigh],0             ; mark that we are running lo
                                    ; 11/12/2022
                                    ; ds = cs
00000A92 C606[6C02]00               mov     byte [runhigh],0 ; 27/03/2019
                                LoadDosHi_ok:       ; 18/12/2022
00000A97 C3                         retn

                                ; ---------------------------------------------------------------------------
                                ;
                                ; procedure : TryToMovDOSHi
                                ;
                                ;           This tries to move DOS into HMA.
                                ;           Returns CY if it failed.
                                ;           If it succeeds returns with carry cleared.
                                ;
                                ;           For ROMDOS, dos_segreinit must be called again to allow
                                ;           the A20 switching code in the low mem stub to be installed.
                                ;
                                ; ---------------------------------------------------------------------------

                                    ; 23/10/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 IO.SYS)
                                    ; (MSDOS 5.0 IO.SYS - SYSINIT:092Ah)
                                TryToMovDOSHi:
                                    ; 11/12/2022
                                    ; 27/03/2019 - Retro DOS v4.0
                                    ; ds = cs
00000A98 E81300                     call    MovDOSHi
00000A9B 7210                       jc      short ttldhx

                                ;ifndef ROMDOS
                                    ; 23/10/2022
                                    ;mov    es,[cs:CURRENT_DOS_LOCATION] ; give dos its temporary loc.
                                    ;;mov   es,[cs:FINAL_DOS_LOCATION] ; 24/03/2019 - Retro DOS v4.0
                                    ; 11/12/2022
                                    ; ds = cs
00000A9D 8E06[7302]                 mov     es,[CURRENT_DOS_LOCATION]
                                ;else
                                ;   ..
                                ;endif ; ROMDOS

                                    ; 11/12/2022
                                    ; ds = cs
00000AA1 31C0                       xor     ax,ax                   ; ax = 00 ---> install stub
                                    ;call   far [cs:dos_segreinit]; call dos segreinit
00000AA3 FF1E[7D02]                 call    far [dos_segreinit]
                                    ;mov    byte [cs:runhigh],1
00000AA7 C606[6C02]01               mov     byte [runhigh],1
00000AAC F8                         clc
                                ttldhx:
00000AAD C3                         retn

                                ; ---------------------------------------------------------------------------
                                ;
                                ; procedure : MovDOSHi
                                ;
                                ;           Tries to allocate HMA and Move DOS/BIOS code into HMA
                                ;           For ROMDOS, the code is not actually moved, but the
                                ;           HMA is allocated and prepared for sub-allocation.
                                ;
                                ;           Returns : CY if it failed
                                ;
                                ; ---------------------------------------------------------------------------

                                    ; 23/10/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 IO.SYS)
                                MovDOSHi:
                                    ; 14/05/2019
                                    ; 27/03/2019 - Retro DOS v4.0
                                    ; ds = cs
00000AAE E8D600                     call    AllocHMA
00000AB1 7213                       jc      short mdhx                  ; did we get HMA?
00000AB3 B8FFFF                     mov     ax,0FFFFh                   ; yes, HMA seg = 0ffffh
00000AB6 8EC0                       mov     es,ax

                                ;ifndef ROMDOS
                                    ; actually move the BIOS and DOS

                                    ; NOTE: Retro DOS v4.0 does not move BIOS (IO.SYS) to HMA
                                    ; 24/03/2019

                                    ; 23/10/2022
00000AB8 E83200                     call    MovBIOS                     ; First move BIOS into HMA

                                    ; ES:DI points to free HMA after BIOS

                                    ; 14/05/2019
                                    ; 24/03/2019 - Retro DOS v4.0
                                    ;xor    di,di

                                    ; 23/10/2022
                                    ;mov    cx,[cs:hi_doscod_size]      ; pass the code size of DOS
                                    ; 11/12/2022
                                    ; ds = cs
00000ABB 8B0E[8302]                 mov     cx,[hi_doscod_size]         ; when it is in HMA
00000ABF E81100                     call    MovDOS                      ; and move it

                                    ; ES:DI points to free HMA after DOS
                                ;else
```

```
24168                                   ;   ; allocate space at beginning of HMA to allow for CPMHack
24169                                   ;
24170                                   ;   mov     di,0E0h                     ; room for 5 bytes at ffff:d0
24171                                   ;
24172                                   ;endif ; ROMDOS
24173
24174 00000AC2 E87602                       call    SaveFreeHMAPtr              ; Save the Free HMA ptr
24175 00000AC5 F8                           clc
24176                                   mdhx:
24177 00000AC6 C3                           retn
24178
24179                                   ; ----------------------------------------------------------------------
24180                                   ;
24181                                   ; procedure : MovDOSLo
24182                                   ;
24183                                   ;           Allocates memory from DOS and moves BIOS/DOS code into it
24184                                   ;
24185                                   ; ----------------------------------------------------------------------
24186
24187                                       ; 23/10/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 IO.SYS)
24188
24189                                   ;ifndef ROMDOS
24190
24191                                   MovDOSLo:
24192                                       ; 14/05/2019
24193                                       ; 27/03/2019 - Retro DOS v4.0
24194                                       ; ds = cs
24195 00000AC7 E84500                       call    AllocMemForDOS              ; incestuosly!!!
24196
24197                                       ; 23/10/2022
24198                                       ; 14/05/2019
24199                                       ;inc    ax  ; skip MCB
24200
24201 00000ACA 8EC0                         mov     es,ax                       ; pass the segment to MovBIOS
24202                                       ; NOTE: Retro DOS v4.0 does not move BIOS (IO.SYS) to HMA
24203                                       ; 24/03/2019
24204
24205                                       ; 23/10/2022
24206 00000ACC E81E00                       call    MovBIOS
24207
24208                                   ;------ ES:DI points memory immediately after BIOS
24209
24210                                       ; 14/05/2019
24211                                       ; NOTE:
24212                                       ;     Order of (RETRO) DOS kernel sections at memory:
24213                                       ;       BIOSDATA+BIOSCODE+BIOSDATAINIT+DOSDATA+DOSCODE(LOW)
24214
24215                                       ; 24/03/2019 - Retro DOS v4.0
24216                                       ;xor    di,di
24217
24218                                       ; 23/10/2022
24219                                       ;mov    cx,[cs:lo_doscod_size]      ; DOS code size when loaded
24220                                       ; 11/12/2022
24221                                       ; ds = cs
24222 00000ACF 8B0E[8102]                  mov     cx,[lo_doscod_size]         ; low
24223                                       ;call   MovDOS
24224                                       ;retn
24225                                       ; 11/12/2022
24226                                       ;jmp    short MovDOS
24227
24228                                   ;endif ; ROMDOS
24229
24230                                   ; 11/12/2022
24231
24232                                   ; ----------------------------------------------------------------------
24233                                   ;
24234                                   ; procedure : MovDOS
24235                                   ;
24236                                   ;           Moves DOS code into requested area
24237                                   ;
24238                                   ;   In : ES:DI - pointer to memory where DOS is to be moved
24239                                   ;        CX    - size of DOS code to be moved
24240                                   ;
24241                                   ;   Out : ES:DI - pointer to memory immediately after DOS
24242                                   ;
24243                                   ; ----------------------------------------------------------------------
24244
24245                                       ; 11/12/2022
24246                                       ; 23/10/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 IO.SYS)
24247
24248                                   ;ifndef ROMDOS
24249
24250                                   MovDOS:
24251                                       ; 14/05/2019
24252                                       ; 27/03/2019 - Retro DOS v4.0
24253
24254                                       ; 11/12/2022
24255                                       ; ds = cs
24256
24257                                       ; 23/10/2022
24258                                       ;push   ds ; *//
24259
24260 00000AD3 06                           push    es
24261 00000AD4 57                           push    di
24262
24263                                       ; 11/12/2022
24264 00000AD5 1E                           push    ds ; *// ; 11/12/202
24265
24266                                       ; 29/04/2019
24267 00000AD6 C536[7102]                  lds     si,[dosinit] ; 11/12/2022
24268                                       ; 23/10/2022
24269                                       ;lds    si,[cs:dosinit]
24270                                       ; 03/09/2023
24271 00000ADA 89F0                         mov     ax,si
24272
24273 00000ADC F3A4                         rep     movsb
24274
24275 00000ADE 1F                           pop     ds ; *// ; 11/12/2022
24276
24277 00000ADF 5B                           pop     bx                          ; get back offset into which
24278                                                                           ;  DOS was moved
24279                                       ; 03/09/2023
24280                                       ;;mov   ax,[cs:dosinit]                     ; get the offset at which DOS
24281                                                                           ;  wants to run
24282                                       ; 03/09/2023
24283                                       ;mov    ax,[dosinit]
24284                                       ; ax = [dosinit]
24285
24286 00000AE0 29D8                         sub     ax,bx
24287 00000AE2 E8B502                       call    off_to_para
24288 00000AE5 5B                           pop     bx                          ; get the segment at which
24289                                                                           ;  we moved DOS into
24290 00000AE6 29C3                         sub     bx,ax                       ; Adjust segment
24291
```

```
                                    ; 11/12/2022
                                    ; 23/10/2022
                                    ;mov    [cs:CURRENT_DOS_LOCATION],bx  ; and save it
                                    ;;mov   [cs:FINAL_DOS_LOCATION],bx
                                    ; 11/12/2022
00000AE8 891E[7302]                 mov     [CURRENT_DOS_LOCATION],bx

                                    ; 27/03/2019
                                    ;pop    ds ; *//
                                    ; ds = cs
                                    ;mov    [FINAL_DOS_LOCATION],bx

00000AEC C3                         retn

                                ;endif ;ROMDOS

                                ; NOTE: Retro DOS v4.0 does not move BIOS (IO.SYS) to HMA
                                ; 24/03/2019
                                ; ---------------------------------------------------------------------------
                                ;
                                ; procedure : MovBIOS
                                ;
                                ;          Moves BIOS code into requested segment
                                ;
                                ;   In : ES - segment to which BIOS is to be moved
                                ;             ( it moves always into offset BCode_Start)
                                ;
                                ;   Out : ES:DI - pointer to memory immediately after BIOS
                                ;
                                ; ---------------------------------------------------------------------------

                                    ; 23/10/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 IO.SYS)
                                    ; 08/04/2024 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMBIO.COM)

                                ;ifndef ROMDOS

                                MovBIOS: ; proc    near
                                    ; 11/12/2022
00000AED 1E                         push    ds ; ds = cs
                                    ;
                                    ; 23/10/2022
                                    ;mov    ds,[cs:temp_bcode_seg]        ; current BIOS code seg
                                    ; 17/09/2023 ; 08/04/2024
00000AEE 8E1E[8902]                 mov     ds,[temp_bcode_seg]
                                    ;mov    si,BCODE_START ; mov si,30h
                                    ; 09/12/2022
00000AF2 BE[3000]                   mov     si,BCODESTART ; 30h
00000AF5 89F7                       mov     di,si
                                    ;mov    cx,1E00h ; PCDOS 7.1 IBMBIO.COM ; 08/04/2024
00000AF7 B9701D                     mov     cx,BCODE_END ; mov cx,1A60h
00000AFA 29F1                       sub     cx,si                         ; size of BIOS
00000AFC D1E9                       shr     cx,1                          ; Both the labels are para
                                                                          ;  aligned
00000AFE F3A5                       rep     movsw

                                    ; 11/12/2022
00000B00 1F                         pop     ds ; ds = cs
                                    ;
00000B01 06                         push    es
00000B02 57                         push    di                            ; save end of BIOS
00000B03 8CC0                       mov     ax,es
                                    ;
                                    ; 11/12/2022
                                    ;mov    [cs:BCodeSeg],ax              ; save it for later use
                                    ;;call  dword ptr cs:_seg_reinit_ptr
                                    ;call   far [cs:seg_reinit_ptr]               ; far call to seg_reinit (M022)
                                    ; ds = cs
00000B05 A3[8A03]                   mov     [BCodeSeg],ax
00000B08 FF1E[8702]                 call    far [seg_reinit_ptr]
                                    ;
00000B0C 5F                         pop     di
00000B0D 07                         pop     es                            ; get back end of BIOS
00000B0E C3                         retn

                                ;MovBIOS endp

                                ;endif ; ROMDOS

                                ; 11/12/2022
                                %if 0

                                ; 24/03/2019

                                ; ---------------------------------------------------------------------------
                                ;
                                ; procedure : MovDOS
                                ;
                                ;          Moves DOS code into requested area
                                ;
                                ;   In : ES:DI - pointer to memory where DOS is to be moved
                                ;        CX    - size of DOS code to be moved
                                ;
                                ;   Out : ES:DI - pointer to memory immediately after DOS
                                ;
                                ; ---------------------------------------------------------------------------

                                    ; 23/10/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 IO.SYS)

                                ;ifndef ROMDOS

                                MovDOS:
                                    ; 14/05/2019
                                    ; 27/03/2019 - Retro DOS v4.0

                                    ; 11/12/2022
                                    ; ds = cs

                                    ; 23/10/2022
                                    ;push   ds ; *//

                                    push    es
                                    push    di

                                    ; 11/12/2022
                                    push    ds ; *// ; 11/12/202

                                    ; 29/04/2019
                                    lds     si,[dosinit] ; 11/12/2022
                                    ; 23/10/2022
                                    ;lds    si,[cs:dosinit]
                                    ; 03/09/2023
                                    mov     ax,si

                                    rep     movsb
```

```
                                    pop     ds ; *// ; 11/12/2022

                                    pop     bx                              ; get back offset into which
                                                                            ;  DOS was moved
                                    ;mov    ax,[dosinit] ; 03/09/2023
                                    ;;mov   ax,[cs:dosinit]                         ; get the offset at which DOS
                                                                            ;  wants to run
                                    sub     ax,bx
                                    call    off_to_para
                                    pop     bx                              ; get the segment at which
                                                                            ;  we moved DOS into
                                    sub     bx,ax                           ; Adjust segment

                                    ; 11/12/2022
                                    ; 23/10/2022
                                    ;mov    [cs:CURRENT_DOS_LOCATION],bx  ; and save it
                                    ;;mov   [cs:FINAL_DOS_LOCATION],bx
                                    ; 11/12/2022
                                    mov     [CURRENT_DOS_LOCATION],bx

                                    ; 27/03/2019
                                    ;pop    ds ; *//
                                    ; ds = cs
                                    ;mov    [FINAL_DOS_LOCATION],bx

                                    retn

                                ;endif ;ROMDOS

                                %endif

                                ; ----------------------------------------------------------------------
                                ;
                                ; procedure : AllocMemForDOS
                                ;
                                ;           Allocate memory for DOS/BIOS code from DOS !!!
                                ;
                                ;   Out : AX - seg of allocated memoryblock
                                ;
                                ; ----------------------------------------------------------------------

                                    ; 23/10/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 IO.SYS)
                                    ; 08/04/2024 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMBIO.COM)

                                ;ifndef ROMDOS

                                AllocMemForDOS:
                                    ; 11/12/2022
                                    ; 14/05/2019
                                    ; 27/03/2019 - Retro DOS v4.0
                                    ; ds = cs
                                    ;mov    ax,BCode_end
                                    ;sub    ax,BCode_start        ; BIOS code size
                                    ; 23/10/2022
00000B0F B8701D                     mov     ax,BCODE_END ; 1A60h  ; 1A70h for MSDOS 6.21
                                                                  ; 30/12/2022
                                    ;mov    ax,1E00h ; PCDOS 7.1 IBMBIO.COM ; 08/04/2024
                                    ;sub    ax,BCODE_START ; 30h
                                    ; 09/12/2022
00000B12 2D[3000]                   sub     ax,BCODESTART ; sub ax,30h ; 08/04/2024
                                    ; 24/03/2019 - Retro DOS v4.0
                                    ; 02/11/2022
                                    ;add    ax,[cs:lo_doscod_size]; DOS code size
                                    ; 11/12/2022
                                    ; ds = cs
00000B15 0306[8102]                 add     ax,[lo_doscod_size]
00000B19 83C00F                     add     ax,15
00000B1C E87B02                     call    off_to_para                 ; convert to para
                                    ; 23/10/2022
                                    ; 14/05/2019
                                    ;inc    ax ; + 1 paragraph for MCB
00000B1F 09DB                       or      bx,bx                       ; M012
00000B21 89C3                       mov     bx,ax                       ;  can we use int 21 for alloc
00000B23 741A                       jz      short update_arena          ; M012
00000B25 B448                       mov     ah,48h                      ; request DOS
00000B27 CD21                       int     21h
00000B29 7250                       jc      short FatalErr              ; IF ERR WE ARE HOSED
                                    ; 23/10/2022
                                    ; 24/03/2019 - Retro DOS v4.0 (ORG 0)
00000B2B 83E803                     sub     ax,3                        ; Take care ORG 30h of
                                                                        ;  BIOS code
00000B2E 8EC0                       mov     es,ax
                                    ;mov    word [es:20h+ARENA.OWNER],08h ; mark it as system
                                    ;mov    word [es:20h+ARENA.NAME],'SC' ;  code area
                                    ; 14/05/2019
                                    ;mov    word [es:ARENA.OWNER],08h     ; mark it as system
                                    ;mov    word [es:ARENA.NAME],'SC'     ;  code area
                                    ; 08/04/2024 (PCDOS 7.1 IBMBIO.COM)
                                    ; 23/10/2022
00000B30 26C70621000800             mov     word [es:20h+1],08h         ; mark it as system
00000B37 26C70628005343             mov     word [es:20h+8],'SC' ; 4353h ;  code area

00000B3E C3                         retn

                                ; BUGBUG -- 5 Aug 92 -- chuckst -- Allocating space for DOS
                                ;     using DOS itself causes an arena to be generated.
                                ;     Unfortunately, certain programs (like PROTMAN$)
                                ;     assume that the device drivers are loaded into
                                ;     the first arena. For this reason, MagicDrv's
                                ;     main device driver header arena is manually
                                ;     truncated from the arena chain, and the space
                                ;     for DOS is allocated using the following
                                ;     simple code, which also assumes that the
                                ;     first arena is the free one where DOS's low
                                ;     stub will go.
                                ;
                                ; M012 : BEGIN

                                    ; 23/10/2022
                                update_arena:
00000B3F 1E                         push    ds ; ds = cs
00000B40 57                         push    di
00000B41 51                         push    cx
00000B42 52                         push    dx
                                    ; 23/10/2022
                                    ;lds    di,[cs:DOSINFO]                     ; get ptr to DOS var
                                    ; 11/12/2022
                                    ; ds = cs
00000B43 C53E[6D02]                 lds     di,[DOSINFO] ; 27/03/2019
00000B47 4F                         dec     di
00000B48 4F                         dec     di                          ; Arena head is immediately
                                                                        ;  before sysvar
00000B49 8E05                       mov     es,[di]                     ; es = arena head
```

```
24540                                   ;mov    cx,[es:ARENA.SIZE]          ; cx = total low mem size
24541 00000B4B 268B0E0300               mov     cx,[es:3]
24542 00000B50 39D9                     cmp     cx,bx                       ; is it sufficient ?
24543 00000B52 7227                     jb      short FatalErr              ; no, fatal error
24544
24545                                   ;mov    dl,[es:ARENA.SIGNATURE]
24546 00000B54 268A160000               mov     dl,[es:0]
24547 00000B59 8CC0                     mov     ax,es
24548 00000B5B 01D8                     add     ax,bx                       ; ax = new arena head
24549 00000B5D 8905                     mov     [di],ax                     ; store it in DOS data area
24550 00000B5F 8ED8                     mov     ds,ax
24551                                   ;mov    [ARENA.SIGNATURE],dl        ; type of arena
24552 00000B61 88160000                 mov     [0],dl
24553                                   ;mov    word [ARENA.OWNER],0        ; free
24554 00000B65 C70601000000             mov     word [1],0
24555 00000B6B 29D9                     sub     cx,bx                       ; size of the new block
24556                                   ;mov    [ARENA.SIZE],cx                     ; store it in the arena
24557 00000B6D 890E0300                 mov     [3],cx
24558 00000B71 8CC0                     mov     ax,es                       ; return seg to the caller
24559                                   ; 23/10/2022
24560                                   ;; 24/03/2019 - Retro DOS v4.0 (ORG 0)      ; Take care ORG 30h of
24561 00000B73 83E803                   sub     ax,3                        ;  BIOS code
24562 00000B76 5A                       pop     dx
24563 00000B77 59                       pop     cx
24564 00000B78 5F                       pop     di
24565 00000B79 1F                       pop     ds ; ds = cs
24566 00000B7A C3                       retn
24567                                 ;
24568                                 ; M012 : END
24569                                 ;
24570                                 FatalErr:
24571 00000B7B 0E                       push    cs
24572 00000B7C 1F                       pop     ds
24573 00000B7D BA[480A]                 mov     dx,FEmsg
24574 00000B80 B409                     mov     ah,9
24575 00000B82 CD21                     int     21h             ; DOS - PRINT STRING
24576                                                           ; DS:DX -> string terminated by "$"
24577                                   ; 30/12/2022 (MSDOS 6.21 SYSINIT)
24578 00000B84 E9C707                   jmp     stall
24579                                   ; 23/10/2022 (MSDOS 5.0 SYSINIT)
24580                                   ;cli
24581                                   ;hlt
24582
24583                                 ;endif ;ROMDOS
24584
24585                                 ; 25/03/2019 - Retro DOS v4.0
24586
24587                                 ; ----------------------------------------------------------------------
24588                                 ;
24589                                 ; procedure : AllocHMA
24590                                 ;
24591                                 ;   grab_the_hma tries to enable a20 and make sure there is memory
24592                                 ;     up there. If it gets any sort of error, it will return with
24593                                 ;     carry set so that we can resort to running low.
24594                                 ;
24595                                 ;   It also returns ES: -> 0ffffh if it returns success
24596                                 ;
24597                                 ; ----------------------------------------------------------------------
24598
24599                                 AllocHMA:
24600                                 ;   cas note:  The pre-286 check is no longer needed here since the
24601                                 ;              presence of XMS is sufficient. However, this code hasn't
24602                                 ;              been deleted because it can be recycled for skipping the
24603                                 ;              extra pass of CONFIG.SYS and assuming we're running low
24604                                 ;              in the case of a pre-286.
24605
24606                                 ;;  see if we're running on a pre-286. If not, force low.
24607                                 ;
24608                                 ;   xor     ax,ax
24609                                 ;   pushf                   ; save flags (like int)
24610                                 ;   push    ax
24611                                 ;   popf
24612                                 ;   pushf
24613                                 ;   pop     ax
24614                                 ;   popf                    ; restore original flags (like int)
24615                                 ;   and     ax,0F000h
24616                                 ;   cmp     ax,0F000h       ; 8088/8086?
24617                                 ;   jz      short grab_hma_error
24618
24619                                     ; 23/10/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 IO.SYS)
24620                                     ; (SYSINIT:0A26h)
24621
24622                                     ; 13/04/2024 - Retro DOS v5.0
24623                                     ; (PCDOS 7.1 IBMBIO.COM - SYSINIT:0C41h)
24624
24625 00000B87 1E                        push    ds
24626                                   ;;mov   ax,Bios_Data
24627                                   ;mov    ax,KERNEL_SEGMENT
24628                                   ; 21/10/2022
24629 00000B88 B87000                   mov     ax,DOSBIODATASEG ; 70h
24630 00000B8B 8ED8                     mov     ds,ax
24631
24632 00000B8D E84A00                   call    IsXMSLoaded
24633 00000B90 7545                     jnz     short grabhma_error
24634
24635 00000B92 B81043                   mov     ax,4310h
24636 00000B95 CD2F                     int     2Fh             ; get the vector into es:bx
24637                                                           ; - Multiplex - XMS - GET DRIVER ADDRESS
24638                                                           ; Return: ES:BX -> driver entry point
24639
24640 00000B97 891E[0E00]               mov     [xms],bx
24641                                   ;mov    [0Eh], bx
24642 00000B9B 8C06[1000]               mov     [xms+2],es
24643                                   ;mov    [10h],es
24644
24645 00000B9F B401                     mov     ah,1            ; request HMA
24646 00000BA1 BAFFFF                   mov     dx,0FFFFh
24647                                   ;call   dword ptr ds:0Eh
24648 00000BA4 FF1E[0E00]               call    far [xms]
24649 00000BA8 48                       dec     ax
24650 00000BA9 7409                     jz      short allocHMA_1 ; error if not able to allocate HMA
24651
24652                                 ;------ Himem may be lying because it has allocated mem for int 15
24653
24654 00000BAB B488                      mov     ah,88h
24655 00000BAD CD15                     int     15h
24656                                                           ; Get Extended Memory Size
24657                                                           ; Return: CF clear on success
24658                                                           ; AX = size of memory above 1M in K
24659 00000BAF 83F840                   cmp     ax,64           ; less than 64 K of hma ?
24660                                   ;jb     short grabhma_error
24661                                   ; 11/12/2022
24662 00000BB2 7224                     jb      short grabhma_err ; cf=1
24663                                 allocHMA_1:
```

```
00000BB4 B405                           mov     ah,5            ; localenableA20
                                        ;call   dword ptr ds:0Eh
00000BB6 FF1E[0E00]                     call    far [xms]
00000BBA 48                             dec     ax
00000BBB 751A                           jnz     short grabhma_error ; error if couldn't enable A20

00000BBD E89D01                         call    IsVDiskInstalled
00000BC0 7415                           jz      short grabhma_error ; yes, we cant use HMA

00000BC2 B8FFFF                         mov     ax,0FFFFh
00000BC5 8EC0                           mov     es,ax
00000BC7 26C70610003412                 mov     word [es:10h],1234h ; see if we can really read/write there
00000BCE 26813E10003412                 cmp     word [es:10h],1234h
                                        ;jne    short grabhma_error ; don't try to load there if XMS lied
                                        ; 11/12/2022
00000BD5 7401                           je      short allocHMA_ok

                                ; 11/12/2022
                                ;   ; 11/12/2022
                                ;   ; cf=0
                                ;   ;clc
                                ;   pop     ds
                                ;   retn

                                grabhma_error:
00000BD7 F9                             stc
                                        ; 11/12/022
                                grabhma_err:        ; cf=1
                                allocHMA_ok:        ; cf=0
00000BD8 1F                             pop     ds
00000BD9 C3                             retn

                                ; ---------------------------------------------------------------------------
                                ;
                                ; procedure : IsXMSLoaded
                                ;
                                ;             Checks whether a XMS driver is loaded
                                ;
                                ; Returns : Z flag set if XMS driver loaded
                                ;           Z flag reset if no XMS drivers are present
                                ;
                                ; ---------------------------------------------------------------------------

                                IsXMSLoaded:
00000BDA B80043                         mov     ax,4300h
00000BDD CD2F                           int     2Fh             ; - Multiplex - XMS - INSTALLATION CHECK
                                                                ; Return: AL = 80h XMS driver installed
                                                                ; AL <> 80h no driver
00000BDF 3C80                           cmp     al,80h          ; XMS installed?
00000BE1 C3                             retn

                                ; ---------------------------------------------------------------------------
                                ; procedure : FTryToMovDOSHi
                                ;
                                ;           Called from HMA suballoc calls
                                ;
                                ; ---------------------------------------------------------------------------

                                        ; 23/10/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 IO.SYS)
                                        ; (MSDOS 5.0 IO.SYS - SYSINIT:0A84h)

                                        ; 08/04/2024 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMBIO.COM)
                                        ; (PCDOS 7.1 IBMBIO.COM - SYSINIT:0C9Fh)

                                        ; ((MSDOS 6.22 IO.SYS - SYSINIT:0B8Ch))

                                FTryToMovDOSHi:    ; proc  far

00000BE2 50                             push    ax
00000BE3 53                             push    bx
00000BE4 51                             push    cx
00000BE5 52                             push    dx
00000BE6 56                             push    si
00000BE7 57                             push    di
00000BE8 1E                             push    ds
00000BE9 06                             push    es

                                        ; 23/10/2022
                                        ; 27/03/2019 - Retro DOS v4.0
                                        ; 11/12/2022
00000BEA 0E                             push    cs
00000BEB 1F                             pop     ds

                                        ;cmp    byte [cs:runhigh],0FFh
                                        ; 11/12/2022
00000BEC 803E[6C02]FF                   cmp     byte [runhigh],0FFh
00000BF1 7503                           jne     short _ftymdh_1

                                        ; ds = cs
00000BF3 E8A2FE                         call    TryToMovDOSHi
                                _ftymdh_1:
00000BF6 07                             pop     es
00000BF7 1F                             pop     ds
00000BF8 5F                             pop     di
00000BF9 5E                             pop     si
00000BFA 5A                             pop     dx
00000BFB 59                             pop     cx
00000BFC 5B                             pop     bx
00000BFD 58                             pop     ax

00000BFE CB                             retf

                                ; ---------------------------------------------------------------------------
                                ;
                                ; following piece of code will be moved into a para boundary. And the para
                                ; address posted in seg of int 19h vector. Offset of int 19h will point to
                                ; VDint19. This is to protect HMA from apps which use VDISK header method
                                ; to determine free extended memory.
                                ;
                                ; For more details read "power programming" column by Ray Duncan in the
                                ; May 30 1989 issue of PC Magazine (pp 377-388) [USING EXTENDED MEMORY,PART 1]
                                ;
                                ; ---------------------------------------------------------------------------

                                        ; 30/12/2023 - Retro DOS 5.0
00000BFF 00                             db      0

                                ; 13/04/2024
                                ;align 2

                                        ; 30/12/2023
                                        ; PCDOS v7.1 IBMBIO.COM, SYSYINIT:0CBCh

                                StartVDHead:
```

```
24788                                   ;-------------- what follows is a dummy device driver header (not used by DOS)
24789
24790 00000C00 00000000                     dd      0               ; link to next device driver
24791 00000C04 0080                         dw      8000h           ; device attribute
24792 00000C06 0000                         dw      0               ; strategy routine offset
24793 00000C08 0000                         dw      0               ; interrupt routine offset
24794 00000C0A 01                           db      1               ; number of units
24795                                       ;db     7 dup(0)
24796 00000C0B 00<rep 7h>                   times   7 db 0          ; reserved area
24797                                   VDiskSig1:
24798 00000C12 564449534B                   db      'VDISK'
24799
24800                                   VLEN1       equ     ($-VDiskSig1)
24801
24802 00000C17 202056332E33                 db      '  V3.3'        ; vdisk label
24803                                       ;db     15 dup (0)      ; pad
24804 00000C1D 00<rep Fh>                   times   15 db 0
24805 00000C2C 0000                         dw      0               ; bits 0-15 of free HMA
24806 00000C2E 11                           db      11h             ; bits 16-23 of free HMA (1M + 64K)
24807                                   VDInt19:
24808 00000C2F EA                           db      0EAh            ; jmp to old vector
24809                                   OldVDInt19:
24810 00000C30 00000000                     dd      0               ; Saved int 19 vector
24811
24812                                   EndVDHead: ; label byte
24813
24814                                   VDiskHMAHead:
24815 00000C34 000000                       db      0,0,0           ; non-bootable disk
24816                                   VDiskSig2:
24817 00000C37 564449534B                   db      'VDISK'
24818
24819                                   VLEN2       equ     ($-VDiskSig2)
24820
24821 00000C3C 332E33                       db      '3.3'           ; OEM - signature
24822 00000C3F 8000                         dw      128             ; number of bytes/sector
24823 00000C41 01                           db      1               ; sectors/cluster
24824 00000C42 0100                         dw      1               ; reserved sectors
24825 00000C44 01                           db      1               ; number of FAT copies
24826 00000C45 4000                         dw      64              ; number of root dir entries
24827 00000C47 0002                         dw      512             ; number of sectors
24828 00000C49 FE                           db      0FEh            ; media descriptor
24829 00000C4A 0600                         dw      6               ; number of sectors/FAT
24830 00000C4C 0800                         dw      8               ; sectors per track
24831 00000C4E 0100                         dw      1               ; number of heads
24832 00000C50 0000                         dw      0               ; number of hidden sectors
24833 00000C52 4004                         dw      440h            ; Start of free HMA in K (1M+64K)
24834
24835                                   EndVDiskHMAHead: ; label byte
24836
24837                                   ; --------------------------------------------------------------------
24838                                   ;
24839                                   ; procedure : InstVDiskHeader
24840                                   ;
24841                                   ;             Installs the VDISK header to reserve the 64k of HMA
24842                                   ;         It puts a 32 byte header at 10000:0 and
24843                                   ;         another header at (seg of int19):0
24844                                   ;
24845                                   ; Inputs : None
24846                                   ;
24847                                   ; Outputs : None
24848                                   ;
24849                                   ; USES : DS,SI,AX,CX,DX
24850                                   ;
24851                                   ; --------------------------------------------------------------------
24852
24853                                       ; 23/10/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 IO.SYS)
24854
24855                                   InstVDiskHeader:
24856 00000C54 31C0                         xor     ax,ax
24857 00000C56 8ED8                         mov     ds,ax                   ; seg of int vect table
24858
24859                                   ;-------------- save old int 19 vector
24860
24861                                       ; 23/10/2022
24862 00000C58 A16400                       mov     ax,[19h*4]
24863                                       ;mov    [OldVDInt19],ax
24864 00000C5B 2EA3[300C]                   mov     [cs:OldVDInt19],ax
24865 00000C5F A16600                       mov     ax,[19h*4+2]
24866                                       ;mov    [OldVDInt19+2],ax
24867 00000C62 2EA3[320C]                   mov     [cs:OldVDInt19+2],ax
24868
24869                                   ;-------------- calculate seg of new int 19 handler
24870
24871 00000C66 B448                         mov     ah,48h                  ; allocate memory
24872                                       ;mov    bx,(EndVDHead-StartVDHead+15)>>4
24873                                       ; 23/10/2022
24874 00000C68 BB0400                       mov     bx,4
24875 00000C6B CD21                         int     21h
24876
24877                                   ;   if carry, fatal hanging error!!!!!
24878
24879 00000C6D 48                           dec     ax                      ; point to arena
24880 00000C6E 8EC0                         mov     es,ax
24881                                       ;mov    word [es:ARENA.OWNER],8       ; owner = System
24882 00000C70 26C70601000800               mov     word [es:1],8
24883                                       ;mov    word [es:ARENA.NAME],'SC' ; System Code
24884 00000C77 26C70608005343               mov     word [es:8],'SC' ; 4353h
24885 00000C7E 40                           inc     ax
24886 00000C7F 8EC0                         mov     es,ax                   ; get back to allocated memory
24887
24888                                   ;-------------- install new int 19 vector
24889
24890 00000C81 FA                           cli                             ; no reboots at this time
24891                                       ;mov    word [19h*4],(VDInt19-StartVDHead)
24892 00000C82 C70664002F00                 mov     word [19h*4],47
24893 00000C88 A36600                       mov     [19h*4+2],ax
24894
24895                                   ;-------------- move the code into proper place
24896
24897                                       ;mov    cx,(EndVDHead-StartVDHead)
24898 00000C8B B93400                       mov     cx,52
24899 00000C8E BE[000C]                     mov     si,StartVDHead
24900 00000C91 31FF                         xor     di,di
24901 00000C93 0E                           push    cs
24902 00000C94 1F                           pop     ds
24903 00000C95 FC                           cld
24904 00000C96 F3A4                         rep     movsb
24905 00000C98 FB                           sti                             ; BUGBUG is sti OK now?
24906
24907                                   ;-------------- mov the HMA VDisk head into HMA
24908
24909                                       ; 23/10/2022
24910 00000C99 57                           push    di
24911 00000C9A 06                           push    es
```

```
24912
24913                                   ;mov    ax,0FFFFh
24914                                   ;mov    es,ax
24915                                   ; 03/09/2023
24916 00000C9B 49                       dec     cx
24917                                   ; cx = 0FFFFh
24918 00000C9C 8EC1                     mov     es,cx
24919
24920 00000C9E BF1000                   mov     di,10h
24921                                   ;mov    cx,(EndVDiskHMAHead-VDiskHMAHead)
24922 00000CA1 B92000                   mov     cx,32
24923 00000CA4 BE[340C]                 mov     si,VDiskHMAHead
24924 00000CA7 F3A4                     rep     movsb                   ; ds already set to cs
24925
24926 00000CA9 5F                       pop     di
24927 00000CAA 07                       pop     es
24928
24929 00000CAB C3                       retn
24930
24931                                 ; ----------------------------------------------------------------------
24932                                 ; procedure : ClrVDISKHeader
24933                                 ;
24934                                 ;         Clears the first 32 bytes at 1MB boundary
24935                                 ;         So that DOS/HIMEM is not confused about the VDISK header
24936                                 ;         left by previous DOS=HIGH session
24937                                 ;
24938                                 ; ----------------------------------------------------------------------
24939
24940                                 struc desc
24941 00000000 ????                   .seg_lim:  resw   1             ; seg limit 64K
24942 00000002 ????                   .lo_word:  resw   1             ; 24 bit seg physical
24943 00000004 ??                     .hi_byte:  resb   1             ; address
24944 00000005 ??                     .acc_rights:     resb   1             ; access rights ( CPL0 - R/W )
24945 00000006 ????                   .reserved: resw   1             ;
24946                                 .size:
24947                                 endstruc
24948
24949                                           ; 23/10/2022
24950                                 bmove:            ;label byte
24951
24952                                 dummy:            ;times desc.size db 0 ; desc <>
24953 00000CAC 00<rep 8h>                       times 8 db 0
24954                                 gdt:      ;times desc.size db 0 ; desc <>
24955 00000CB4 00<rep 8h>                       times 8 db 0
24956 00000CBC FFFF                 src_desc: dw      0FFFFh        ; desc <0ffffh,0,0,93h,0>
24957 00000CBE 0000                             dw      0
24958 00000CC0 00                               db      0
24959 00000CC1 93                               db      93h
24960 00000CC2 0000                             dw      0
24961 00000CC4 FFFF                 tgt_desc: dw      0FFFFh        ; desc <0ffffh,0,10h,93h,0>  ; 1MB
24962 00000CC6 0000                             dw      0
24963 00000CC8 10                               db      10h
24964 00000CC9 93                               db      93h
24965 00000CCA 0000                             dw      0
24966
24967                                 rombios_code:     ;times desc.size db 0 ; desc <>
24968 00000CCC 00<rep 8h>                       times 8 db 0
24969                                 temp_stack: ;times desc.size db 0 ; desc <>
24970 00000CD4 00<rep 8h>                       times 8 db 0
24971
24972 00000CDC 00<rep 20h>          ClrdVDISKHead:    times 32 db 0         ; db 32 dup (0)
24973
24974
24975                                 ; 25/03/2019 - Retro DOS v4.0 (MSDOS 6.21 IO.SYS, MSDOS 6.0 SYSINIT1.ASM)
24976
24977                                 ; 30/12/2022 - Retro DOS v4.2 (Modified MSDOS 6.21 IO.SYS)
24978                                 ; (SYSINIT:0CA6h)
24979
24980                                 ClrVDISKHeader:   ; proc  near
24981
24982                                 ;; 04/07/2023 - Retro DOS v4.2 IO.SYS (optimized)
24983                                 ;;-------------------------------------------------------        ;I070
24984                                 ;; The following workaround get around a problem with the            ;I070
24985                                 ;; Tortugas and PS/2 30-286 BIOS when password server mode           ;I070
24986                                 ;; is set. On those machines the INT 15h block move code             ;I070
24987                                 ;; goes through the 8042 to twiddle A20 instead of port 92h.         ;I070
24988                                 ;; In password server mode the 8042 is disabled so the block         ;I070
24989                                 ;; move crashes the system. We can do this because these             ;I070
24990                                 ;; systems clear all of memory on a cold boot.                       ;I070
24991                                 ;                                                       ;I070
24992                                 ;               in      al,64h          ; Test for password servr mode ;I070
24993                                 ;               test    al,10h          ; Is keyboard inhibited?      ;I070
24994                                 ;               jnz     short ClrVDISKok ; No, go do block move.      ;I070
24995                                 ;                                       ; Check for Tortugas...       ;I070
24996                                 ;               cmp     word [cs:sys_model_byte],19F8h                ;I070
24997                                 ;               je      short ClrVDISKno                              ;I070
24998                                 ;                                       ; Check for mod 30-286        ;I070
24999                                 ;               cmp     word [cs:sys_model_byte],09FCh                ;I070
25000                                 ;               jne     short ClrVDISKok                ;I070
25001                                 ;ClrVDISKno:       retn                 ; Return w/o block move.           ;I070
25002                                 ;                                                       ;I070
25003                                 ;ClrVDISKok:                                                    ;I070
25004                                 ;-------------------------------------------------------        ;I070
25005
25006                                     ; 30/12/2023 - Retro DOS v5.0
25007                                     ; PCDOS 7.1 IBMBIO.COM - SYSINIT:0DBAh
25008                                 ClrVDISKHeader:
25009 00000CFC E464                     in      al,64h  ; 8042 keyboard controller status register
25010                                                   ; 7:  PERR    1=parity error in data received from keyboard
25011                                                   ;   +---------- AT Mode ----------+------------ PS/2 Mode ------------+
25012                                                   ; 6: |RxTO    receive (Rx) timeout  | TO     general timeout (Rx or Tx) |
25013                                                   ; 5: |TxTO    transmit (Tx) timeout | MOBF   mouse output buffer full   |
25014                                                   ;   +------------------------------+-----------------------------------+
25015                                                   ; 4:  INH     0=keyboard communications inhibited
25016                                                   ; 3:  A2      0=60h was the port last written to, 1=64h was last
25017                                                   ; 2:  SYS     distinguishes reset types: 0=cold reboot, 1=warm reboot
25018                                                   ; 1:  IBF     1=input buffer full (keyboard can't accept data)
25019                                                   ; 0:  OBF     1=output buffer full (data from keyboard is available)
25020 00000CFE A810                     test    al,10h  ; test bit 4 - Is keyboard inhibited?
25021 00000D00 7511                     jnz     short ClrVDISKok ; No, go do block move
25022                                   ; 30/12/2023
25023                                   ; ds = cs
25024 00000D02 813E[BB02]F819           cmp     word [sys_model_byte],19F8h ; check for TORTUGA models
25025 00000D08 7408                     jz      short ClrVDISKno ; do not use INT 15h block move code
25026                                                   ; (while 8042 is disabled)
25027 00000D0A 813E[BB02]FC09           cmp     word [sys_model_byte],9FCh ; check for PS/2 30-286 model
25028 00000D10 7501                     jnz     short ClrVDISKok
25029                                 ClrVDISKno:
25030 00000D12 C3                       retn
25031                                 ; ----------------------------------------------------------------------
25032                                   ; 30/12/2023
25033                                 ClrVDISKok:
25034                                   ; 12/12/2022
25035                                   ; ds = cs
```

```
                                        ; 30/12/2022 - Retro DOS v4.2
                                        ; (MSDOS 6.21 IO.SYS SYSINIT:0CBFh)
00000D13 06                             push    es
00000D14 8CC8                           mov     ax,cs
00000D16 89C2                           mov     dx,ax
00000D18 B10C                           mov     cl,12
00000D1A D3EA                           shr     dx,cl
00000D1C B104                           mov     cl,4
00000D1E D3E0                           shl     ax,cl
00000D20 05[DC0C]                       add     ax,ClrdVDISKHead
00000D23 80D200                         adc     dl,0
                                        ;; 23/10/2022
                                        ;;mov   [cs:src_desc+desc.lo_word],ax
                                        ;mov    [cs:src_desc+2],ax
                                        ;;mov   [cs:src_desc+desc.hi_byte],dl
                                        ;mov    [cs:src_desc+4],dl
                                        ; 12/12/2022
                                        ;mov    [src_desc+desc.lo_word],ax
00000D26 A3[BE0C]                       mov     [src_desc+2],ax
                                        ;mov    [src_desc+desc.hi_byte],dl
00000D29 8816[C00C]                     mov     [src_desc+4],dl
00000D2D B91000                         mov     cx,16   ; 16 words
00000D30 0E                             push    cs
00000D31 07                             pop     es
00000D32 BE[AC0C]                       mov     si,bmove
00000D35 B487                           mov     ah,87h
00000D37 CD15                           int     15h     ; EXTENDED MEMORY - BLOCK MOVE (AT,XT286,PS)
                                                        ; CX = number of words to move
                                                        ; ES:SI -> global descriptor table
                                                        ; Return: CF set on error, AH = status
00000D39 07                             pop     es
00000D3A C3                             retn

                                ; ---------------------------------------------------------------------------
                                ;
                                ; procedure : SaveFreeHMAPtr
                                ;
                                ;           Save the Free HMA pointer in BIOS variable for later use.
                                ;           (INT 2f ax==4a01 call returns pointer to free HMA)
                                ;           Normalizes the pointer to ffff:xxxx format and stores only
                                ;           the offset.
                                ;
                                ; Inputs : ES:DI - pointer to free HMA
                                ; Output : FreeHMAPtr in BIOS data segment updated
                                ;
                                ; ---------------------------------------------------------------------------

                                SaveFreeHMAPtr:
                                    ; 03/09/2023
00000D3B 1E                         push    ds
00000D3C B87000                     mov     ax,DOSBIODATASEG ; 0070h
00000D3F 8ED8                       mov     ds,ax
                                    ;
00000D41 8CC3                       mov     bx,es
00000D43 B8FFFF                     mov     ax,0FFFFh         ; HMA segment
                                    ; 03/09/2023
00000D46 A2[0D00]                   mov     [inHMA],al ; 0FFh  ; (BIOSDATA:000Dh) ; 08/04/2024
                                    ;
00000D49 29D8                       sub     ax,bx
00000D4B 83C70F                     add     di,15             ; para round
00000D4E 83E7F0                     and     di,0FFF0h
00000D51 B104                       mov     cl,4
00000D53 D3E0                       shl     ax,cl
00000D55 29C7                       sub     di,ax
                                    ;
                                    ; 03/09/2023
                                    ;push   ds
                                    ;;mov   ax,Bios_Data ; 0070h
                                    ;mov    ax,KERNEL_SEGMENT ; 0070h
                                    ; 21/10/2022
                                    ; 03/09/2023
                                    ;mov    ax,DOSBIODATASEG ; 0070h
                                    ;mov    ds,ax
                                            ; (BIOSDATA:07D7h for PCDOS 7.1 IBMBIO.COM) ; 08/04/2024
00000D57 893E[D707]                 mov     [FreeHMAPtr],di          ; (ds:8F7h for MSDOS 6.21 IO.SYS)
                                    ;mov    byte [inHMA],0FFh  ; (ds:0Dh)
00000D5B 1F                         pop     ds
00000D5C C3                         retn

                                ; ---------------------------------------------------------------------------
                                ;
                                ; procedure : IsVDiskInstalled
                                ;
                                ;           Checks for the presence of VDISK header at 1MB boundary
                                ;           & INT 19 vector
                                ;
                                ; Inputs  : A20 flag should be ON
                                ; Outputs : Zero set if VDISK header found else Zero cleared
                                ;
                                ; ---------------------------------------------------------------------------

                                IsVDiskInstalled:
00000D5D 31C0                       xor     ax,ax
00000D5F 8ED8                       mov     ds,ax
00000D61 8E1E4E00                   mov     ds,[19*4+2]
                                    ;mov    si,VDiskSig1-StartVDHead ; 12h
                                    ; 23/10/2022
00000D65 BE1200                     mov     si,12h ; 18
                                    ;mov    cx,VLEN1 ; 5
00000D68 B90500                     mov     cx,5
00000D6B 0E                         push    cs
00000D6C 07                         pop     es
00000D6D BF[120C]                   mov     di,VDiskSig1
00000D70 F3A6                       rep     cmpsb
00000D72 740F                       je      short ivdins_retn
00000D74 B8FFFF                     mov     ax,0FFFFh
00000D77 8ED8                       mov     ds,ax
                                    ;mov    si,10h+(VDiskSig2-VDiskHMAHead) ; 13h
00000D79 BE1300                     mov     si,13h
00000D7C BF[370C]                   mov     di,VDiskSig2
                                    ;;mov   cx,VLEN2  ; 5
                                    ;mov    cx,5
                                    ; 03/09/2023
00000D7F B105                       mov     cl,5
00000D81 F3A6                       rep     cmpsb
                                ivdins_retn:
00000D83 C3                         retn                    ; returns the Zero flag

                                ; ---------------------------------------------------------------------------
                                ;
```

```
                                ; procedure : CPMHack
                                ;
                                ;          Copies the code from 0:c0 into ffff:0d0h
                                ;          for CPM compatibility
                                ;
                                ; ---------------------------------------------------------------------------

                                    ; 11/12/2022
                                CPMHack:
00000D84 1E                         push    ds
00000D85 B9FFFF                     mov     cx,0FFFFh
00000D88 8EC1                       mov     es,cx           ; ES = FFFF
                                    ;xor    cx,cx
                                    ; 11/12/2022
00000D8A 41                         inc     cx  ; cx = 0
00000D8B 8ED9                       mov     ds,cx           ; DS = 0
00000D8D BEC000                     mov     si,0C0h
00000D90 BFD000                     mov     di,0D0h
                                    ;mov    cx,5
00000D93 B105                       mov     cl,5
00000D95 FC                         cld
00000D96 F3A4                       rep     movsb           ; move 5 bytes from 0:C0 to FFFF:D0
00000D98 1F                         pop     ds
00000D99 C3                         retn

                                ; ---------------------------------------------------------------------------
                                ;
                                ; procedure : off_to_para
                                ;
                                ; ---------------------------------------------------------------------------
                                off_to_para:
00000D9A D1E8                       shr     ax,1
00000D9C D1E8                       shr     ax,1
00000D9E D1E8                       shr     ax,1
00000DA0 D1E8                       shr     ax,1
00000DA2 C3                         retn

                                ; ---------------------------------------------------------------------------
                                ;** TempCDS - Create (Temporary?) CDS
                                ;
                                ;   ENTRY   ?? BUGBUG
                                ;           (DS) = SysInitSeg
                                ;   EXIT    ?? BUGBUG
                                ;   USES    ?? BUGBUG
                                ; ---------------------------------------------------------------------------

                                    ; 23/10/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 IO.SYS)
                                    ; 30/12/2022 - Retro DOS v4.2 (Modified MSDOS 6.21 IO.SYS)
                                    ; 30/12/2023 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMBIO.COM)
                                TempCDS:
00000DA3 C43E[6D02]                 les     di,[DOSINFO]
00000DA7 268A4D20                   mov     cl,[es:di+SYSI_NUMIO]

                                    ;mov    cl,[es:di+20h]
00000DAB 30ED                       xor     ch,ch                   ; (cx) = # of block devices

00000DAD 26884D21                   mov     [es:di+SYSI_NCDS],cl  ; one CDS per device
                                    ;mov    [es:di+21h],cl

                                    ;mov    al,cl
                                    ;mov    ah,curdirlen ; curdir_list.size ; 88
                                    ;;mov   ah,88
                                    ;mul    ah                      ; (ax) = byte size for those CDSs
                                    ; 30/12/2023
00000DB1 B058                       mov     al,curdirlen ; curdir_list.size ; 88
                                    ;mov    al,88
00000DB3 F6E1                       mul     cl                      ; (ax) = byte size for those CDSs

00000DB5 E85D05                     call    ParaRound               ; (ax) = paragraph size for CDSs
00000DB8 8B36[A702]                 mov     si,[top_of_cdss] ; 31/12/2022

                                ;   BUGBUG - we don't update confbot - won't someone else use it?
                                ;   chuckst -- answer: no. Confbot is used to access the CDSs,
                                ;   25 jul 92  which are stored BELOW it. Alloclim is the
                                ;              variable which has the top of free memory for
                                ;              device driver loads, etc.

00000DBC 29C6                       sub     si,ax

                                ;   chuckst, 25 Jul 92 -- note: I'm removing the code here
                                ;          that automatically updates alloclim every time we
                                ;          set up some new CDSs. Instead, I've added code
                                ;          which pre-allocates space for 26 CDSs. This
                                ;           way we've got room for worst case CDSs before
                                ;          we place MagicDrv.sys
                                ;
                                ;   mov     [ALLOCLIM],si         ; can't alloc past here!

                                    ; 30/12/2022
                                    ; 02/11/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
                                    ; (SYSINIT:0C52h)
                                    ;mov    [ALLOCLIM],si ; (MSDOS 5.0 SYSINIT)

00000DBE 26897518                   mov     [es:di+SYSI_CDS+2],si
                                    ;mov    [es:di+18h],si
00000DC2 89F0                       mov     ax,si
00000DC4 26C745160000               mov     word [es:di+SYSI_CDS],0       ; set address of CDS list
                                    ;mov    [word es:di+16h],0
                                    ;lds    si,[es:di+SYSI_DPB]    ; (ds:si) = address of first DPB
00000DCA 26C535                     lds     si,[es:di]
00000DCD 8EC0                       mov     es,ax
00000DCF 31FF                       xor     di,di                   ; (es:di) = address of 1st CDS

                                ;*  Initialize our temporary CDSs. We'll init each CDS with the
                                ;   info from the corresponding DPB.
                                ;
                                ;   (cx) = count of CDSs left to process
                                ;   (es:di) = address of next CDS

                                fooset:
                                    ; 23/10/2022
00000DD1 2EA1[A902]                 mov     ax,[cs:DirStrng] ; "A:"
00000DD5 AB                         stosw                           ; setup the root as the curdir

                                    ; 23/10/2022 (MSDOS 5.0 SYSINIT)
                                    ;call   get_dpb_for_drive_al  ; get dpb for drive in dpb

                                    ; 30/12/2022
                                    ; (MSDOS 6.21 SYSINIT:0D8Bh)
00000DD6 E85200                     call    get_dpb_for_drive_al  ; get dpb for drive in dpb

                                ;   (ds:si) = address of DPB
                                ;            (si) = -1 if no drive
```

```
25284 00000DD9 2EA1[AB02]               mov     ax,[cs:DirStrng+2] ; "\",0
25285 00000DDD AB                       stosw
25286 00000DDE 2EFE06[A902]             inc     byte [cs:DirStrng]
25287 00000DE3 31C0                     xor     ax,ax ; 0
25288 00000DE5 51                       push    cx
25289                                   ;mov    cx,curdir_list.cdir_flags - 4 ; 63
25290 00000DE6 B93F00                   mov     cx,63   ; 23/10/2022
25291 00000DE9 F3AA                     rep     stosb                   ; zero out rest of CURDIR_TEXTs
25292
25293                           ;       should handle the system that does not have any floppies.
25294                           ;       in this case,we are going to pretended there are two dummy floppies
25295                           ;       in the system. still they have dpb and cds,but we are going to
25296                           ;       0 out curdir_flags,curdir_devptr of cds so ibmdos can issue
25297                           ;       "invalid drive specification" message when the user try to
25298                           ;       access them.
25299                           ;
25300                           ;       (ax) = 0
25301                           ;       (es:di) = CURDIR_FLAGS in the CDS records
25302                           ;       (ds:si) = Next DPB (-1 if none)
25303
25304 00000DEB 83FEFF                     cmp     si,-1   ; cmp si,0FFFFh
25305 00000DEE 740C                     je      short fooset_zero       ; don't have any physical drive.
25306
25307                           ;       check to see if we are faking floppy drives. if not go to normcds.
25308                           ;       if we are faking floppy drives then see if this cds being initialised
25309                           ;       is for drive a: or b: by checking the appropriate field in the dpb
25310                           ;       pointed to by ds:si. if not for a: or b: then go to normcds. if
25311                           ;       for a: or b: then execute the code given below starting at fooset_zero.
25312                           ;       for dpb offsets look at inc\dpb.inc.
25313
25314                                   ; 03/09/2023
25315 00000DF0 41                       inc     cx   ; cx = 1
25316
25317 00000DF1 2E380E[8B02]             cmp     [cs:fake_floppy_drv],cl ; 1 ; 03/09/2023
25318                                   ;cmp    byte [cs:fake_floppy_drv],1
25319 00000DF6 750A                     jne     short normcds           ; machine has floppy drives
25320                                   ;cmp    byte [si+DPB.drive],1 ; if dpb_drive = 0 (a) or 1 (b).
25321                                   ;cmp    byte [si],1
25322 00000DF8 380C                     cmp     [si],cl ; 1 ; 03/09/2023
25323 00000DFA 7706                     ja      short normcds
25324
25325                                   ; 30/12/2023
25326                                   ; ax = 0
25327                           fooset_zero:
25328 00000DFC B103                     mov     cl,3                    ; the next dbp pointer
25329                                                                   ; AX should be zero here
25330 00000DFE F3AB                     rep     stosw
25331                                   ; 30/12/2023
25332                                   ;pop    cx
25333 00000E00 EB0F                     jmp     short get_next_dpb ; findcds
25334
25335                           ;       (ax) = 0
25336
25337                                   ; 30/12/2023
25338                           ;fooset_zero:
25339                                   ;mov    cl,3
25340                                   ;rep    stosw
25341                                   ;pop    cx
25342                                   ;jmp    short fincds
25343
25344                           ;*      We have a "normal" DPB and thus a normal CDS.
25345                           ;
25346                           ;       (ax) = 0
25347                           ;       (es:di) = CURDIR_FLAGS in the CDS records
25348                           ;       (ds:si) = Next DPB (-1 if none)
25349
25350                           normcds:
25351                                   ; 30/12/2023
25352                                   ;pop    cx
25353
25354                           ;       if a non-fat based media is detected (by dpb.numberoffat == 0), then
25355                           ;       set curdir_flags to 0. this is for signaling ibmdos and ifsfunc that
25356                           ;       this media is a non-fat based one.
25357
25358                                   ;cmp    byte [si+DPB.FAT_COUNT],0 ; non fat system?
25359                                   ; 23/10/2022
25360                                   ;cmp    byte [si+8],0
25361                                   ; 03/09/2023 (ax=0)
25362 00000E02 384408                   cmp     [si+8],al ; 0
25363 00000E05 7403                     je      short setnormcds        ; yes. set curdir_flags to 0. ax = 0 now.
25364 00000E07 B80040                   mov     ax,curdir_inuse ; 4000h         ; else,fat system. set the flag to curdir_inuse.
25365                                   ;mov    ax,4000h
25366                           setnormcds:
25367 00000E0A AB                       stosw                           ; curdir_flags
25368 00000E0B 89F0                     mov     ax,si
25369 00000E0D AB                       stosw                           ; curdir_devptr
25370 00000E0E 8CD8                     mov     ax,ds
25371 00000E10 AB                       stosw
25372
25373                           get_next_dpb:                           ; entry point for fake_fooset_zero
25374                                   ; 30/12/2022
25375                                   ; (MSDOS 6.21 SYSINIT:0DD1h)
25376                                   ; 23/10/2022
25377                                   ;lds    si,[si+19h] ; (MSDOS 5.0 SYSINIT)
25378                                   ;;lds   si,[si+DPB.NEXT_DPB] ; [si+19h]
25379                           fincds:     ; get_next_dpb
25380                                   ; 30/12/2023
25381 00000E11 59                       pop     cx
25382                                   ; 30/12/2022
25383                                   ; (MSDOS 6.21 SYSINIT:0DD1h)
25384 00000E12 B8FFFF                   mov     ax,-1   ; mov ax,0FFFFh
25385 00000E15 AB                       stosw                           ; curdir_id
25386 00000E16 AB                       stosw                           ; curdir_id
25387 00000E17 AB                       stosw                           ; curdir_user_word
25388 00000E18 B80200                   mov     ax,2
25389 00000E1B AB                       stosw                           ; curdir_end
25390 00000E1C B000                     mov     al,0                    ; clear out 7 bytes (curdir_type,
25391 00000E1E AA                       stosb
25392 00000E1F AB                       stosw                           ;  curdir_ifs_hdr,curdir_fsda)
25393 00000E20 AB                       stosw
25394 00000E21 AB                       stosw
25395
25396 00000E22 E2AD                     loop    fooset
25397
25398 00000E24 2EC606[A902]41           mov     byte [cs:DirStrng],"A"; "A:\"
25399
25400 00000E2A C3                       retn
25401
25402                           ; ----------------------------------------------------------------------
25403                           ;***get_dpb_for_drive_al -- lookup the DPB for drive in al
25404                           ;
25405                           ;       entry:
25406                           ;          al == ASCII CAPS drive letter
25407                           ;
```

```
                                ;   exit:
                                ;      ds:si -> DPB, or si = -1 if not found
                                ; ----------------------------------------------------------------------

                                ; 30/12/2023
                                ; PCDOS 7.1 IBMBIO.COM - SYSINIT:0EFEh

                                    ; 30/12/2022
                                    ; (MSDOS 6.21 SYSINIT:0DEAh)
                                    ; 23/10/2022
                                get_dpb_for_drive_al:
00000E2B 2EC536[6D02]               lds     si,[cs:DOSINFO]             ; point to first DPB
                                    ;lds    si,[si+SYSI_DPB]      ; (ds:si) = address of first DPB
00000E30 C534                       lds     si,[si]
00000E32 2C41                       sub     al,'A'

                                get_dpb_for_drive_1:
                                    ;cmp    al,[si+DPB.DRIVE]     ; match?
00000E34 3A04                       cmp     al,[si]
00000E36 7408                       je      short got_dpb_for_drive       ;  done if so

00000E38 C57419                     lds     si,[si+DPB.NEXT_DPB] ; [si+19h]
00000E3B 83FEFF                     cmp     si,-1
00000E3E 75F4                       jne     short get_dpb_for_drive_1 ; loop until hit end of DPBs

                                got_dpb_for_drive:
00000E40 C3                         retn

                                ;============================================================================

                                ;** EndFile - Build DOS structures
                                ;
                                ; This procedure is called after the config.sys has been processed and
                                ; installable device drivers have been loaded (but before "install="
                                ; programs are loaded) to create the dos structures such as SFTs, buffers,
                                ; FCBs, CDSs, etc. It also loads the sysinit_base module in low memory
                                ; to allow for the safe EXECing of "install=" programs. All memory
                                ; above these structures is deallocated back to DOS.
                                ;
                                ;   ENTRY   ?? BUGBUG
                                ;   EXIT    ?? BUGBUG
                                ;   USES    ?? BUGBUG

                                ;============================================================================
                                ; allocate files
                                ; ----------------------------------------------------------------------

                                    ; 23/10/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 IO.SYS)
                                    ; (SYSINIT:0CCDh)

                                    ; 30/12/2022 - Retro DOS v4.2 (Modified MSDOS 6.21 IO.SYS)
                                    ; (SYSINIT:0E00h)

                                    ; 09/04/2024 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMBIO.COM)
                                    ; (PCDOS 7.1 IBMBIO.COM - SYSINIT:0F14h)

                                    ; ((MSDOS 6.22 IO.SYS - SYSINIT:0E00h))

                                endfile:
                                ; we are now setting up final cdss,buffers,files,fcss strings etc. we no
                                ; longer need the space taken by the temp stuff below confbot,so set alloclim
                                ; to confbot.

                                ;   if this procedure has been called to take care of install= command,
                                ;      then we have to save es,si registers.

                                    ; 11/12/2022
                                    ; ds = cs

                                    ; 23/10/2022
                                    ; 31/03/2019
00000E41 1E                         push    ds

                                    ;;mov   ax,Bios_Data ; 0070h
                                    ;mov    ax,KERNEL_SEGMENT ; 0070h
                                    ; 21/10/2022
00000E42 B87000                     mov     ax,DOSBIODATASEG ; 0070h
00000E45 8ED8                       mov     ds,ax

                                    ;cmp    word [052Fh],0
00000E47 833E[A004]00               cmp     word [multrk_flag],multrk_off1 ;=0,multrack= command entered?
00000E4C 7505                       jne     short multrk_flag_done
                                    ; 02/11/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
                                    ;or     word [multrk_flag],multrk_on ; 80h  ; default will be on.
                                    ; 12/12/2022
00000E4E 800E[A004]80               or      byte [multrk_flag],multrk_on ; 80h
                                multrk_flag_done:
                                    ; 23/10/2022
                                    ; 31/03/2019
00000E53 1F                         pop     ds

                                    ; 11/12/2022
                                    ; ds = cs
                                    ;mov    ax,[top_of_cdss] ; mov ax,[CONFBOT]
                                    ; 31/12/2022 - Retro DOS v4.2 (Modified MSDOS 6.21 IO.SYS)
                                    ; (SYSINIT:0E14h)
00000E54 A1[A302]                   mov     ax,[CONFBOT]
00000E57 A3[A502]                   mov     [ALLOCLIM],ax
                                    ; 23/10/2022
                                    ;mov    ax, [cs:top_of_cdss]
                                    ;mov    [cs:ALLOCLIM], ax

                                    ; 11/12/2022
                                    ; ds = cs
                                    ;push   cs
                                    ;pop    ds

                                    ;mov    ax,[CONFBOT]
                                    ;mov    [ALLOCLIM],ax

                                    ; 09/04/2024 - Retro DOS v5.0 (PCDOS 7.1 IBMBIO.COM)
                                    ;;;
                                    ;;mov   ax,[cs:ALLOCLIM]
                                    ;mov    ax,[ALLOCLIM]
                                    ;mov    [cs:prev_alloclim],ax
00000E5A A3[6C03]                   mov     [prev_alloclim],ax
                                    ;mov    ax,[cs:memhi]
00000E5D A1[6403]                   mov     ax,[memhi]
                                    ;mov    [cs:prev_memhi],ax
00000E60 A3[6A03]                   mov     [prev_memhi],ax
                                dosfts:
                                    ;;;

00000E63 E88C39                     call    round
```

```
                                ; 11/12/2022
                                ; ds = cs
00000E66 A0[9F02]               mov     al,[FILES]
                                ; 23/10/2022
                                ;mov    al,[cs:FILES]
00000E69 2C05                   sub     al,5
00000E6B 764B                   jbe     short dofcbs

00000E6D 50                     push    ax
                                ;mov    al,devmark_files ; 'F'
00000E6E B046                   mov     al,'F'
00000E70 E81808                 call    setdevmark              ; set devmark for sfts (files)
00000E73 58                     pop     ax
00000E74 30E4                   xor     ah,ah                   ; do not use cbw instruction!!!!!
                                                                ;  it does sign extend.
                                ; 11/12/2022
                                ; ds = cs
00000E76 8B1E[6203]             mov     bx,[memlo]
00000E7A 8B16[6403]             mov     dx,[memhi]
00000E7E C53E[6D02]             lds     di,[DOSINFO]            ;get pointer to dos data
                                ; 23/10/2022
                                ;mov    bx,[cs:memlo]
                                ;mov    dx,[cs:memhi]
                                ;lds    di,[cs:DOSINFO]

                                ;lds    di,[di+SYSI_SFT]        ;ds:bp points to sft
00000E82 C57D04                 lds     di,[di+4]

                                ;mov    [di+SF.SFLink],bx
00000E85 891D                   mov     [di],bx
00000E87 895502                 mov     [di+SF.SFLink+2],dx     ;set pointer to new sft

00000E8A 0E                     push    cs
00000E8B 1F                     pop     ds

                                ; 11/12/2022
                                ; ds = cs
00000E8C C43E[6203]             les     di,[memlo]              ;point to new sft
                                ; 23/10/2022
                                ;les    di,[cs:memlo]

                                ;mov    word [es:di+SF.SFLink],-1
00000E90 26C705FFFF             mov     word [es:di],-1                 ; 0FFFFh
                                ;mov    [es:di+SF.SFCount],ax
00000E95 26894504               mov     [es:di+4],ax
                                ; 09/04/2024
00000E99 B33B                   mov     bl,SF_ENTRY.size ; 59
                                ;mov    bl,59
00000E9B F6E3                   mul     bl                      ;ax = number of bytes to clear
00000E9D 89C1                   mov     cx,ax
                                ; 11/12/2022
                                ; ds = cs
00000E9F 0106[6203]             add     [memlo],ax              ;allocate memory
                                ; 23/10/2022
                                ;add    [cs:memlo],ax
00000EA3 B80600                 mov     ax,6
                                ; 11/12/2022
00000EA6 0106[6203]             add     [memlo],ax              ;remember the header too
                                ;add    [cs:memlo],ax
                                ; 11/12/2022
00000EAA 800E[6919]02           or      byte [setdevmarkflag],for_devmark ; 2
                                ; 23/10/2022
                                ;or     byte [cs:setdevmarkflag],2
00000EAF E84039                 call    round                   ; check for mem error before the stosb
00000EB2 01C7                   add     di,ax
00000EB4 31C0                   xor     ax,ax
00000EB6 F3AA                   rep     stosb                   ;clean out the stuff

                                ; allocate fcbs
                                ; ----------------------------------------------------------------------

                                    ; 23/10/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 IO.SYS)
                                    ; (SYSINIT:0D48h)
                                dofcbs:
                                    ; 11/12/2022
                                    ; ds = cs
                                    ;push   cs
                                    ;pop    ds
00000EB8 E83739                     call    round
                                    ;mov    al,devmark_fcbs         ; 'X'   ;='x'
00000EBB B058                       mov     al,'X'
00000EBD E8CB07                     call    setdevmark
                                    ; 11/12/2022
                                    ; ds = cs
00000EC0 A0[A002]                   mov     al,[FCBS]
                                    ;mov    al,[cs:FCBS]
00000EC3 30E4                       xor     ah,ah                   ; do not use cbw instruction!!!!!
                                                                    ;  it does sign extend.
                                    ; 11/12/2022
00000EC5 8B1E[6203]                 mov     bx,[memlo]
00000EC9 8B16[6403]                 mov     dx,[memhi]
00000ECD C53E[6D02]                 lds     di,[DOSINFO]            ;get pointer to dos data
                                    ; 23/10/2022
                                    ;mov    bx,[cs:memlo]
                                    ;mov    dx,[cs:memhi]
                                    ;lds    di,[cs:DOSINFO]

                                    ;mov    [di+SYSI_FCB],bx
                                    ;mov    [di+SYSI_FCB+2],dx ;set pointer to new table
                                    ; 23/10/2022
00000ED1 895D1A                     mov     [di+1Ah],bx             ; [di+SYSI_FCB]
00000ED4 89551C                     mov     [di+1Ch],dx             ; [di+SYSI_FCB+2]

00000ED7 2E8A1E[A102]               mov     bl,[cs:KEEP]
00000EDC 30FF                       xor     bh,bh
                                    ;mov    [di+SYSI_KEEP],bx
00000EDE 895D1E                     mov     [di+1Eh],bx             ; [di+SYSI_KEEP]

00000EE1 0E                         push    cs
00000EE2 1F                         pop     ds

00000EE3 C43E[6203]                 les     di,[memlo]              ;point to new table
                                    ;mov    word [es:di+SF.SFLink],-1
00000EE7 26C705FFFF                 mov     word [es:di],-1
                                    ;mov    [es:di+SF.SFCount],ax
                                    ; 02/11/2022
00000EEC 26894504                   mov     [es:di+4],ax
00000EF0 B33B                       mov     bl,SF_ENTRY.size ; 59
00000EF2 89C1                       mov     cx,ax
00000EF4 F6E3                       mul     bl                      ;ax = number of bytes to clear
00000EF6 0106[6203]                 add     [memlo],ax              ;allocate memory
                                    ;mov    ax,6
00000EFA B80600                     mov     ax,SF.size-2 ; 6
```

```
25656 00000EFD 0106[6203]               add     [memlo],ax              ;remember the header too
25657                                   ;or     byte [setdevmarkflag],for_devmark ; 2
25658 00000F01 800E[6919]02             or      byte [setdevmarkflag],2
25659 00000F06 E8E938                   call    round                   ; check for mem error before the stosb
25660 00000F09 01C7                     add     di,ax                   ;skip over header
25661 00000F0B B041                     mov     al,'A'
25662                                 fillloop:
25663 00000F0D 51                       push    cx                      ; save count
25664 00000F0E B93B00                   mov     cx,SF_ENTRY.size ; 59 ; number of bytes to fill
25665 00000F11 FC                       cld
25666 00000F12 F3AA                     rep     stosb                   ; filled
25667
25668                                   ;mov    word [es:di-(SF_ENTRY.size)+SF_ENTRY.sf_ref_count],0  ; [es:di-59]
25669                                   ;mov    word [es:di-(SF_ENTRY.size)+SF_ENTRY.sf_position],0    ; [es:di-38]
25670                                   ;mov    word [es:di-(SF_ENTRY.size)+SF_ENTRY.sf_position+2],0 ; [es:di-36]
25671
25672                                   ; 18/12/2022
25673                                   ;cx = 0
25674 00000F14 26894DC5                 mov     [es:di-(SF_ENTRY.size)+SF_ENTRY.sf_ref_count],cx ;0  ; [es:di-59]
25675 00000F18 26894DDA                 mov     [es:di-(SF_ENTRY.size)+SF_ENTRY.sf_position],cx ;0   ; [es:di-38]
25676 00000F1C 26894DDC                 mov     [es:di-(SF_ENTRY.size)+SF_ENTRY.sf_position+2],cx ;0 ; [es:di-36]
25677
25678                                   ; 23/10/2022
25679                                   ;mov     word [es:di-3Bh],0
25680                                   ;mov     word [es:di-26h],0
25681                                   ;mov     word [es:di-24h],0
25682
25683 00000F20 59                       pop     cx
25684 00000F21 E2EA                     loop    fillloop
25685
25686                                 ; allocate buffers
25687                                 ; ---------------------------------------------------------------------------
25688
25689                                 ; search through the list of media supported and allocate 3 buffers if the
25690                                 ; capacity of the drive is > 360kb
25691
25692                                   ; 18/12/2022
25693                                   ; cx = 0
25694 00000F23 833E[9902]FF             cmp     word [buffers],-1       ; has buffers been already set?
25695 00000F28 7403                     je      short dodefaultbuff
25696 00000F2A E98000                   jmp     dobuff                  ; the user entered the buffers=.
25697
25698                                 dodefaultbuff:
25699                                   ; 18/12/2022
25700 00000F2D 890E[9B02]               mov     [h_buffers],cx ; 0
25701                                   ;inc    cx
25702                                   ;inc    cx
25703                                   ;mov    [buffers],cx ; 2
25704                                   ; 10/04/2024
25705 00000F31 C706[9902]0200           mov     word [buffers],2
25706
25707                                   ;mov    word [h_buffers],0      ; default is no heuristic buffers.
25708                                   ;mov    word [buffers],2        ; default to 2 buffers
25709
25710                                   ; 23/10/2022
25711                                   ; 04/09/2023
25712                                   ;push   ax
25713                                   ;push   ds ; 26/03/2019
25714
25715                                   ; 04/09/2023
25716                                   ; ds = cs
25717 00000F37 C42E[6D02]               les     bp,[DOSINFO]            ; search through the dpb's
25718                                   ;les    bp,[cs:DOSINFO]
25719                                   ;les    bp,[es:bp+SYSI_DPB]     ; get first dpb
25720                                   ; 11/12/2022
25721 00000F3B 26C46E00                 les     bp,[es:bp]
25722                                   ; 23/10/2022
25723                                   ;les    bp,[es:bp+0]    ; ! (MSDOS 5.0 IO.SYS address compability) !
25724
25725                                   ; 04/09/2023
25726                                   ; ds = cs
25727                                   ;push   cs
25728                                   ;pop    ds
25729                                 ;SYSINIT:0DE2h:
25730                                 nextdpb:                          ; test if the drive supports removeable media
25731                                   ;mov    bl,[es:bp+DPB.drive]
25732                                   ; 11/12/2022
25733 00000F3F 268A5E00                 mov     bl,[es:bp]
25734                                   ; 23/10/2022
25735                                   ;mov    bl,[es:bp+0]    ; ! (MSDOS 5.0 IO.SYS address compability) !
25736
25737                                   ;inc    bl
25738                                   ; 18/12/2022
25739 00000F43 43                       inc     bx
25740
25741                                   ;mov    ax,(IOCTL<<8)|8
25742 00000F44 B80844                   mov     ax,4408h
25743 00000F47 CD21                     int     21h             ; DOS - 2+ - IOCTL -
25744
25745                                 ; ignore fixed disks
25746
25747 00000F49 09C0                     or      ax,ax                   ; ax is nonzero if disk is nonremoveable
25748 00000F4B 7534                     jnz     short nosetbuf
25749
25750                                 ; get parameters of drive
25751
25752 00000F4D 31DB                     xor     bx,bx
25753                                   ;;mov   bl,[es:bp+DPB.drive]
25754                                   ; 11/12/2022
25755 00000F4F 268A5E00                 mov     bl,[es:bp]
25756                                   ; 23/10/2022
25757                                   ;mov    bl,[es:bp+0]    ; ! (MSDOS 5.0 IO.SYS address compability) !
25758
25759                                   ;inc    bl
25760                                   ; 18/12/2022
25761 00000F53 43                       inc     bx
25762
25763 00000F54 BA[BE4D]                 mov     dx,deviceparameters
25764                                   ;mov    ax,(IOCTL<<8)|GENERIC_IOCTL
25765 00000F57 B80D44                   mov     ax,440Dh
25766                                   ;mov    cx,(RAWIO<<8)|GET_DEVICE_PARAMETERS
25767 00000F5A B96008                   mov     cx,860h
25768 00000F5D CD21                     int     21h             ; DOS - 2+ - IOCTL -
25769 00000F5F 7220                     jc      short nosetbuf          ; get next dpb if driver doesn't support
25770                                                                   ; generic ioctl
25771                                 ; determine capacity of drive
25772                                 ; media capacity = #sectors * bytes/sector
25773
25774                                   ;mov    bx,[deviceparameters+A_DEVICEPARAMETERS.DP_BPB+A_BPB.TOTALSECTORS]
25775                                   ; 23/10/2022
25776 00000F61 8B1E[CD4D]               mov     bx,[deviceparameters+15] ; total sectors (16 bit)
25777
25778                                 ; to keep the magnitude of the media capacity within a word,
25779                                 ; scale the sector size
```

```
                                ; (ie. 1 -> 512 bytes,2 -> 1024 bytes,...)
                                
                                    ;mov    ax,[deviceparameters+A_DEVICEPARAMETERS.DP_BPB+A_BPB.BYTESPERSECTOR]
                                    ; 23/10/2022
00000F65 A1[C54D]                   mov     ax,[deviceparameters+7] ; bytes per sector
00000F68 31D2                       xor     dx,dx
00000F6A B90002                     mov     cx,512
00000F6D F7F1                       div     cx                      ; scale sector size in factor of
                                                                    ; 512 bytes
00000F6F F7E3                       mul     bx                      ; ax = #sectors * size factor
00000F71 09D2                       or      dx,dx                   ; just in case of large floppies
00000F73 7505                       jnz     short setbuf
00000F75 3DD002                     cmp     ax,720                  ; 720 sectors * size factor of 1
00000F78 7607                       jbe     short nosetbuf
                                setbuf:
                                    ; 18/12/2022
                                    ; word [buffers] = 2
00000F7A C606[9902]03               mov     byte [buffers],3
                                    ;mov    word [buffers],3
00000F7F EB0D                       jmp     short chk_memsize_for_buffers ; now check the memory size
                                                                    ; for default buffer count
                                nosetbuf:
                                    ; 23/10/2022
                                    ;cmp    word [es:bp+DPB.NEXT_DPB],-1
00000F81 26837E19FF                 cmp     word [es:bp+19h], -1 ; 0FFFFh
00000F86 7406                       je      short chk_memsize_for_buffers
                                    ;les    bp,[es:bp+DPB.NEXT_DPB] ; [es:bp+19h]
00000F88 26C46E19                   les     bp,[es:bp+19h]
00000F8C EBB1                       jmp     short nextdpb
                                
                                ;from dos 3.3,the default number of buffers will be changed according to the
                                ;memory size too.
                                ; default buffers = 2
                                ; if diskette media > 360 kb,then default buffers = 3
                                ; if memory size > 128 kb (2000h para),then default buffers = 5
                                ; if memory size > 256 kb (4000h para),then default buffers = 10
                                ; if memory size > 512 kb (8000h para),then default buffers = 15.
                                
                                chk_memsize_for_buffers:
                                    ; 18/12/2022
                                    ;cmp    word [MEMORY_SIZE],2000h
                                    ;jbe    short bufset
                                    ;mov    word [buffers],5
                                    ;cmp    word [MEMORY_SIZE],4000h
                                    ;jbe    short bufset
                                    ;mov    word [buffers],10
                                    ;cmp    word [MEMORY_SIZE],8000h
                                    ;jbe    short bufset
                                    ;mov    word [buffers],15
                                
                                    ; 18/12/2022
                                    ; word [buffers] = 3 or 2
00000F8E BB[9902]                   mov     bx,buffers
00000F91 A1[9402]                   mov     ax,[MEMORY_SIZE]
00000F94 48                         dec     ax      ; [MEMORY_SIZE] - 1
                                
00000F95 80FC20                     cmp     ah,20h  ; ax >= 2000h ([MEMORY_SIZE] > 2000h) ; *
00000F98 7213                       jb      short bufset
00000F9A C6070F                     mov     byte [bx],15 ; [buffers] = 15 ; ***
00000F9D 80FC80                     cmp     ah,80h  ; ax >= 8000h ([MEMORY_SIZE] > 8000h) ; ***
00000FA0 730B                       jnb     short bufset
00000FA2 C6070A                     mov     byte [bx],10 ; [buffers] = 10 ; **
00000FA5 80FC40                     cmp     ah,40h  ; ax >= 4000h ([MEMORY_SIZE] > 4000h) ; **
00000FA8 7303                       jnb     short bufset
00000FAA C60705                     mov     byte [bx],5  ; [buffers] = 5 ; *
                                bufset:
                                    ; 23/10/2022
                                    ; 26/03/2019
                                    ; 04/09/2023
                                    ;pop    ds
                                    ;pop    ax
                                
                                ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
                                ;j.k. here we should put extended stuff and new allocation scheme!!!
                                ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
                                
                                ; 26/03/2019
                                
                                ;*****************************************************************************
                                ;                                                                            *
                                ; function: actually allocate buffers in the memory and initialize it.       *
                                ; input :                                                                     *
                                ;    memhi:memlo - start of the next available memory                        *
                                ;    buffers = number of buffers                                             *
                                ;    h_buffers = number of secondary buffers                                 *
                                ;                                                                            *
                                ; output:                                                                    *
                                ;   buffinfo.cache_count - # of caches to be installed.                      *
                                ;   buffinfo set.                                                            *
                                ;   bufferqueue set.                                                        *
                                ;                                                                            *
                                ; subroutines to be called:                                                  *
                                ;                                                                            *
                                ;*****************************************************************************
                                
                                    ; 23/10/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 IO.SYS)
                                    ; (SYSINIT:0E60h)
                                dobuff:
                                    ; ds = cs ; 31/03/2019
                                    ; 23/10/2022
                                    ;lds    bx,[cs:DOSINFO]         ; ds:bx -> sysinitvar
                                    ; 04/09/2023
00000FAD A1[9902]                   mov     ax,[buffers] ; 31/03/2019
00000FB0 8B0E[9B02]                 mov     cx,[h_buffers] ; *
00000FB4 C51E[6D02]                 lds     bx,[DOSINFO]
                                    ;mov    ax,[cs:buffers]         ; set sysi_buffers
                                    ;mov    [bx+SYSI_BUFFERS],ax ; [bx+3Fh]
00000FB8 89473F                     mov     [bx+3Fh],ax
                                    ; 04/09/2023
                                    ;mov    ax,[cs:h_buffers]
                                    ;;mov   [bx+SYSI_BUFFERS+2],ax ; [bx+41h]
                                    ;mov    [bx+41h],ax
                                    ; 04/09/2023
00000FBB 894F41                     mov     [bx+41h],cx ; *
00000FBE C55F12                     lds     bx,[bx+12h]
                                    ;lds    bx,[bx+SYSI_BUF] ; now,ds:bx -> buffinfo
00000FC1 E82E38                     call    round           ; get [memhi]:[memlo]
                                    ;mov    al,devmark_buf ; ='B'
00000FC4 B042                       mov     al,'B'
00000FC6 E8C206                     call    setdevmark
                                
                                ;allocate buffers
                                
00000FC9 1E                         push    ds                      ; save buffer info. ptr.
```

```
25904 00000FCA 53                       push    bx
25905
25906 00000FCB E8D403                   call    set_buffer
25907
25908 00000FCE 5B                       pop     bx
25909 00000FCF 1F                       pop     ds
25910
25911                                ;now set the secondary buffer if specified.
25912
25913 00000FD0 2E833E[9B02]00           cmp     word [cs:h_buffers],0
25914 00000FD6 742D                     je      short xif16
25915 00000FD8 E81738                   call    round
25916                                   ; 23/10/2022
25917 00000FDB 2E8B0E[6203]             mov     cx,[cs:memlo]
25918                                   ;mov    [bx+BUFFINF.Cache_ptr],cx  ; [bx+6]
25919 00000FE0 894F06                   mov     [bx+6],cx
25920 00000FE3 2E8B0E[6403]             mov     cx,[cs:memhi]
25921                                   ;mov    [bx+BUFFINF.Cache_ptr+2],cx ; [bx+8]
25922 00000FE8 894F08                   mov     [bx+8],cx
25923 00000FEB 2E8B0E[9B02]             mov     cx,[cs:h_buffers]
25924                                   ;mov    [bx+BUFFINF.Cache_count],cx ; [bx+10]
25925 00000FF0 894F0A                   mov     [bx+10],cx
25926 00000FF3 B80002                   mov     ax,512                  ; 512 byte
25927 00000FF6 F7E1                     mul     cx
25928 00000FF8 2EA3[6203]               mov     [cs:memlo],ax
25929                                   ;or     byte [cs:setdevmarkflag],for_devmark ; 2
25930 00000FFC 2E800E[6919]02           or      byte [cs:setdevmarkflag],2
25931 00001002 E8ED37                   call    round
25932                                xif16:
25933
25934                                ; ------------------------------------------------------------------------
25935                                ; allocate cdss
25936                                ; ------------------------------------------------------------------------
25937
25938                                buf1:
25939 00001005 E8EA37                   call    round
25940
25941 00001008 50                        push    ax
25942                                   ; 23/10/2022
25943                                   ;mov    ax,devmark_cds          ;='L'
25944 00001009 B84C00                   mov     ax, 'L'
25945 0000100C E87C06                   call    setdevmark
25946 0000100F 58                       pop     ax
25947
25948 00001010 2EC43E[6D02]             les     di,[cs:DOSINFO]
25949                                   ;mov    cl,[es:di+SYSI_NUMIO]
25950 00001015 268A4D20                 mov     cl,[es:di+20h]
25951 00001019 2E3A0E[A202]             cmp     cl,[cs:NUM_CDS]
25952 0000101E 7305                     jae     short gotncds           ; user setting must be at least numio
25953 00001020 2E8A0E[A202]             mov     cl,[cs:NUM_CDS]
25954                                gotncds:
25955 00001025 30ED                     xor     ch,ch
25956                                   ;mov    [es:di+SYSI_NCDS],cl  ; [es:di+33]
25957 00001027 26884D21                 mov     [es:di+21h],cl
25958 0000102B 2EA1[6403]               mov     ax,[cs:memhi]
25959                                   ;mov    [es:di+SYSI_CDS+2],ax
25960 0000102F 26894518                 mov     [es:di+18h],ax
25961 00001033 2EA1[6203]               mov     ax,[cs:memlo]
25962                                   ;mov    [es:di+SYSI_CDS],ax
25963 00001037 26894516                 mov     [es:di+16h],ax
25964 0000103B 88C8                     mov     al,cl
25965                                   ;mov    ah,curdirlen ; curdir_list.size
25966 0000103D B458                     mov     ah,88
25967 0000103F F6E4                     mul     ah
25968 00001041 E8D102                   call    ParaRound
25969 00001044 2E0106[6403]             add     [cs:memhi],ax
25970
25971                                   ;or     byte [cs:setdevmarkflag],for_devmark ; 2
25972 00001049 2E800E[6919]02           or      byte [cs:setdevmarkflag],2
25973 0000104F E8A037                   call    round                   ; check for mem error before initializing
25974                                   ;lds    si,[es:di+SYSI_DPB] ; [es:di+0]
25975 00001052 26C535                   lds     si,[es:di]
25976                                   ;les    di,[es:di+SYSI_CDS] ; [es:di+22]
25977 00001055 26C47D16                 les     di,[es:di+16h]
25978 00001059 E875FD                   call    fooset
25979
25980                                ; ------------------------------------------------------------------------
25981                                ; allocate space for internal stack
25982                                ; ------------------------------------------------------------------------
25983
25984 0000105C 0E                        push    cs
25985 0000105D 1F                       pop     ds
25986
25987                                ;   if the user did not entered stacks= command, as a default, do not install
25988                                ;   sytem stacks for pc1,pc xt,pc portable cases.
25989                                ;   otherwise,install it to the user specified value or to the default
25990                                ;   value of 9,128 for other systems.
25991
25992 0000105E 833E[9002]FF              cmp     word [stack_addr],-1 ; has the user entered "stacks=" command?
25993 00001063 740E                     je      short doinstallstack  ; then install as specified by the user
25994 00001065 803E[BC02]00             cmp     byte [sys_scnd_model_byte],0 ; pc1,xt has the secondary model byte = 0
25995 0000106A 7507                     jne     short doinstallstack  ; other model should have default stack of 9,128
25996 0000106C 803E[BB02]FE             cmp     byte [sys_model_byte],0FEh ; pc1, pc/xt or pc portable ?
25997 00001071 736D                     jae     short skipstack
25998                                doinstallstack:
25999 00001073 A1[8C02]                 mov     ax,[stack_count]        ; stack_count = 0?
26000 00001076 09C0                     or      ax,ax                   ; then, stack size must be 0 too.
26001 00001078 7466                     jz      short skipstack                 ; don't install stack.
26002
26003                                ;   dynamic relocation of stack code.
26004
26005 0000107A E87537                   call    round                   ;[memhi] = seg. for stack code
26006                                                                   ;[memlo] = 0
26007
26008                                ; set devmark block into memory for mem command
26009                                ; devmark_id = 's' for stack
26010
26011                                   ;mov    al,devmark_stk ;='S'
26012                                   ; 23/10/2022
26013 0000107D B053                     mov     al,'S'
26014 0000107F E80906                   call    setdevmark
26015
26016 00001082 A1[6403]                 mov     ax,[memhi]
26017 00001085 8EC0                     mov     es,ax           ;es -> seg. the stack code is going to move.
26018                                   ; 02/11/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
26019                                   ; 11/12/2022
26020                                   ; ds = cs
26021                                   ;push   cs
26022                                   ;pop    ds
26023 00001087 31F6                     xor     si,si           ;!!we know that stack code is at the beginning of sysinit.
26024 00001089 31FF                     xor     di,di
26025 0000108B B9[6902]                 mov     cx,endstackcode
26026 0000108E 890E[6203]               mov     [memlo],cx
26027 00001092 E85D37                   call    round           ;have enough space for relocation?
```

```
26028 00001095 F3A4                     rep     movsb
26029
26030 00001097 1E                       push    ds              ; stick the location of the NextStack entry
26031                                   ;;mov   ax,Bios_Data    ; into the Win386 Instance Data tables
26032                                   ;mov    ax,KERNEL_SEGMENT ; 70h
26033                                   ; 21/10/2022
26034 00001098 B87000                   mov     ax,DOSBIODATASEG ; 0070h
26035 0000109B 8ED8                     mov     ds,ax
26036 0000109D C706[0208][1000]         mov     word [NextStack],nextentry ; (8C0h for MSDOS 6.21 IO.SYS)
26037 000010A3 8C06[0408]               mov     [NextStack+2],es          ; (8C2h for MSDOS 6.21 IO.SYS)
26038
26039 000010A7 2EA1[6203]               mov     ax,[cs:memlo]
26040 000010AB 2EA3[9002]               mov     [cs:stack_addr],ax ;set for stack area initialization
26041 000010AF A3[0808]                 mov     [IT_StackLoc],ax ; pass it as Instance Data, too
26042 000010B2 2EA1[6403]               mov     ax,[cs:memhi]  ;this will be used by stack_init routine.
26043 000010B6 2EA3[9202]               mov     [cs:stack_addr+2],ax
26044 000010BA A3[0A08]                 mov     [IT_StackLoc+2],ax
26045
26046                                ;  space for internal stack area = stack_count(entrysize + stack_size)
26047
26048                                     ;mov   ax,entrysize ; mov ax,8
26049                                     ; 23/10/2022
26050 000010BD B80800                   mov     ax,8
26051 000010C0 2E0306[8E02]             add     ax,[cs:stack_size]
26052 000010C5 2EF726[8C02]             mul     word [cs:stack_count]
26053
26054 000010CA A3[0C08]                 mov     [IT_StackSize],ax ; pass through to Instance Tables
26055
26056 000010CD 1F                       pop     ds              ; no more need to access Instance Table
26057
26058 000010CE E84402                   call    ParaRound       ; convert size to paragraphs
26059
26060                                   ; 11/12/2022
26061                                   ; ds = cs
26062                                   ;add    [cs:memhi],ax
26063 000010D1 0106[6403]               add     [memhi],ax
26064                                   ;or     byte [cs:setdevmarkflag],for_devmark ; 2
26065                                   ;or     byte [cs:setdevmarkflag],2
26066 000010D5 800E[6919]02             or      byte [setdevmarkflag],2
26067                                   ;or     byte [setdevmarkflag],for_devmark ; 2
26068                                                   ;to set the devmark_size for stack by round routine.
26069 000010DA E81537                   call    round           ; check for memory error before
26070                                                   ; continuing
26071 000010DD E87D03                   call    stackinit       ; initialize hardware stack.
26072                                                   ; cs=ds=sysinitseg,es=relocated stack code & data
26073                                skipstack:
26074
26075                                   ; 10/04/2024 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMBIO.COM)
26076                                   ; (PCDOS 7.1 IBMBIO.COM - SYSINIT:11F0h)
26077                                   ;;;
26078                                   ;push   cs
26079                                   ;pop    ds
26080                                   ; ds = cs
26081 000010E0 803E[6E03]01             cmp     byte [dosdata_umb],1  ; PCDOS 7 feature - DOSDATA=UMB/NOUMB configuration
26082                                                                 ; 1 = DOSDATA=UMB, 2 = (UMB) done, 0 = NOUMB
26083 000010E5 7773                     ja      short dosdata_umb_done; 2 - done
26084 000010E7 727D                     jb      short dosdata_noumb   ; 0 - DOSDATA=NOUMB
26085
26086 000010E9 803E[8B16]EA             cmp     byte [setdevmark],0EAh
26087 000010EE 7476                     je      short dosdata_noumb
26088
26089 000010F0 B80258                   mov     ax,5802h
26090 000010F3 CD21                     int     21h             ; DOS - 3+ - GET/SET MEMORY ALLOCATION STRATEGY
26091                                                           ; AL = function code: (DOS 5beta) get UMB link state
26092 000010F5 98                       cbw
26093 000010F6 89C7                     mov     di,ax                   ; al = 01h -> UMBs in DOS memory chain
26094                                                                   ; save current (previous) UMB link state
26095 000010F8 BB0100                   mov     bx,1                    ; bx = 01h -> add UMBs to DOS memory chain
26096
26097 000010FB B80358                   mov     ax,5803h
26098 000010FE CD21                     int     21h
26099 00001100 7264                     jc      short dosdata_noumb
26100
26101 00001102 B80058                   mov     ax,5800h
26102 00001105 CD21                     int     21h             ; DOS - 3+ - GET/SET MEMORY ALLOCATION STRATEGY
26103                                                           ; AL = function code: get allocation strategy
26104
26105 00001107 89C6                     mov     si,ax                   ; ax = current strategy
26106                                                                   ; save current (previous) allocation strategy
26107 00001109 BB4000                   mov     bx,40h                  ; bl = new strategy = 40h - high memory first fit
26108
26109 0000110C B80158                   mov     ax,5801h
26110 0000110F CD21                     int     21h
26111
26112 00001111 8B1E[6403]               mov     bx,[memhi]
26113 00001115 2B1E[6A03]               sub     bx,[prev_memhi]
26114
26115 00001119 B448                     mov     ah,48h
26116 0000111B CD21                     int     21h             ; DOS - 2+ - ALLOCATE MEMORY
26117                                                           ; BX = number of 16-byte paragraphs desired
26118 0000111D 89C1                     mov     cx,ax                   ; ax = segment of allocated block
26119 0000111F 89FB                     mov     bx,di                   ; restore previous UMB link state
26120
26121 00001121 B80358                   mov     ax,5803h
26122 00001124 CD21                     int     21h             ; DOS - 3+ - GET/SET MEMORY ALLOCATION STRATEGY
26123                                                           ; AL = function code: (DOS 5beta) set UMB link state
26124 00001126 89F3                     mov     bx,si                   ; restore previous allocation strategy
26125
26126 00001128 B80158                   mov     ax,5801h
26127 0000112B CD21                     int     21h             ; DOS - 3+ - GET/SET MEMORY ALLOCATION STRATEGY
26128                                                           ; AL = function code: set allocation strategy
26129 0000112D 81F900A0                 cmp     cx,0A000h               ; Is the allocated memory block (segment) a UMB?
26130 00001131 7233                     jb      short dosdata_noumb   ; no
26131
26132                                   ;mov    word [ALLOCLIM],0FFFFh
26133                                   ;mov    word [memlo],0
26134 00001133 890E[6403]               mov     [memhi],cx
26135 00001137 49                       dec     cx
26136 00001138 8EC1                     mov     es,cx                   ; point to arena/mcb
26137                                   ; 10/04/2024
26138 0000113A 31C9                     xor     cx,cx ; 0
26139 0000113C 890E[6203]               mov     [memlo],cx ; 0
26140 00001140 49                       dec     cx
26141 00001141 890E[A502]               mov     [ALLOCLIM],cx ; 0FFFFh
26142
26143 00001145 26C70601000800           mov     word [es:1],8           ; [es:arena_owner], 8 ; set impossible owner
26144 0000114C 26C70608005344           mov     word [es:8],4453h       ; [es:arena_name],'SD' ; System Data
26145 00001153 FE06[6E03]               inc     byte [dosdata_umb]      ; 1 -> 2 ; DOSDATA=UMB done.
26146 00001157 E909FD                   jmp     dosfts
26147
26148                                dosdata_umb_done:
26149 0000115A A1[6A03]                 mov     ax,[prev_memhi]               ; (recent memory block/segment before UMBs)
26150 0000115D A3[6403]                 mov     [memhi],ax
26151 00001160 A1[6C03]                 mov     ax,[prev_alloclim]
```

```
26152 00001163 A3[A502]                 mov     [ALLOCLIM],ax
26153                                   dosdata_noumb:
26154                                       ;;;
26155
26156                                   ;skipstack:
26157                                       ; 24/10/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 IO.SYS)
26158                                       ; (SYSINIT:0F99h)
26159
26160                                       ; 11/12/2022
26161                                       ; ds = cs
26162                                       ;push  cs
26163                                       ;pop   ds
26164
26165 00001166 A0[9F02]                     mov     al,[FILES]
26166 00001169 30E4                         xor     ah,ah           ; do not use cbw instruction!!!!!
26167                                                               ;  it does sign extend.
26168 0000116B 89C1                         mov     cx,ax
26169 0000116D 31DB                         xor     bx,bx           ;close standard input
26170 0000116F B43E                         mov     ah,3Eh ; CLOSE
26171 00001171 CD21                         int     21h
26172 00001173 BB0200                       mov     bx,2
26173                                   rcclloop:                   ;close everybody but standard output
26174 00001176 B43E                         mov     ah,3Eh ; CLOSE ; need output so we can print message
26175 00001178 CD21                         int     21h             ; in case we can't get new one open.
26176 0000117A 43                           inc     bx
26177 0000117B E2F9                         loop    rcclloop
26178
26179 0000117D BA[C54A]                     mov     dx,condev
26180 00001180 B002                         mov     al,2
26181 00001182 B43D                         mov     ah,3Dh ; OPEN  ;open con for read/write
26182 00001184 F9                           stc                     ; set for possible int 24
26183 00001185 CD21                         int     21h
26184 00001187 7305                         jnc     short goaux
26185 00001189 E89C38                       call    badfil
26186 0000118C EB13                         jmp     short goaux2
26187                                   goaux:
26188 0000118E 50                           push    ax
26189 0000118F BB0100                       mov     bx,1            ;close standard output
26190 00001192 B43E                         mov     ah,3Eh ; CLOSE
26191 00001194 CD21                         int     21h
26192 00001196 58                           pop     ax
26193
26194 00001197 89C3                         mov     bx,ax           ;new device handle
26195 00001199 B445                         mov     ah,45h ; XDUP
26196 0000119B CD21                         int     21h             ;dup to 1,stdout
26197 0000119D B445                         mov     ah,45h ; XDUP
26198 0000119F CD21                         int     21h             ;dup to 2,stderr
26199                                   goaux2:
26200 000011A1 BA[C94A]                     mov     dx,auxdev
26201 000011A4 B002                         mov     al,2            ;read/write access
26202 000011A6 E8B038                       call    open_dev
26203
26204 000011A9 BA[CD4A]                     mov     dx,prndev
26205 000011AC B001                         mov     al,1            ;write only
26206 000011AE E8A838                       call    open_dev
26207
26208                                   ;global rearm command for shared interrupt devices attached in the system;
26209                                   ;shared interrupt attachment has some problem when it issues interrupt
26210                                   ;during a warm reboot. once the interrupt is presented by the attachment,
26211                                   ;no further interrupts on that level will be presented until a global rearm
26212                                   ;is issued. by the request of the system architecture group, msbio will
26213                                   ;issue a global rearm after every device driver is loaded.
26214                                   ;to issue a global rearm: ;for pc1,xt,palace
26215                                   ;
26216                                   ;                       out 02f2h,xx  ; interrupt level 2
26217                                   ;                       out 02f3h,xx  ; interrupt level 3
26218                                   ;                       out 02f4h,xx  ; interrupt level 4
26219                                   ;                       out 02f5h,xx  ; interrupt level 5
26220                                   ;                       out 02f6h,xx  ; interrupt level 6
26221                                   ;                       out 02f7h,xx  ; interrupt level 7
26222                                   ;
26223                                   ;   for pc at,in addition to the above commands,
26224                                   ;   need to handle the secondary interrupt handler
26225                                   ;
26226                                   ;                       out 06f2h,xx  ; interrupt level 10
26227                                   ;                       out 06f3h,xx  ; interrupt level 11
26228                                   ;                       out 06f4h,xx  ; interrupt level 12
26229                                   ;                       out 06f6h,xx  ; interrupt level 14
26230                                   ;                       out 06f7h,xx  ; interrupt level 15
26231                                   ;
26232                                   ;   for round-up machine
26233                                   ;
26234                                   ;                       none.
26235
26236                                   ; where xx stands for any value.
26237                                   ;
26238                                   ; for your information,after naples level machine,the system service bios
26239                                   ; call (int 15h),function ah=0c0h returns the system configuration parameters
26240
26241                                       ; 24/10/2022
26242
26243 000011B1 50                           push    ax
26244 000011B2 53                           push    bx
26245 000011B3 52                           push    dx
26246 000011B4 06                           push    es
26247
26248 000011B5 B0FF                         mov     al,0FFh             ;reset h/w by writing to port
26249 000011B7 BAF202                       mov     dx,2F2h             ;get starting address
26250 000011BA EE                           out     dx,al               ; out 02f2h,0ffh
26251 000011BB 42                           inc     dx
26252 000011BC EE                           out     dx,al               ; out 02f3h,0ffh
26253 000011BD 42                           inc     dx
26254 000011BE EE                           out     dx,al               ; out 02f4h,0ffh
26255 000011BF 42                           inc     dx
26256 000011C0 EE                           out     dx,al               ; out 02f5h,0ffh
26257 000011C1 42                           inc     dx
26258 000011C2 EE                           out     dx,al               ; out 02f6h,0ffh
26259 000011C3 42                           inc     dx
26260 000011C4 EE                           out     dx,al               ; out 02f7h,0ffh
26261
26262                                   ;sb secondary global rearm
26263
26264 000011C5 B800F0                       mov     ax,0F000h           ;get machine type
26265 000011C8 8EC0                         mov     es,ax
26266 000011CA 26803EFEFFFC                 cmp     byte [es:0FFFEh],0FCh ;q:is it a at type machine
26267 000011D0 740D                         je      short startrearm    ; *if at no need to check
26268
26269 000011D2 B4C0                         mov     ah,0C0h             ;get system configuration
26270 000011D4 CD15                         int     15h                 ; *
26271 000011D6 7216                         jc      short finishrearm   ; *jmp if old rom
26272
26273                                   ; test feature byte for secondary interrupt controller
26274
26275 000011D8 26F6470540                   test    byte [es:bx+5],40h
```

```
                                    ; 24/10/2022
                                    ;test  byte [es:bx+ROMBIOS_DESC.bios_sd_featurebyte1],ScndIntController
000011DD 740F                       je     short finishrearm     ;jmp if it is there

                                startrearm:
000011DF B0FF                       mov    al,0FFh               ;write any pattern to port
000011E1 BAF206                     mov    dx,6F2h               ;get starting address
000011E4 EE                         out    dx,al                 ;out 06f2h,0ffh
000011E5 42                         inc    dx                    ;bump address
000011E6 EE                         out    dx,al                 ;out 06f3h,0ffh
000011E7 42                         inc    dx                    ;bump address
000011E8 EE                         out    dx,al                 ;out 06f4h,0ffh
000011E9 42                         inc    dx                    ;bump address
000011EA 42                         inc    dx                    ;bump address
000011EB EE                         out    dx,al                 ;out 06f6h,0ffh
000011EC 42                         inc    dx                    ;bump address
000011ED EE                         out    dx,al                 ;out 06f7h,0ffh

                                finishrearm:
000011EE 07                         pop    es
000011EF 5A                         pop    dx
000011F0 5B                         pop    bx
000011F1 58                         pop    ax

                                ;    global rearm end ********************

                                ; ---------------------------------------------------------------------------
                                ; allocate sysinit_base for install= command
                                ; ---------------------------------------------------------------------------
                                ; sysinit_base allocation.
                                ;   check if endfile has been called to handle install= command.

                                set_sysinit_base:

                                ; ---------------------------------------------------------------------------
                                ;sysinit_base will be established in the secure area of
                                ;lower memory when it handles the first install= command.
                                ;sysinit_base is the place where the actual exec function will be called and
                                ;will check sysinit module in high memory if it is damaged by the application
                                ;program.  if sysinit module has been broken,then "memory error..." message
                                ;is displayed by sysinit_base.
                                ; ---------------------------------------------------------------------------

                                    ; 24/10/2022 - Retro DOS v4.0 (MSDOS 5.0 IO.SYS, SYSINIT)
                                    ; (SYSINIT:1028h)

                                    ; 11/12/2022
                                    ; ds = cs
000011F2 50                         push   ax                    ; set devmark for mem command
000011F3 A1[6403]                   mov    ax,[memhi]
000011F6 2B06[6803]                 sub    ax,[area]
000011FA A3[6003]                   mov    [impossible_owner_size],ax ;remember the size in case.
                                    ;mov   al,devmark_inst ; 'T'
000011FD B054                       mov    al,'T'
000011FF E88904                     call   setdevmark
00001202 58                         pop    ax

00001203 8B3E[6403]                 mov    di,[memhi]
00001207 8EC7                       mov    es,di
00001209 893E[D402]                 mov    [sysinit_base_ptr+2],di ; save this entry for the next use.
0000120D 31FF                       xor    di,di
0000120F 893E[D202]                 mov    [sysinit_base_ptr],di ; es:di -> destination.
00001213 BE[2113]                   mov    si,sysinit_base              ;ds:si -> source code to be relocated.
00001216 B98100                     mov    cx,end_sysinit_base-sysinit_base ; 129
                                    ; 24/10/2022
                                    ;mov   cx,128  ; 11DCh-115Ch  ; (MSDOS 5.0 IO.SYS, SYSINIT)
00001219 010E[6203]                 add    [memlo],cx
                                    ;or    byte cs:[setdevmarkflag],for_devmark ; 2
                                    ; 11/12/2022
                                    ; ds = cs
                                    ;or    byte [cs:setdevmarkflag],2
0000121D 800E[6919]02               or     byte [setdevmarkflag],2
                                    ;or    byte [setdevmarkflag],for_devmark
00001222 E8CD35                     call   round                 ; check mem error. also,readjust memhi for the next use.
00001225 F3A4                       rep    movsb                 ; reallocate it.

00001227 C706[D602][0813]           mov    word [sysinit_ptr],sysinitptr ; returning address from
0000122D 8C0E[D802]                 mov    [sysinit_ptr+2],cs    ; sysinit_base back to sysinit.
                                    ;or    word [install_flag],has_installed ; set the flag.
                                    ;or    byte [install_flag],has_installed ; 2
                                    ; 11/12/2022
00001231 800E[CE02]02               or     byte [install_flag],2
                                    ; 24/10/2022
                                    ;or    word [install_flag],2

                                ; ---------------------------------------------------------------------------
                                ; free the rest of the memory from memhi to confbot. still from confbot to
                                ; the top of the memory will be allocated for sysinit and config.sys if
                                ; have_install_cmd.
                                ; ---------------------------------------------------------------------------

00001236 E8B935                     call   round
00001239 8B1E[6403]                 mov    bx,[memhi]
0000123D A1[6803]                   mov    ax,[area]
00001240 A3[5E03]                   mov    [old_area],ax         ; save [area]
00001243 8EC0                       mov    es,ax                 ;calc what we needed
00001245 29C3                       sub    bx,ax
                                    ; 24/10/2022
00001247 B44A                       mov    ah,4Ah ; SETBLOCK
00001249 CD21                       int    21h                   ;give the rest back
                                           ; DOS - 2+ - ADJUST MEMORY BLOCK SIZE (SETBLOCK)
                                           ; ES = segment address of block to change
                                           ; BX = new size in paragraphs
0000124B 06                         push   es
0000124C 8CC0                       mov    ax,es
0000124E 48                         dec    ax
0000124F 8EC0                       mov    es,ax                 ;point to arena
                                    ;mov   word [es:ARENA.OWNER],8       ;set impossible owner
                                    ;;mov  word [es:ARENA.NAME],4453h    ; System Data
                                    ;mov   word [es:ARENA.NAME],'SD'     ; System Data
                                    ; 24/10/2022
00001251 26C70601000800             mov    word [es:1],8         ;set impossible owner
00001258 26C70608005344             mov    word [es:8],'SD'      ; System Data
0000125F 07                         pop    es

00001260 BBFFFF                     mov    bx,0FFFFh
00001263 B448                       mov    ah,48h ; ALLOC
00001265 CD21                       int    21h
00001267 B448                       mov    ah,48h ; ALLOC
00001269 CD21                       int    21h                   ; allocate the rest of the memory
                                           ; DOS - 2+ - ALLOCATE MEMORY
                                           ; BX = number of 16-byte paragraphs desired
0000126B A3[6403]                   mov    [memhi],ax            ; start of the allocated memory
0000126E C706[6203]0000             mov    word [memlo],0        ;  to be used next.
```

```
26400
26401                                   ;;;; at this moment,memory from [memhi]:0 to top-of-the memory is
26402                                   ;;;; allocated.
26403                                   ;;;; to protect sysinit,confbot module (from confbot (or =alloclim at
26404                                   ;;;; this time) to the top-of-the memory),here we are going to
26405                                   ;;;; 1). "setblock" from memhi to confbot.
26406                                   ;;;; 2). "alloc" from confbot to the top of the memory.
26407                                   ;;;; 3). "free alloc memory" from memhi to confbot.
26408
26409                                   ;memory allocation for sysinit,confbot module.
26410
26411 00001274 8EC0                        mov     es,ax
26412                                     ; 31/12/2022 - Retro DOS v4.2 (Modified MSDOS 6.21 IO.SYS)
26413                                     ; (SYSINIT:11DFh)
26414 00001276 8B1E[A302]                mov     bx,[CONFBOT]
26415                                     ; 24/10/2022
26416                                     ;mov    bx,[top_of_cdss] ; mov bx,[confbot]
26417 0000127A 29C3                      sub     bx,ax                   ; confbot - memhi
26418 0000127C 4B                        dec     bx                      ; make a room for the memory block id.
26419 0000127D 4B                        dec     bx                      ; make sure!!!.
26420 0000127E B44A                      mov     ah,4Ah ; SETBLOCK
26421 00001280 CD21                      int     21h                     ; this will free (confbot to top of memory)
26422                                           ; DOS - 2+ - ADJUST MEMORY BLOCK SIZE (SETBLOCK)
26423                                           ; ES = segment address of block to change
26424                                           ; BX = new size in paragraphs
26425 00001282 BBFFFF                    mov     bx,0FFFFh
26426 00001285 B448                      mov     ah,48h ; ALLOC
26427 00001287 CD21                      int     21h
26428 00001289 B448                      mov     ah,48h ; ALLOC
26429 0000128B CD21                      int     21h                     ; allocate (confbot to top of memory)
26430                                           ; DOS - 2+ - ALLOCATE MEMORY
26431                                           ; BX = number of 16-byte paragraphs desired
26432 0000128D A3[6803]                  mov     [area],ax               ; save allocated memory segment.
26433                                                                     ; need this to free this area for command.com.
26434 00001290 8E06[6403]                mov     es,[memhi]
26435 00001294 B449                      mov     ah,49h                  ; free allocated memory.
26436 00001296 CD21                      int     21h                     ; free (memhi to confbot(=area))
26437                                           ; DOS - 2+ - FREE MEMORY
26438                                           ; ES = segment address of area to be freed
26439                                   endfile_ret:
26440 00001298 C3                            retn
26441
26442                                   ; End of "EndFile" DOS structure configuration.
26443
26444                                   ; ----------------------------------------------------------------------
26445                                   ; 26/03/2019 - Retro DOS v4.0
26446                                   ; 24/10/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 IO.SYS, SYSINIT)
26447                                   ; ----------------------------------------------------------------------
26448                                   ; Do_Install_Exec
26449                                   ;
26450                                   ; This procedure is used to EXEC a program being loaded via the
26451                                   ; "install=" mechanism in config.sys. It does this by setting up
26452                                   ; the parameters, and then jumping to sysinit_base, which has been
26453                                   ; setup in low memory. When complete, sysinit_base will jump back
26454                                   ; up to this procedure (if sysinit remains uncorrupted by the installed
26455                                   ; program).
26456
26457                                   ;SYSINIT:10CFh:
26458
26459                                   do_install_exec:                  ; now,handles install= command.
26460
26461 00001299 56                            push    si                  ; save si for config.sys again.
26462
26463                                   ; we are going to call load/exec function.
26464                                   ; set es:bx to the parameter block here;;;;;;;
26465                                   ; set ds:dx to the asciiz string. remember that we already has 0
26466                                   ; after the filename. so parameter starts after that. if next
26467                                   ; character is a line feed (i.e. 10),then assume that the 0
26468                                   ; we already encountered used to be a carrage return. in this
26469                                   ; case,let's set the length to 0 which will be followed by
26470                                   ; carridge return.
26471
26472                                   ; es:si -> command line in config.sys. points to the first non blank
26473                                   ;character after =.
26474
26475 0000129A 06                            push    es
26476 0000129B 1E                            push    ds
26477 0000129C 07                            pop     es
26478 0000129D 1F                            pop     ds                  ; es->sysinitseg,ds->confbot seg
26479 0000129E 89F2                          mov     dx,si               ; ds:dx->file name,0 in config.sys image.
26480
26481 000012A0 31C9                          xor     cx,cx
26482 000012A2 FC                            cld
26483 000012A3 2EC606[F102]20                mov     byte [cs:ldexec_start],' ' ; clear out the parm area
26484 000012A9 BF[F202]                      mov     di,ldexec_parm
26485                                   installfilename:                  ; skip the file name
26486 000012AC AC                            lodsb                       ; al = ds:si; si++
26487                                         ; 05/09/2023
26488 000012AD 08C0                          or      al,al
26489                                         ;cmp    al,0
26490                                         ;je     short got_installparm
26491                                         ;jmp    short installfilename
26492                                         ; 10/04/2024
26493 000012AF 75FB                          jnz     short installfilename
26494                                   got_installparm:                  ; copy the parameters to ldexec_parm
26495 000012B1 AC                            lodsb
26496 000012B2 268805                        mov     [es:di],al
26497 000012B5 3C0A                          cmp     al,lf   ; cmp al,0Ah   ; line feed?
26498 000012B7 7405                          je      short done_installparm
26499 000012B9 FEC1                          inc     cl                  ; # of char. in the parm.
26500 000012BB 47                            inc     di
26501 000012BC EBF3                          jmp     short got_installparm
26502                                   done_installparm:
26503 000012BE 2E880E[F002]                  mov     byte [cs:ldexec_line],cl ; length of the parm.
26504                                         ; 05/09/2023
26505 000012C3 08C9                          or      cl,cl
26506                                         ;cmp    cl,0                ; if no parm,then
26507 000012C5 7506                          jne     short install_seg_set ; let the parm area
26508 000012C7 2EC606[F102]0D                mov     byte [cs:ldexec_start],cr ; 0Dh
26509                                                                     ; starts with cr.
26510                                   install_seg_set:
26511                                         ; 05/09/2023 - Retro DOS v4.2 IO.SYS (Optimization)
26512 000012CD 31DB                          xor     bx,bx
26513                                         ;mov    word [cs:0],0       ; make a null environment segment
26514 000012CF 2E891F                        mov     [cs:bx],bx ; 05/09/2023
26515 000012D2 8CC8                          mov     ax,cs               ; by overlap jmp instruction of sysinitseg.
26516
26517                                   ;--------------------------------------------------M067----------------
26518                                   ;
26519                                   ;   the environment pointer is made 0. so the current environment ptr.
26520                                   ;   will be the same as pdb_environ which after dosinit is 0.
26521                                   ;
26522                                   ;   mov     cs:[instexe.exec0_environ],0 ; set the environment seg.
26523                                   ;
```

```
                                ;   instexe.exec0_environ need not be initialized to 0 above. It was
                                ;   done as a fix for bug #529. The actual bug was in NLSFUNC and
                                ;   was fixed.
                                ;
                                ; ----------------------------------------------------------------------

                                ;;ifdef MULTI_CONFIG

                                ; If there's any environment data in "config_wrkseg", pass to app

                                ; 30/12/2022 - Retro DOS v4.0 (Modified MSDOS 6.21 IO.SYS SYSINIT)
                                ; 24/10/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 IO.SYS SYSINIT)
                                ;%if 0
000012D4 89C1                       mov     cx,ax ; *
                                    ; 05/09/2023
000012D6 2E391E[6019]               cmp     [cs:config_envlen],bx ; 0
                                    ;cmp    word [cs:config_envlen],0
000012DB 7405                       je      short no_envdata2
000012DD 2E8B0E[6219]               mov     cx,[cs:config_wrkseg] ; *
                                no_envdata2:
                                ;;endif  ;MULTI_CONFIG

                                ;%endif     ; 24/10/2022

                                    ;mov    [cs:instexe.exec0_environ],cx ; set the environment seg.
                                    ; 05/09/2023 (BugFix)
                                    ; 24/10/2022
000012E2 2E890E[4203]               mov     [cs:iexec.environ],cx ; *
                                    ; 02/11/2022
                                    ;mov    [cs:iexec.environ],ax ; 05/09/2023

                                    ;mov    [cs:instexe.exec0_com_line+2],ax ; set the seg.
000012E7 2EA3[4603]                 mov     [cs:iexec.ldexec_line+2],ax
                                    ;mov    [cs:instexe.exec0_5c_fcb+2],ax
000012EB 2EA3[4A03]                 mov     [cs:iexec.ldexec_5c_fcb+2],ax
                                    ;mov    [cs:instexe.exec0_6c_fcb+2],ax
000012EF 2EA3[4E03]                 mov     [cs:iexec.ldexec_6c_fcb+2],ax
000012F3 E86000                     call    sum_up
000012F6 26A3[DA02]                 mov     [es:checksum],ax      ; save the value of the sum
000012FA 31C0                       xor     ax,ax
000012FC B44B                       mov     ah,4Bh ; EXEC         ; load/exec
000012FE BB[4203]                   mov     bx,instexe            ; es:bx -> parm block.
00001301 06                         push    es                    ; save es,ds for load/exec
00001302 1E                         push    ds                    ; these registers will be restored in sysinit_base.
00001303 2EFF2E[D202]               jmp     far [cs:sysinit_base_ptr] ; jmp to sysinit_base to execute
                                                                  ; load/exec function and check sum.

                                ;---------------------------------------

                                ;j.k. this is the returning address from sysinit_base.

                                    ; 24/10/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 IO.SYS SYSINIT)

                                sysinitptr:                       ; returning far address from sysinit_base
00001308 5E                         pop     si                    ; restore si for config.sys file.
00001309 06                         push    es
0000130A 1E                         push    ds
0000130B 07                         pop     es
0000130C 1F                         pop     ds                    ; now ds - sysinitseg, es - confbot
0000130D 7305                           jnc     short install_exit_ret

0000130F 56                         push    si                    ; error in loading the file for install=.
00001310 E81937                     call    badload               ; es:si-> path,filename,0.
00001313 5E                         pop     si

                                    ; 24/10/2022
                                    ;jmp    short sysinitptr_retn ; (MSDOS 5.0 IO.SYS, SYSINIT:1140h)
                                    ; 11/12/2022
                                    ; ds = cs

                                    ; 30/12/2022 - Retro DOS v4.2
                                    ; (MSDOS 6.21 IO.SYS, SYSINIT:1283h)

                                install_exit_ret:
00001314 C3                         retn

                                ; 30/12/2022 - Retro DOS v4.2
                                %if 0
                                install_exit_ret:
                                    ;retn          ; retn (MSDOS 6.21 IO.SYS, SYSINIT:1283h) ; 18/12/2022

                                    ; 24/10/2022 (MSDOS 5.0 IO.SYS SYSINIT)
                                ;SYSINIT:1142h:
                                    mov     ah,4Dh
                                    int     21h             ; DOS - 2+ - GET EXIT CODE OF SUBPROGRAM (WAIT)
                                    cmp     ah,3
                                    jz      short sysinitptr_retn
                                    call    error_line
                                    stc
                                sysinitptr_retn:   ; (SYSINIT:114Fh)
                                    retn

                                %endif ; 24/10/2022

                                ; ----------------------------------------------------------------------

                                ;** ParaRound - Round Up length to paragraph multiple
                                ;
                                ;   ParaRound rounds a byte count up to a multiple of 16, then divides
                                ;   by 16 yielding a "length in paragraphs" value.
                                ;
                                ;   ENTRY   (ax) = byte length
                                ;   EXIT    (ax) = rounded up length in paragraphs
                                ;   USES    ax, flags

                                ParaRound:
00001315 83C00F                     add     ax,15
00001318 D1D8                       rcr     ax,1
0000131A D1E8                       shr     ax,1
0000131C D1E8                       shr     ax,1
0000131E D1E8                       shr     ax,1
00001320 C3                         retn

                                ; ----------------------------------------------------------------------
                                ; sysinit_base module.
                                ;
                                ; This module is relocated by the routine EndFile to a location in low
                                ; memory. It is then called by SYSINIT to perform the EXEC of programs
                                ; that are being loaded by the "install=" command. After the EXEC call
                                ; completes, this module performs a checksum on the SYSINIT code (at the
                                ; top of memory) to be sure that the EXECed program did not damage it.
                                ; If it did, then this module will print an error message and stop the
                                ; system. Otherwise, it returns control to SYSINIT.
                                ;
```

```
                                ;in: after relocation,
                                ;    ax = 4b00h - load and execute the program dos function.
                                ;    ds = confbot. segment of config.sys file image
                                ;    es = sysinitseg. segment of sysinit module itself.
                                ;    ds:dx = pointer to asciiz string of the path,filename to be executed.
                                ;    es:bx = pointer to a parameter block for load.
                                ;    SI_end (byte) - offset vaule of end of sysinit module label
                                ;    bigsize (word) - # of word from confbot to SI_end.
                                ;    chksum (word) - sum of every byte from confbot to SI_end in a
                                ;                    word boundary moduler form.
                                ;    sysinit_ptr (dword ptr) - return address to sysinit module.
                                ;
                                ;note: sysinit should save necessary registers and when the control is back

                                    ; 24/10/2022
                                    ; (SYSINIT:115Ch for MSDOS 5.0 SYSINIT)
                                sysinit_base:
00001321 2E8C166200                 mov     [cs:sysinit_base_ss],ss      ; save stack
00001326 2E89266400                 mov     [cs:sysinit_base_sp],sp
0000132B CD21                       int     21h                          ; load/exec dos call.
0000132D 2E8E166200                 mov     ss,[cs:sysinit_base_ss]      ; restore stack
00001332 2E8B266400                 mov     sp,[cs:sysinit_base_sp]
00001337 1F                         pop     ds                           ; restore confbot seg
00001338 07                         pop     es                           ; restore sysinitseg
00001339 7216                       jc      short sysinit_base_end; load/exec function failed.
                                                                         ; at this time,i don't have to worry about
                                                                         ; that sysinit module has been broken or not.
0000133B E81800                     call    sum_up                       ; otherwise,check if it is good.
0000133E 263906[DA02]               cmp     [es:checksum],ax
00001343 740C                       je      short sysinit_base_end

                                ;   memory broken. show "memory allocation error" message and stall.

00001345 B409                       mov     ah,9
00001347 0E                         push    cs
00001348 1F                         pop     ds
                                    ; 30/12/2022
                                    ; (MSDOS 6.21 IO.SYS, SYSINIT:12B8h)
                                    ;mov    dx,102
00001349 BA6600                     mov     dx,mem_alloc_err_msgx-sysinit_base ; 65h (for MSDOS 5.0 SYSINIT)
                                                                               ; 66h (for MSDOS 6.21 SYSINIT)
0000134C CD21                       int     21h
                                            ; DOS - PRINT STRING
                                            ; DS:DX -> string terminated by "$"

                                    ; 30/12/2022 - Retro DOS v4.2
                                stall:
                                    ; 24/10/2022
                                _stall:
                                    ; 11/12/2022
0000134E F4                         hlt
                                    ; 02/11/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
                                    ;hlt                         ;use HLT to minimize energy consumption
0000134F EBFD                           jmp short _stall

                                sysinit_base_end:
00001351 26FF2E[D602]               jmp     far [es:sysinit_ptr]  ;return back to sysinit module

                                ;--------------------------------------

                                sum_up:

                                ;in:   es - sysinitseg.
                                ;out:  ax - result
                                ;
                                ;remark: since this routine will only check starting from "locstack" to the end of
                                ;    sysinit segment,the data area, and the current stack area are not
                                ;    coverd. in this sense,this check sum routine only gives a minimal
                                ;    gaurantee to be safe.
                                ;
                                ;first sum up confbot seg.

00001356 1E                         push    ds
                                    ; 31/12/2022 - Retro DOS v4.2 (Modified MSDOS 6.21 IO.SYS)
                                    ; (SYSINIT:12C6h)
00001357 26A1[A302]                 mov     ax,[es:CONFBOT]
                                    ; 24/10/2022
                                    ;mov    ax,[es:top_of_cdss]
0000135B 8ED8                       mov     ds,ax
0000135D 31F6                       xor     si,si
0000135F 31C0                       xor     ax,ax
00001361 268B0E[D002]               mov     cx,[es:config_size]   ; if config_size has been broken,then this
                                                                  ; whole test better fail.
00001366 D1E9                       shr     cx,1                  ; make it a word count
00001368 7406                       jz      short sum_sys_code    ; when config.sys file not exist.
                                sum1:
0000136A 0304                       add     ax,[si]
0000136C 46                         inc     si
0000136D 46                         inc     si
0000136E E2FA                       loop    sum1
                                ;now,sum up sysinit module.
                                sum_sys_code:
                                    ; 24/10/2022
00001370 BE7013                     mov     si,locstack ;; 5A6h (MSDOS 5.0 IO.SYS, SYSINIT)
                                                        ;; 532h (MSDOS 6.21 IO.SYS, SYSINIT)
                                                        ; 10/04/2024
                                                        ; 586h (PCDOS 7.1 IBMBIO.COM, SYSINIT)
                                                                 ; starting after the stack.  M069
                                                                 ;  this does not cover the possible stack code!!!
                                    ;;;mov  cx,22688  ; for MSDOS 6.21 IO.SYS
                                    ; 02/11/2022
                                    ;;mov   cx,3D20h  ; (15648) for MSDOS 5.0 IO.SYS (SYSINIT)
                                    ; 10/04/2024
                                    ;mov    cx,5B40h  ; (23360) for PCDOS 7.1 IBMBIO.COM (SYSINIT)
                                    ; 30/12/2022
00001373 B9[1054]                   mov     cx,SI_end ; (22688)   ; SI_end is the label at the end of sysinit
00001376 29F1                       sub     cx,si                 ;  from after_checksum to SI_end
00001378 D1E9                       shr     cx,1
                                sum2:
0000137A 260304                     add     ax,[es:si]
0000137D 46                         inc     si
0000137E 46                         inc     si
0000137F E2F9                       loop    sum2
00001381 1F                         pop     ds
00001382 C3                         retn

                                ; 24/10/2022 - Retro DOS 4.0 (Modified MSDOS 5.0 IO.SYS, SYSINIT)
                                ; 30/12/2022 - Retro DOS 4.2 (Modified MSDOS 6.21 IO.SYS, SYSINIT)
                                ; (SYSINIT:12F2h)
                                ; 10/04/2024 - Retro DOS 5.0 (Modified PCDOS 7.1 IBMBIO.COM, SYSINIT)
                                ; (SYSINIT:149Dh)

                                sysinit_base_ss equ $-sysinit_base  ; = 61 (MSDOS 5.0 IO.SYS, SYSINIT:115Ch)
                                ;SYSINIT:11BDh:                     ; = 62 (MSDOS 6.21 IO.SYS, SYSINIT:1290h)
```

```
                                                    ; = 62 (PCDOS 7.1 IBMBIO.COM, SYSINIT:143Bh)
                                sysinit_base_ssx:
00001383 0000                       dw      0
                                sysinit_base_sp equ $-sysinit_base  ; = 63 (MSDOS 5.0 IO.SYS, SYSINIT:1161h)
                                ;SYSINIT:11BFh:                     ; = 64 (MSDOS 6.21 IO.SYS, SYSINIT:1295h)
                                                    ; = 64 (PCDOS 7.1 IBMBIO.COM, SYSINIT:1440h)
                                sysinit_base_spx:
00001385 0000                       dw      0

                                mem_alloc_err_msgx:

                                       ;include msbio.cl4        ; memory allocation error message

                                ;(SYSINIT:12F6h: ; MSDOS 6.21 IO.SYS)
                                ;SYSINIT:14A1h:    ; PCDOS 7.1 IBMBIO.COM
00001387 0D0A                       db      0Dh,0Ah
00001389 4D656D6F727920616C-        db      'Memory allocation error $'
00001392 6C6F636174696F6E20-
0000139B 6572726F722024


                                end_sysinit_base: ; label byte
                                    ; 24/10/2022
                                    ; (SYSINIT:11DCh for MSDOS 5.0 SYSINIT)

                                ; ------------------------------------------------------------------------
                                ; Set_Buffer
                                ;
                                ;function: set buffers in the real memory.
                                ;      lastly set the memhi,memlo for the next available free address.
                                ;
                                ;input:    ds:bx -> buffinfo.
                                ;      [memhi]:[memlo = 0] = available space for the hash bucket.
                                ;      singlebuffersize = buffer header size + sector size
                                ;
                                ;output:   buffers Queue established.
                                ;      [memhi]:[memlo] = address of the next available free space.
                                ; ------------------------------------------------------------------------
                                    ; 25/10/2022 - Retro DOS 4.0 (Modified MSDOS 5.0 IO.SYS, SYSINIT)
                                    ; (SYSINIT:11DCh)

                                    ; 11/04/2024 - Retro DOS 5.0 (Modified PCDOS 7.1 IBMBIO.COM)
                                    ; (PCDOS 7.1 IBMBIO.COM - SYSINIT:14BCh)

                                set_buffer:
000013A2 30D2                       xor     dl,dl                       ; assume buffers not in HMA
000013A4 E85500                     call    GetBufferAddr
000013A7 7402                       jz      short set_buff_1
000013A9 B201                       mov     dl,1                        ; buffers in HMA
                                set_buff_1:
                                    ; 25/10/2022
                                    ;mov    [bx+BUFFINF.Buff_Queue],di  ; head of Buff Q
000013AB 893F                       mov     [bx],di
                                    ;mov    [bx+BUFFINF.Buff_Queue+2],es
000013AD 8C4702                     mov     [bx+2],es
                                    ;mov    word [bx+BUFFINF.Dirty_Buff_Count],0 ;set dirty_count to 0.
000013B0 C747040000                 mov     word [bx+4],0

000013B5 89F8                       mov     ax,di
000013B7 2E8B0E[9902]               mov     cx,[cs:buffers]
000013BC 57                         push    di                          ; remember first buffer

                                ;   for each buffer

                                nxt_buff:
000013BD E87500                     call    set_buffer_info             ; set buf_link,buf_id...
000013C0 89C7                       mov     di,ax
000013C2 E2F9                       loop    nxt_buff

000013C4 2E2B3E[9D02]               sub     di,[cs:singlebuffersize]    ; point to last buffer

000013C9 59                         pop     cx                          ; get first buffer
                                    ;mov    [es:di+buffinfo.buf_next],cx ; last->next = first
000013CA 26890D                     mov     [es:di],cx
000013CD 87CF                       xchg    cx,di
                                    ;mov    [es:di+buffinfo.buf_prev],cx ; first->prev = last
                                    ; 25/10/2022
000013CF 26894D02                   mov     [es:di+2],cx

000013D3 08D2                       or      dl,dl                       ; In HMa ?
000013D5 7417                       jz      short set_buff_2            ; no
                                    ;mov    byte [bx+BUFFINF.Buff_In_HMA],1
000013D7 C6470C01                   mov     byte [bx+12],1
000013DB 2EA1[6403]                 mov     ax,[cs:memhi]               ; seg of scratch buff
                                    ;mov    word [bx+BUFFINF.Lo_Mem_Buff],0      ; offset of sctarch buff is 0
000013DF C7470D0000                 mov     word [bx+13],0
                                    ;mov    [bx+BUFFINF.Lo_Mem_Buff+2],ax
000013E4 89470F                     mov     word [bx+15],ax
000013E7 2EA1[9D02]                 mov     ax,[cs:singlebuffersize]    ; size of scratch buff
                                    ; 11/04/2024 - Retro DOS v5.0
                                    ; 05/09/2023
                                    ;;sub   ax,bufinsiz ; 20            ; buffer head not required
                                    ;sub    ax,20
000013EB 83E818                     sub     ax,24 ; bufinsiz            ; (bufinsiz is 24 in PCDOS 7.1)

                                set_buff_2:
000013EE 2E0106[6203]               add     [cs:memlo],ax
                                    ;or     byte [cs:setdevmarkflag],for_devmark ; 2
000013F3 2E800E[6919]02             or      byte [cs:setdevmarkflag],2
                                    ;call   round
                                    ;retn
                                    ; 12/12/2022
000013F9 E9F633                     jmp     round

                                ; ------------------------------------------------------------------------
                                ; procedure : GetBufferAddr
                                ;
                                ;         Gets the buffer address either in HMA or in Lo Mem
                                ;
                                ; returns in es:di the buffer adress
                                ; returns NZ if allocated in HMA
                                ; ------------------------------------------------------------------------

                                    ; 25/10/2022
                                GetBufferAddr:
000013FC 53                         push    bx
000013FD 52                         push    dx

                                    ; 11/04/2024 - Retro DOS v5.0
                                    ; PCDOS 7.1 IBMBIO.COM
                                    ;;;
000013FE 2E803E[6E03]02             cmp     byte [cs:dosdata_umb],2
                                                    ; is dosdata moved to UMB ? (DOSDATA=UMB done)
00001404 7506                       jne     short gba_1     ; no
```

```
26894 00001406 837F02FF                 cmp     word [bx+2],0FFFFh ; is the buffer (already) in HMA ?
26895 0000140A 7423                     je      short gba_2     ; yes
26896                                   gba_1:
26897                                   ;;;
26898
26899 0000140C 2EA1[9D02]               mov     ax, [cs:singlebuffersize]
26900 00001410 2EF726[9902]             mul     word [cs:buffers]
26901                                   ;add    ax,0Fh
26902 00001415 83C00F                   add     ax,15
26903                                   ; 02/11/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
26904                                   ;and    ax,~15  ; 0FFF0h        ; para round
26905                                   ; 12/12/2022
26906 00001418 24F0                     and     al,~15  ; 0F0h
26907 0000141A 89C3                     mov     bx,ax
26908 0000141C B8024A                   mov     ax,4A02h
26909                                   ;mov    ax,((multMULT<<8)+multMULTALLOCHMA)
26910 0000141F CD2F                     int     2Fh     ; DOS 5+ - ALLOCATE HMA SPACE
26911                                                   ;     AX = 4A02h
26912                                                   ;     BX = number of bytes
26913                                                   ; Return:
26914                                                   ;     ES:DI -> start of allocated HMA block or FFFFh:FFFFh
26915                                                   ;     BX = number of bytes actually allocated
26916                                                   ;          (rounded up to next paragraph)
26917                                                   ; Notes:
26918                                                   ;     this call is not valid unless DOS is loaded in the HMA
26919                                                   ;     (DOS=HIGH)
26920
26921 00001421 83FFFF                     cmp     di,0FFFFh
26922 00001424 7506                     jne     short got_hma
26923
26924                                   ;mov    di,0            ; dont xor di,di Z flag needed
26925                                   ; 05/09/2023
26926                                   ; zf=1
26927 00001426 47                       inc     di ; 0FFFFh -> 0
26928                                   ; zf=1
26929
26930                                   ;zf=1
26931                                   ;xor    di,di   ; 25/10/2022
26932                                   ;zf=1
26933 00001427 2E8E06[6403]             mov     es,[cs:memhi]
26934                                   got_hma:
26935 0000142C 5A                       pop     dx
26936 0000142D 5B                       pop     bx
26937 0000142E C3                       retn
26938
26939                                   ; 11/04/2024 - Retro DOS v5.0
26940                                   ; PCDOS 7.1 IBMBIO.COM
26941                                   ;;;
26942                                   gba_2:
26943 0000142F C43F                     les     di,[bx]
26944 00001431 09FF                     or      di,di
26945                                   ;pop    dx
26946                                   ;pop    bx
26947                                   ;retn
26948                                   ; 11/04/2024 - Retro DOS v5.0
26949 00001433 EBF7                     jmp     short got_hma
26950                                   ;;;
26951
26952                                   ; ---------------------------------------------------------------------------
26953
26954                                   set_buffer_info:
26955
26956                                   ;function: set buf_link,buf_id,buf_sector
26957                                   ;
26958                                   ;in: es:di -> buffer header to be set.
26959                                   ;    ax = di
26960                                   ;
26961                                   ;out:
26962                                   ;    above entries set.
26963
26964                                       ; 25/10/2022
26965 00001435 2EFF36[BD02]             push    word [cs:buf_prev_off]
26966                                   ;pop    word [es:di+buffinfo.buf_prev]
26967 0000143A 268F4502                 pop     word [es:di+2]
26968 0000143E 2EA3[BD02]               mov     [cs:buf_prev_off],ax
26969 00001442 2E0306[9D02]             add     ax,[cs:singlebuffersize]      ;adjust ax
26970                                   ;mov    [es:di+buffinfo.buf_next],ax
26971 00001447 268905                   mov     [es:di],ax
26972                                   ;mov    word [es:di+buffinfo.buf_ID],00FFh  ; new buffer free
26973 0000144A 26C74504FF00             mov     word [es:di+4],00FFh
26974                                   ;mov    word [es:di+buffinfo.buf_sector],0   ; to compensate the masm 3 bug
26975 00001450 26C745060000             mov     word [es:di+6],0
26976                                   ;mov    word [es:di+buffinfo.buf_sector+2],0 ; to compensate the masm 3 bug
26977 00001456 26C745080000             mov     word [es:di+8],0
26978 0000145C C3                       retn
26979
26980                                   ; ==========================================================================
26981                                   ; MSSTACK initialization routine - MSDOS 6.0 - SYSDINIT1.ASM - 1991
26982                                   ; ---------------------------------------------------------------------------
26983                                   ; 27/03/2019 - Retro DOS v4.0
26984
26985                                   ; ---------------------------------------------------------------------------
26986                                   ; ibmstack initialization routine.
26987                                   ;
26988                                   ;   to follow the standard interrupt sharing scheme, msstack.asm
26989                                   ;   has been modified. this initialization routine also has to
26990                                   ;   be modified because for the interrupt level 7 and 15, firstflag
26991                                   ;   should be set to signal that this interrupt handler is the
26992                                   ;   first handler hooked to this interrupt vector.
26993                                   ;   we determine this by looking at the instruction pointed by
26994                                   ;   this vector. if it is iret, then this handler should be the
26995                                   ;   first one. in our case, only the interrupt vector 77h is the
26996                                   ;   interrupt level 15. (we don't hook interrupt level 7.)
26997                                   ;
26998                                   ;   the followings are mainly due to m.r.t; ptm fix of p886 12/3/86
26999                                   ;   some design changes are needed to the above interrupt sharing
27000                                   ;   method. the above sharing scheme assumes that 1). interrupt
27001                                   ;   sharing is never done on levels that have bios support. 2). "phantom"
27002                                   ;   interrupts would only be generated on levels 7 and 15.
27003                                   ;   these assumptions are not true any more. we have to use the firstflag
27004                                   ;   for every level of interrupt. we will set the firstflag on the following
27005                                   ;   conditions:
27006                                   ;
27007                                   ;    a.     if the cs portion of the vector is 0000, then "first"
27008                                   ;    b. else if cs:ip points to valid shared header, then not "first"
27009                                   ;    c. else if cs:ip points to an iret, then "first"
27010                                   ;    d. else if cs:ip points to dummy, then "first"
27011                                   ;
27012                                   ;   where dummy is - the cs portion must be f000, and the ip portion must
27013                                   ;   be equal to the value at f000:ff01. this location is the initial value
27014                                   ;   from vector_table for interrupt 7, one of the preserved addresses in all
27015                                   ;   the bioses for all of the machines.
27016                                   ;
27017                                   ;   system design group requests bios to handle the phantom interrupts.
```

```
                                ;
                                ;   the "phantom" interrupt is an illegal interrupt such as an interrupt
                                ;   produced by the bogus adapter card even without interrupt request is
                                ;   set.  more specifically, 1). the 8259 has a feature when running in
                                ;   edge triggered mode to latch a pulse and present the interrupt when
                                ;   the processor indicates interrupt acknowledge (inta). the interrupt
                                ;   pulse was exist at the time of inta to get a "phantom" interrupt.
                                ;   2). or, this is caused by adapter cards placing a glitch on the
                                ;   interrupt line.
                                ;
                                ;   to handle those "phantom" interrupts, the main stack code will check
                                ;   the own firstflag, and if it is not "first" (which means the forward
                                ;   pointer points to the legal shared interrupt handler), then pass the
                                ;   control. if it is the first, then the following action should be
                                ;   taken. we don't have to implement skack logic in this case.
                                ;
                                ;   to implement this logic, we rather choose a simple method.
                                ;   if ont of the above "firstflag" conditions is met, we are not
                                ;   going to hook this interrupt vector. the reason is if the original
                                ;   vector points to "iret" and do nothing, we don't need
                                ;   to implement the stack logic for it. this will simplify implementation
                                ;   while maintaining compatibility with the old version of dos.
                                ;   this implies that in the main stack code, there might be a stack code
                                ;   that will never be used, a dead code.
                                ;
                                ;in - cs, ds -> sysinitseg, es -> relocated stack code & data.

                                    ; 25/10/2022 - Retro DOS 4.0 (Modified MSDOS 5.0 IO.SYS, SYSINIT)
                                    ; (SYSINIT:1287h)

                                    ; 11/04/2024 - Retro DOS 5.0 (Modified PCDOS 7.1 IBMBIO.COM)
                                    ; (PCDOS 7.1 IBMBIO.COM - SYSINIT:157Ch)

                                    ; 14/12/2022
                                stackinit:
0000145D 50                         push    ax
0000145E 1E                         push    ds
0000145F 06                         push    es
00001460 53                         push    bx
00001461 51                         push    cx
00001462 52                         push    dx
00001463 57                         push    di
00001464 56                         push    si
00001465 55                         push    bp

                                ;currently es -> stack code area

                                    ; 12/12/2022
                                    ; ds = cs
00001466 A1[8C02]                   mov     ax,[stack_count]
00001469 89C1                       mov     cx,ax  ; *!*!*
                                    ; 02/11/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
                                    ; (MSDOS 5.0 IO.SYS - SYSINIT:1290h)
                                    ;mov    ax,[cs:stack_count] ; !!      ;defined in cs
0000146B 26A3[0200]                 mov     [es:stackcount],ax            ;defined in stack code area
                                    ; (MSDOS 5.0 IO.SYS - SYSINIT:1298h)
0000146F A1[8E02]                   mov     ax,[stack_size]         ; !!          ;in cs
00001472 26A3[0600]                 mov     [es:stacksize],ax
                                    ; 12/12/2022
00001476 A1[9002]                   mov     ax,[stack_addr]                       ; offset
                                    ; 02/11/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
                                    ; (MSDOS 5.0 IO.SYS - SYSINIT:129Fh)
                                    ;mov    ax,[cs:stack_addr]  ; !!
00001479 26A3[0800]                 mov     [es:stacks],ax
                                    ; 12/12/2022
0000147D 89C5                       mov     bp,ax ; *!*
0000147F A1[9202]                   mov     ax,[stack_addr+2]
                                    ; 02/11/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
                                    ; (MSDOS 5.0 IO.SYS - SYSINIT:129Fh)
                                    ;mov    ax,[cs:stack_addr+2] ; !!     ; segment
00001482 26A3[0A00]                 mov     [es:stacks+2],ax

                                ; initialize the data fields with the parameters

                                ; "firstentry" will always be at stacks

                                    ;mov    bp,[es:stacks]                ; get offset of stack
                                    ; 12/12/2022
                                    ; bp = [es:stacks] ; *!*
00001486 26892E[0C00]               mov     [es:firstentry],bp

                                ; the stacks will always immediately follow the table entries

0000148B B80800                     mov     ax,entrysize ; 8
                                    ;mov    cx,[es:stackcount]
                                    ; 12/12/2022
                                    ; cx = [es:stackcount] ; *!*!*
0000148E F7E1                       mul     cx
00001490 01E8                       add     ax,bp
00001492 26A3[0400]                 mov     [es:stackat],ax
00001496 89C3                       mov     bx,ax
00001498 83EB02                     sub     bx,2

                                ; zero the entire stack area to start with

0000149B 268B3E[0400]               mov     di,[es:stackat]
000014A0 26A1[0600]                 mov     ax,[es:stacksize]
000014A4 F7E1                       mul     cx
000014A6 89C1                       mov     cx,ax
000014A8 31C0                       xor     ax,ax
000014AA 06                         push    es
000014AB 1F                         pop     ds                            ;ds = relocated stack code seg.

                                ;now, ds -> stack code area

000014AC 8E06[0A00]                 mov     es,[stacks+2]                 ; get segment of stack area.
000014B0 FC                         cld
000014B1 F3AA                       rep     stosb

000014B3 8B0E[0200]                 mov     cx,[stackcount]

                                ; loop for "count" times, building a table entry
                                ;  cs = sysinitseg, ds = relocated stack code seg, es = segment of stack space
                                ;  cx = number of entries
                                ;  es:bp => base of stacks - 2
                                ;  es:bx => first table entry

                                buildloop:
                                    ; 11/12/2022
                                    ;mov    byte [es:bp+allocbyte],free   ; mov [es:bp+0],0
                                    ; 25/10/2022
                                    ;mov    byte [es:bp],free
                                    ; 06/07/2023
000014B7 26884600                   mov     [es:bp],al ; 0 ; free
```

```
000014BB 26884601                   mov     [es:bp+intlevel],al   ; ax = 0
                                    ;mov    [es:bp+1],al
000014BF 26894602                   mov     [es:bp+savedsp],ax
                                    ;mov    [es:bp2],ax
000014C3 26894604                   mov     [es:bp+savedss],ax
                                    ;mov    [es:bp+4],ax
000014C7 031E[0600]                 add     bx,[stacksize]
000014CB 26895E06                   mov     [es:bp+newsp],bx              ; mov [es:bp+6],bx
                                    ;mov    [es:bp+6],bx
000014CF 26892F                     mov     [es:bx],bp
000014D2 83C508                     add     bp,entrysize ; 8

000014D5 E2E0                       loop    buildloop

000014D7 83ED08                     sub     bp,entrysize ; 8
000014DA 892E[0E00]                 mov     [lastentry],bp
000014DE 892E[1000]                 mov     [nextentry],bp

000014E2 1E                         push    ds
                                    ;mov    ax,0F000h             ;look at the model byte
                                    ; 05/09/2023
000014E3 B4F0                       mov     ah,0F0h ; ax = 0F000h
000014E5 8ED8                       mov     ds,ax
000014E7 803EFEFFF9                 cmp     byte [0FFFEh],0F9h ; mdl_convert ; convertible?
000014EC 1F                         pop     ds
000014ED 7504                       jne     short skip_disablenmis

000014EF B007                       mov     al,07h                ; disable convertible nmis
000014F1 E672                       out     72h,al

                                skip_disablenmis:
000014F3 31C0                       xor     ax,ax
000014F5 8EC0                       mov     es,ax                 ;es - segid of vector table at 0
                                                                  ;ds - relocated stack code segment
000014F7 FA                         cli

                                    ;irp    aa,<02,08,09,70>
                                    ;
                                    ;mov    si,aa&h*4             ;pass where vector is to be adjusted
                                    ;mov    di,offset int19old&aa ;we have to set old&aa for int19 handler too.
                                    ;mov    bx,offset old&aa      ;pass where to save original owner pointer
                                    ;mov    dx,offset int&aa      ;pass where new handler is
                                    ;call   new_init_loop         ;adjust the vector to new handler,
                                    ;                             ;saving pointer to original owner
                                    ;endm

                                stkinit_02:
000014F8 BE0800                     mov     si,02h*4 ; 8
000014FB BF[B305]                   mov     di,INT19OLD02
000014FE BB[1200]                   mov     bx,old02
00001501 BA[1600]                   mov     dx,int02
00001504 E84801                     call    new_init_loop
                                stkinit_08:
00001507 BE2000                     mov     si,08h*4 ; 32
0000150A BF[B805]                   mov     di,INT19OLD08
0000150D BB[3800]                   mov     bx,old08
00001510 BA[3C00]                   mov     dx,int08
00001513 E83901                     call    new_init_loop
                                stkinit_09:
00001516 BE2400                     mov     si,09h*4 ; 36
00001519 BF[BD05]                   mov     di,INT19OLD09
0000151C BB[4100]                   mov     bx,old09
0000151F BA[4500]                   mov     dx,int09
00001522 E82A01                     call    new_init_loop
                                stkinit_70:
00001525 BEC001                     mov     si,70h*4 ; 448
00001528 BF[DB05]                   mov     di,INT19OLD70
0000152B BB[4E00]                   mov     bx,old70
0000152E BA[5200]                   mov     dx,int70
00001531 E81B01                     call    new_init_loop

                                    ;irp    aa,<0a,0b,0c,0d,0e,72,73,74,76,77> ;shared interrupts
                                    ;
                                    ;mov    si,aa&h*4             ;pass where vector is to be adjusted
                                    ;push   ds                    ;save relocated stack code segment
                                    ;lds    bx, es:[si]           ;ds:bx -> original interrupt handler
                                    ;push   ds
                                    ;pop    dx                    ;dx = segment value
                                    ;
                                    ;cmp    dx,0
                                    ;jz     int&aa&_first
                                    ;
                                    ;cmp    byte ptr ds:[bx],0cfh ;does vector point to an iret?
                                    ;jz     int&aa&_first
                                    ;
                                    ;cmp    word ptr ds:[bx.6],424bh ;magic offset (see int&aa, msstack.inc)
                                    ;jz     int&aa&_not_first
                                    ;
                                    ;cmp    dx,0f000h             ;rom bios segment
                                    ;jnz    int&aa&_not_first
                                    ;
                                    ;push   es
                                    ;push   dx
                                    ;mov    dx,0f000h
                                    ;mov    es,dx
                                    ;cmp    bx,word ptr es:0ff01h
                                            ;pop    dx
                                    ;pop    es
                                    ;jz     int&aa&_first
                                    ;
                                ;int&aa&_not_first:                   ;not the first. we are going to hook vector.
                                    ;pop    ds
                                    ;mov    di, offset int19old&aa;we have to set old&aa for int19 handler too.
                                    ;mov    bx, offset old&aa     ;pass where to save original owner pointer
                                    ;mov    dx, offset int&aa     ;pass where new handler is
                                    ;call   new_init_loop         ;adjust the vector to new handler, saving
                                    ;                             ;pointer to original owner.
                                    ;jmp    short int&aa&_end
                                ;int&aa&_first:                       ;the first. don't have to hook stack code.
                                    ;pop    ds
                                ;int&aa&_end:
                                    ;
                                    ;endm

                                stkinit_0A:
00001534 BE2800                     mov     si,0Ah*4 ; 40

                                ; 14/12/2022
                                %if 0
                                    ; 25/10/2022 (MSDOS 5.0 IO.SYS compatibility)
                                    push    ds

                                    lds     bx,[es:si]
                                    push    ds
```

```
                                    pop     dx
                                
                                    cmp     dx,0
                                    je      short int_0A_first
                                
                                    cmp     byte [bx],0CFh
                                    je      short int_0A_first
                                
                                    cmp     word [bx+6],424Bh
                                    je      short int_0A_not_first
                                
                                    cmp     dx,0F000h
                                    jne     short int_0A_not_first
                                
                                    push    es
                                    push    dx
                                    mov     dx,0F000h
                                    mov     es,dx
                                    cmp     bx,[es:0FF01h]
                                            pop     dx
                                    pop     es
                                    je      short int_0A_first
                                %Endif
                                
                                    ; 14/12/2022
                                    ; 25/10/2022
00001537 E8EB00                     call    int_xx_first_check ; 27/03/2019 - Retro DOS v4.0
0000153A 730C                       jnc     short int_0A_first
                                
                                int_0A_not_first:
                                    ; 14/12/2022
                                    ; 25/10/2022
                                    ;pop    ds
0000153C BF[C205]                   mov     di,INT19OLD0A
0000153F BB[5900]                   mov     bx,old0A
00001542 BA[5700]                   mov     dx,int0A
00001545 E80701                     call    new_init_loop
                                
                                    ; 14/12/2022
                                    ;jmp    short int_0A_end
                                ;int_0A_first:
                                    ; 25/10/2022
                                    ;pop    ds
                                
                                    ; 14/12/2022
                                int_0A_first:
                                int_0A_end:
                                
                                stkinit_0B:
00001548 BE2C00                     mov     si,0Bh*4 ; 44
                                
                                    ; 14/12/2022
                                    ; 25/10/2022
0000154B E8D700                     call    int_xx_first_check ; 27/03/2019 - Retro DOS v4.0
0000154E 730C                       jnc     short int_0B_end ; int_0B_first
                                
                                ; 14/12/2022
                                %if 0
                                    ; 25/10/2022 (MSDOS 5.0 IO.SYS compatibility)
                                    push    ds
                                    lds     bx,[es:si]
                                    push    ds
                                    pop     dx
                                
                                    cmp     dx,0
                                    je      short int_0B_first
                                
                                    cmp     byte [bx],0CFh
                                    je      short int_0B_first
                                
                                    cmp     word [bx+6],424Bh
                                    je      short int_0B_not_first
                                
                                    cmp     dx,0F000h
                                    jne     short int_0B_not_first
                                
                                    push    es
                                    push    dx
                                    mov     dx,0F000h
                                    mov     es,dx
                                    cmp     bx,[es:0FF01h]
                                    pop     dx
                                    pop     es
                                    je      short int_0B_first
                                %endif
                                
                                int_0B_not_first:
                                    ; 14/12/2022
                                    ; 25/10/2022
                                    ;pop    ds
00001550 BF[C705]                   mov     di,INT19OLD0B
00001553 BB[7100]                   mov     bx,old0B
00001556 BA[6F00]                   mov     dx,int0B
00001559 E8F300                     call    new_init_loop
                                
                                    ; 14/12/2022
                                    ;jmp    short int_0B_end
                                ;int_0B_first:
                                    ; 25/10/2022
                                    ;pop    ds
                                
                                int_0B_end:
                                
                                stkinit_0C:
0000155C BE3000                     mov     si,0Ch*4 ; 48
                                
                                    ; 14/12/2022
                                    ; 25/10/2022
0000155F E8C300                     call    int_xx_first_check
00001562 730C                       jnc     short int_0C_end ; int_0C_first
                                
                                ; 14/12/2022
                                %if 0
                                    ; 25/10/2022 (MSDOS 5.0 IO.SYS compatibility)
                                    push    ds
                                    lds     bx,[es:si]
                                    push    ds
                                    pop     dx
                                
                                    cmp     dx,0
                                    je      short int_0C_first
                                
                                    cmp     byte [bx],0CFh
                                    je      short int_0C_first
```

```
27390
27391                                       cmp     word [bx+6],424Bh
27392                                       je      short int_0C_not_first
27393
27394                                       cmp     dx,0F000h
27395                                       jne     short int_0C_not_first
27396
27397                                       push    es
27398                                       push    dx
27399                                       mov     dx,0F000h
27400                                       mov     es,dx
27401                                       cmp     bx,[es:0FF01h]
27402                                       pop     dx
27403                                       pop     es
27404                                       je      short int_0C_first
27405                                   %endif
27406
27407                                   int_0C_not_first:
27408                                       ; 14/12/2022
27409                                       ; 25/10/2022
27410                                       ;pop    ds
27411 00001564 BF[CC05]                   mov     di,INT19OLD0C
27412 00001567 BB[8900]                   mov     bx,old0C
27413 0000156A BA[8700]                   mov     dx,int0C
27414 0000156D E8DF00                     call    new_init_loop
27415
27416                                       ; 14/12/2022
27417                                       ;jmp    short int_0C_end
27418                                   ;int_0C_first:
27419                                       ; 25/10/2022
27420                                       ;pop    ds
27421
27422                                   int_0C_end:
27423
27424                                   stkinit_0D:
27425 00001570 BE3400                     mov     si,0Dh*4 ; 52
27426
27427                                       ; 14/12/2022
27428                                       ; 25/10/2022
27429 00001573 E8AF00                     call    int_xx_first_check
27430 00001576 730C                       jnc     short int_0D_end ; int_0D_first
27431
27432                                   ; 14/12/2022
27433                                   %if 0
27434                                       ; 25/10/2022 (MSDOS 5.0 IO.SYS compatibility)
27435                                       push    ds
27436                                       lds     bx,[es:si]
27437                                       push    ds
27438                                       pop     dx
27439
27440                                       cmp     dx,0
27441                                       je      short int_0D_first
27442
27443                                       cmp     byte [bx],0CFh
27444                                       je      short int_0D_first
27445
27446                                       cmp     word [bx+6],424Bh
27447                                       je      short int_0D_not_first
27448
27449                                       cmp     dx,0F000h
27450                                       jne     short int_0D_not_first
27451
27452                                       push    es
27453                                       push    dx
27454                                       mov     dx,0F000h
27455                                       mov     es,dx
27456                                       cmp     bx,[es:0FF01h]
27457                                       pop     dx
27458                                       pop     es
27459                                       je      short int_0D_first
27460                                   %endif
27461
27462                                   int_0D_not_first:
27463                                       ; 14/12/2022
27464                                       ; 25/10/2022
27465                                       ;pop    ds
27466 00001578 BF[D105]                   mov     di,INT19OLD0D
27467 0000157B BB[A100]                   mov     bx,old0D
27468 0000157E BA[9F00]                   mov     dx,int0D
27469 00001581 E8CB00                     call    new_init_loop
27470
27471                                       ; 14/12/2022
27472                                       ;jmp    short int_0D_end
27473                                       ; 02/11/2022
27474                                   ;int_0D_first:
27475                                       ;pop    ds
27476
27477                                   int_0D_end:
27478
27479                                   stkinit_0E:
27480 00001584 BE3800                     mov     si,0Eh*4 ; 56
27481
27482                                       ; 14/12/2022
27483                                       ; 25/10/2022
27484 00001587 E89B00                     call    int_xx_first_check
27485 0000158A 730C                       jnc     short int_0E_end ; int_0E_first
27486
27487                                   ; 14/12/2022
27488                                   %if 0
27489                                       ; 25/10/2022 (MSDOS 5.0 IO.SYS compatibility)
27490                                       push    ds
27491                                       lds     bx,[es:si]
27492                                       push    ds
27493                                       pop     dx
27494
27495                                       cmp     dx,0
27496                                       je      short int_0E_first
27497
27498                                       cmp     byte [bx],0CFh
27499                                       je      short int_0E_first
27500
27501                                       cmp     word [bx+6],424Bh
27502                                       je      short int_0E_not_first
27503
27504                                       cmp     dx,0F000h
27505                                       jne     short int_0E_not_first
27506
27507                                       push    es
27508                                       push    dx
27509                                       mov     dx,0F000h
27510                                       mov     es,dx
27511                                       cmp     bx,[es:0FF01h]
27512                                       pop     dx
27513                                       pop     es
```

```
                                    je      short int_0E_first
                                %endif

                                int_0E_not_first:
                                    ; 14/12/2022
                                    ; 25/10/2022
                                    ;pop    ds
0000158C BF[D605]                   mov     di,INT19OLD0E
0000158F BB[B900]                   mov     bx,old0E
00001592 BA[B700]                   mov     dx,int0E
00001595 E8B700                     call    new_init_loop

                                    ; 14/12/2022
                                    ;jmp    short int_0E_end
                                ;int_0E_first:
                                    ; 25/10/2022
                                    ;pop    ds

                                int_0E_end:

                                stkinit_72:
00001598 BEC801                     mov     si,72h*4 ; 456

                                    ; 14/12/2022
                                    ; 25/10/2022
0000159B E88700                     call    int_xx_first_check
0000159E 730C                       jnc     short int_72_end ; int_72_first

                                ; 14/12/2022
                                %if 0
                                    ; 25/10/2022 (MSDOS 5.0 IO.SYS compatibility)
                                    push    ds
                                    lds     bx,[es:si]
                                    push    ds
                                    pop     dx

                                    cmp     dx,0
                                    je      short int_72_first

                                    cmp     byte [bx],0CFh
                                    je      short int_72_first

                                    cmp     word [bx+6],424Bh
                                    je      short int_72_not_first

                                    cmp     dx,0F000h
                                    jne     short int_72_not_first

                                    push    es
                                    push    dx
                                    mov     dx,0F000h
                                    mov     es,dx
                                    cmp     bx,[es:0FF01h]
                                    pop     dx
                                    pop     es
                                    je      short int_72_first
                                %endif

                                int_72_not_first:
                                    ; 14/12/2022
                                    ; 25/10/2022
                                    ;pop    ds
000015A0 BF[E005]                   mov     di,INT19OLD72
000015A3 BB[D100]                   mov     bx,old72
000015A6 BA[CF00]                   mov     dx,int72
000015A9 E8A300                     call    new_init_loop

                                    ; 14/12/2022
                                    ;jmp    short int_72_end
                                ;int_72_first:
                                    ; 25/10/2022
                                    ;pop    ds

                                int_72_end:

                                stkinit_73:
000015AC BECC01                     mov     si,73h*4 ; 460

                                    ; 14/12/2022
                                    ; 25/10/2022
000015AF E87300                     call    int_xx_first_check
000015B2 730C                       jnc     short int_73_end ; int_73_first

                                ; 14/12/2022
                                %if 0
                                    ; 25/10/2022 (MSDOS 5.0 IO.SYS compatibility)
                                    push    ds
                                    lds     bx,[es:si]
                                    push    ds
                                    pop     dx

                                    cmp     dx,0
                                    je      short int_73_first

                                    cmp     byte [bx],0CFh
                                    je      short int_73_first

                                    cmp     word [bx+6],424Bh
                                    je      short int_73_not_first

                                    cmp     dx,0F000h
                                    jne     short int_73_not_first

                                    push    es
                                    push    dx
                                    mov     dx,0F000h
                                    mov     es,dx
                                    cmp     bx,[es:0FF01h]
                                    pop     dx
                                    pop     es
                                    je      short int_73_first
                                %endif

                                int_73_not_first:
                                    ; 14/12/2022
                                    ; 25/10/2022
                                    ;pop    ds
000015B4 BF[E505]                   mov     di,INT19OLD73
000015B7 BB[E900]                   mov     bx,old73
000015BA BA[E700]                   mov     dx,int73
000015BD E88F00                     call    new_init_loop

                                    ; 14/12/2022
                                    ;jmp    short int_73_end
```

```
                                    ;int_73_first:
                                        ; 25/10/2022
                                        ;pop    ds

                                    int_73_end:

                                    stkinit_74:
000015C0 BED001                         mov     si,74h*4 ; 464

                                        ; 14/12/2022
                                        ; 25/10/2022
000015C3 E85F00                         call    int_xx_first_check
000015C6 730C                           jnc     short int_74_end ; int_74_first

                                    ; 14/12/2022
                                    %if 0
                                        ; 25/10/2022 (MSDOS 5.0 IO.SYS compatibility)
                                        push    ds
                                        lds     bx,[es:si]
                                        push    ds
                                        pop     dx

                                        cmp     dx,0
                                        je      short int_74_first

                                        cmp     byte [bx],0CFh
                                        je      short int_74_first

                                        cmp     word [bx+6],424Bh
                                        je      short int_74_not_first

                                        cmp     dx,0F000h
                                        jne     short int_74_not_first

                                        push    es
                                        push    dx
                                        mov     dx,0F000h
                                        mov     es,dx
                                        cmp     bx,[es:0FF01h]
                                        pop     dx
                                        pop     es
                                        je      short int_74_first
                                    %endif

                                    int_74_not_first:
                                        ; 14/12/2022
                                        ; 25/10/2022
                                        ;pop    ds
000015C8 BF[EA05]                       mov     di,INT19OLD74
000015CB BB[0101]                       mov     bx,old74
000015CE BA[FF00]                       mov     dx,int74
000015D1 E87B00                         call    new_init_loop

                                        ; 14/12/2022
                                        ;jmp    short int_74_end
                                    ;int_74_first:
                                        ; 25/10/2022
                                        ;pop    ds

                                    int_74_end:

                                    stkinit_76:
000015D4 BED801                         mov     si,76h*4 ; 472

                                        ; 14/12/2022
                                        ; 25/10/2022
000015D7 E84B00                         call    int_xx_first_check
000015DA 730E                           jnc     short int_76_end ; int_76_first

                                    ; 14/12/2022
                                    %if 0
                                        ; 25/10/2022 (MSDOS 5.0 IO.SYS compatibility)
                                        push    ds
                                        lds     bx,[es:si]
                                        push    ds
                                        pop     dx

                                        cmp     dx,0
                                        je      short int_76_first

                                        cmp     byte [bx],0CFh
                                        je      short int_76_first

                                        cmp     word [bx+6],424Bh
                                        je      short int_76_not_first

                                        cmp     dx,0F000h
                                        jne     short int_76_not_first

                                        push    es
                                        push    dx
                                        mov     dx,0F000h
                                        mov     es,dx
                                        cmp     bx,[es:0FF01h]
                                        pop     dx
                                        pop     es
                                        je      short int_76_first
                                    %endif

                                    int_76_not_first:
                                        ; 14/12/2022
                                        ; 25/10/2022
                                        ;pop    ds
000015DC BF[EF05]                       mov     di,INT19OLD76
000015DF BB[1901]                       mov     bx,old76
000015E2 BA[1701]                       mov     dx,int76
000015E5 E86700                         call    new_init_loop

                                        ; 14/12/2022
000015E8 EB00                           jmp     short int_76_end
                                    ;int_76_first:
                                        ; 25/10/2022
                                        ;pop    ds

                                    int_76_end:

                                    stkinit_77:
000015EA BEDC01                         mov     si,77h*4 ; 476

                                        ; 14/12/2022
                                        ; 25/10/2022
000015ED E83500                         call    int_xx_first_check
000015F0 730C                           jnc     short int_77_end ; int_77_first

```

```
                                    ; 14/12/2022
                                    %if 0
                                        ; 25/10/2022 (MSDOS 5.0 IO.SYS compatibility)
                                        push    ds
                                        lds     bx,[es:si]
                                        push    ds
                                        pop     dx

                                        cmp     dx,0
                                        je      short int_77_first

                                        cmp     byte [bx],0CFh
                                        je      short int_77_first

                                        cmp     word [bx+6],424Bh
                                        je      short int_77_not_first

                                        cmp     dx,0F000h
                                        jne     short int_77_not_first

                                        push    es
                                        push    dx
                                        mov     dx,0F000h
                                        mov     es,dx
                                        cmp     bx,[es:0FF01h]
                                        pop     dx
                                        pop     es
                                        je      short int_77_first
                                    %endif

                                    int_77_not_first:
                                        ; 14/12/2022
                                        ; 25/10/2022
                                        ;pop    ds
000015F2 BF[F405]                       mov     di,INT19OLD77
000015F5 BB[3101]                       mov     bx,old77
000015F8 BA[2F01]                       mov     dx,int77
000015FB E85100                         call    new_init_loop

                                        ; 14/12/2022
                                        ;jmp    short int_77_end
                                    ;int_77_first:
                                        ; 25/10/2022
                                        ;pop    ds

                                    int_77_end:
000015FE 1E                             push    ds
000015FF B800F0                         mov     ax,0F000h               ; look at the model byte
00001602 8ED8                           mov     ds,ax
00001604 803EFEFFF9                     cmp     byte [0FFFEh],0F9h ; mdl_convert ; pc convertible?
00001609 1F                             pop     ds
0000160A 7504                           jne     short skip_enablenmis

0000160C B027                           mov     al,27h                  ; enable convertible nmis
0000160E E672                           out     72h,al

                                    ; 25/10/2022
                                    ; (MSDOS 5.0 SYSINIT:15FBh)

                                    skip_enablenmis:
00001610 FB                             sti
                                        ;;mov   ax,Bios_Data ; 70h
                                        ;mov    ax,KERNEL_SEGMENT ; 70h
                                        ; 21/10/2022
00001611 B87000                         mov     ax,DOSBIODATASEG ; 0070h
00001614 8ED8                           mov     ds,ax

                                        ;mov    [640h],1 ; SYSINIT:1736h for MSDOS 6.21 IO.SYS

00001616 C606[B105]01                   mov     byte [INT19SEM],1       ; indicate that int 19
                                                                        ; initialization is complete

0000161B 5D                             pop     bp                      ; restore all
0000161C 5E                             pop     si
0000161D 5F                             pop     di
0000161E 5A                             pop     dx
0000161F 59                             pop     cx
00001620 5B                             pop     bx
00001621 07                             pop     es
00001622 1F                             pop     ds
00001623 58                             pop     ax
00001624 C3                             retn

                                    ; 14/12/2022
                                    ; ----------------------------------------------------------------------

                                        ; 14/12/2022
                                        ; 25/10/2022
                                    ;%if 0
                                        ; 27/03/2019 - Retro DOS v4.0
                                    int_xx_first_check:
00001625 1E                             push    ds
00001626 26C51C                         lds     bx,[es:si]
00001629 1E                             push    ds
0000162A 5A                             pop     dx

                                        ;cmp    dx,0
                                        ;je     short int_xx_first
                                        ; 05/09/2023
0000162B 21D2                           and     dx,dx
0000162D 741E                           jz      short int_xx_first

0000162F 803FCF                         cmp     byte [bx],0CFh
00001632 7419                           je      short int_xx_first

00001634 817F064B42                     cmp     word [bx+6],424Bh
00001639 7411                           je      short int_xx_not_first

0000163B 81FA00F0                       cmp     dx,0F000h
0000163F 750B                           jne     short int_xx_not_first

00001641 06                             push    es
                                        ;push   dx
                                        ;mov    dx,0F000h
00001642 8EC2                           mov     es,dx
00001644 263B1E01FF                     cmp     bx,[es:0FF01h]
                                                ;pop    dx
00001649 07                             pop     es
0000164A 7401                           je      short int_xx_first

                                    int_xx_not_first:
0000164C F9                             stc
                                    int_xx_first:
0000164D 1F                             pop     ds
```

```
0000164E C3                             retn
                                
                                ;%endif
                                
                                ; ----------------------------------------------------------------------
                                ; 27/03/2019 - Retro DOS v4.0
                                
                                ; 25/10/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 IO.SYS, SYSINIT)
                                ; (SYSINIT:1610h)
                                
                                ; 13/04/2024 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMBIO.COM, SYSINIT)
                                ; (PCDOS 7.1 IBMBIO.COM - SYSINIT:1905h)
                                
                                new_init_loop:
                                    ;;; 13/04/2024 (PCDOS 7.1 IBMBIO.COM)
0000164F 2E803E[6E03]02             cmp     byte [cs:dosdata_umb],2
                                                    ; is DOSDATA=UMB done ? (DOSDATA is in UMB)
00001655 7510                       jne     short new_init_loop_1st
00001657 1E                         push    ds              ; restore original/previous interrupt handler
                                                            ; (from int19old?? field in BIOSDATA)
                                    ;mov    ax,70h
00001658 B87000                     mov     ax,DOSBIODATASEG
0000165B 8ED8                       mov     ds,ax
0000165D C505                       lds     ax,[di]         ; restore original Int ?? handler addr from int19old?? field
0000165F 268904                     mov     [es:si],ax      ; copy the original int handler addr to its int vector addr
00001662 268C5C02                   mov     [es:si+2],ds
00001666 1F                         pop     ds
                                new_init_loop_1st:
                                    ;;;
                                
                                ;input: si=ofset into vector table of the particular int vector being adjusted
                                ;   bx=ds:offset of oldxx, where will be saved the pointer to original owner
                                ;   dx=ds:offset of intxx, the new interrupt handler
                                ;   di=offset value of int19old&aa variable in bios.
                                ;   es=zero, segid of vector table
                                ;   ds=relocated stack code segment
                                
                                ; 13/04/2024
                                %if 0
                                    mov     ax,[es:si]          ;remember offset in vector
                                    mov     [bx],ax             ; to original owner in ds
                                    mov     ax,[es:si+2]        ;remember segid in vector
                                    mov     [bx+2],ax           ; to original owner in ds
                                
                                    push    ds
                                    ;;mov   ax,Bios_Data ; 70h
                                    ;mov    ax,KERNEL_SEGMENT ; 70h
                                    ; 21/10/2022
                                    mov     ax,DOSBIODATASEG ; 0070h
                                    mov     ds,ax               ;set int19oldxx value in bios for
                                    mov     ax,[es:si]          ;int 19 handler
                                    mov     [di],ax
                                    mov     ax,[es:si+2]
                                    mov     [di+2],ax
                                    pop     ds
                                %else
                                    ; 13/04/2024 - Retro DOS v5.0
00001667 1E                         push    ds
00001668 268B4402                   mov     ax,[es:si+2]        ;remember segid in vector
0000166C 894702                     mov     [bx+2],ax           ; to original owner in ds
0000166F 50                         push    ax
00001670 268B04                     mov     ax,[es:si]          ;remember offset in vector
00001673 8907                       mov     [bx],ax             ; to original owner in ds
00001675 50                         push    ax
00001676 B87000                     mov     ax,DOSBIODATASEG ; 0070h
00001679 8ED8                       mov     ds,ax               ;set int19oldxx value in bios for
0000167B 58                         pop     ax                  ;int 19 handler
0000167C 8905                       mov     [di],ax
0000167E 58                         pop     ax
0000167F 894502                     mov     [di+2],ax
00001682 1F                         pop     ds
                                %endif
00001683 268914                     mov     [es:si],dx      ;set vector to point to new int handler
00001686 268C5C02                   mov     [es:si+2],ds
0000168A C3                         retn
                                
                                ; End of STACK initialization routine
                                ; ----------------------------------------------------------------------
                                
                                ; ----------------------------------------------------------------------
                                ;set the devmark for mem command.
                                ;in: [memhi] - the address to place devmark
                                ;    [memlo] = 0
                                ;    al = id for devmark_id
                                ;out: devmark established.
                                ;     the address saved in cs:[devmark_addr]
                                ;     [memhi] increase by 1.
                                ; ----------------------------------------------------------------------
                                
                                ; 25/10/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 IO.SYS, SYSINIT)
                                ; (SYSINIT:1637h)
                                ; 04/09/2023 - Retro DOS v4.2 (Modified MSDOS 6.21 IO.SYS, SYSINIT)
                                ; (SYSINIT:176Ch)
                                
                                ; 04/09/2023 - PCDOS 7.1 - IBMBIO.COM (SYSINIT:1944h)
                                
                                setdevmark:
                                
                                    ; 04/09/2023
                                    ;push   es
                                    ;push   cx
                                
0000168B 2E8B0E[6403]               mov     cx,[cs:memhi]
00001690 2E890E[6719]               mov     [cs:devmark_addr],cx
00001695 8EC1                       mov     es,cx
                                    ; 25/10/2022
                                    ;mov    [es:devmark.id],al
00001697 26A20000                   mov     [es:0],al
0000169B 41                         inc     cx
                                    ;mov    [es:devmark.seg],cx
0000169C 26890E0100                 mov     [es:1],cx
                                
                                    ; 04/09/2023
                                    ;pop    cx
                                    ;pop    es
                                
000016A1 2EFF06[6403]               inc     word [cs:memhi]
000016A6 C3                         retn
                                
                                ; ----------------------------------------------------------------------
                                ; SYSPRE.ASM - MSDOS 6.0 - 1992
                                ; ----------------------------------------------------------------------
                                ;; pre-load and final placement of dblspace.bin
                                ;
```

```
                                ; 08/04/2024 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMBIO.COM)
                                ; ============================================================

                                ; (PCDOS 7.1 IBMBIO.COM - SYSINIT:1964h)
                                ;;; ---------------------------------------------------------------------------
000016A7 [AB16]                 MagicDDNamePtr:    dw MagicDDName ; "\DBLSPACE.BIN"
000016A9 433A                               db 'C:'
000016AB 5C44424C5350414345-    MagicDDName:       db '\DBLSPACE.BIN',0
000016B4 2E42494E00
000016B9 433A5C535441434B45-    StackerName:       db 'C:\STACKER.BIN',0
000016C2 522E42494E00
                                tiny_stub_start:
000016C8 FFFF                               dw 0FFFFh      ; phony device driver link
000016CA FFFF                               dw 0FFFFh      ; dw -1, -1
000016CC 0080                               dw 8000h       ; mark as character device for MEM display
000016CE 00000000                           dw 2 dup(0)    ; strategy and interrupt
000016D2 44424C5342494E24                   db 'DBLSBIN$'  ; magic default load
                                tiny_stub_end:                     ; (tiny_stub_end-tiny_stub_start = 18)

                                ; ============== S U B R O U T I N E =======================================

                                ; 08/04/2024 - Retro DOS v5.0
                                ; (PCDOS 7.1 IBMBIO.COM - SYSINIT:1997h)

                                ;***MagicPreload - pre-load dblspace.bin
                                ;
                                ;   EXIT    ax = error code, 00 means none.
                                ;           ZF = true if ax == 0

                                MagicPreload:
                                    ; 13/04/2024 - Retro DOS v5.0
                                    ; ds = cs
                                    ;mov    byte [cs:setdevmarkflag],0 ; not for devmark
000016DA C606[6919]00               mov     byte [setdevmarkflag],0
000016DF E81031                     call    round
000016E2 0E                         push    cs
000016E3 07                         pop     es
                                    ;mov    byte [cs:DeviceHi],0  ; not to be loaded in UMB
                                    ; 13/04/2024
000016E4 C606[5124]00               mov     byte [DeviceHi],0
000016E9 E8791E                     call    InitDevLoad            ; set up sub-arena, DevLoadAddr,
                                                                   ;  DevLoadEnd, and DevEntry
                                                                   ;  gets arena name from bpb_addr
                                    ; 13/04/2024
                                    ; ds = cs

                                ;   check to make sure device driver fits our available space.

                                    ;mov    ax,[cs:DevLoadAddr]
000016EC A1[3524]                   mov     ax,[DevLoadAddr]
                                    ;add    ax,[cs:DevSize]              ; calculate seg after DD load
000016EF 0306[3324]                 add     ax,[DevSize]
000016F3 725E                       jc      short pre_exit_err     ; choke if overflows address space
                                    ;cmp    ax,[cs:DevLoadEnd]     ; does it overflow available space?
000016F5 3B06[3724]                 cmp     ax,[DevLoadEnd]
000016F9 7758                       ja      short pre_exit_err

                                _LoadDev:                          ; we're golden if not
                                    ; 13/04/2024
                                    ; ds = cs
                                    ;push   cs
                                    ;pop    ds
                                    ;mov    dx,[cs:MagicDDNamePtr]
000016FB 8B16[A716]                 mov     dx,[MagicDDNamePtr]
000016FF E8A41F                     call    ExecDev                ; load device driver using exec call
00001702 724F                       jb      short pre_exit_err

                                    ; 13/04/2024
                                    ; ds = cs
                                    ;les    bx,[cs:DevEntry]       ; point to the Magic DD header
00001704 C41E[3924]                 les     bx,[DevEntry]
00001708 26817F122C2E               cmp     word [es:bx+12h],2E2Ch; is it our stamp? ; ',.'
0000170E 7543                       jnz     short pre_exit_err
                                    ;mov    word [cs:MagicBackdoor],14h ; save the backdoor entry.
                                    ;                              ; (initial IP -EXE header offset 20-)
                                    ;mov    [cs:MagicBackdoor+2],es
00001710 C706[9003]1400             mov     word [MagicBackdoor],14h
00001716 8C06[9203]                 mov     [MagicBackdoor+2],es

0000171A 0E                         push    cs
0000171B 07                         pop     es
0000171C BB[6F03]                   mov     bx,packet

                                    ;mov    word [cs:break_addr],0
                                    ;mov    ax,[cs:DevLoadEnd]
                                    ;mov    [cs:break_addr+2],ax
                                    ;mov    al,[cs:drivenumber]    ; pass drive number to DBLSPACE as if
                                    ;mov    [cs:devdrivenum],al    ; it is a normal block device driver
0000171F C706[7D03]0000             mov     word [break_addr],0
00001725 A1[3724]                   mov     ax,[DevLoadEnd]
00001728 A3[7F03]                   mov     [break_addr+2],ax
0000172B A0[8503]                   mov     al,[drivenumber]       ; pass drive number to DBLSPACE as if
0000172E A2[8503]                   mov     [devdrivenum],al       ; it is a normal block device driver

00001731 B80A00                     mov     ax,10                  ; DS_INTERNAL_REVISION
                                                                   ; tell it what revision we expect
                                    ;call   far [cs:MagicBackdoor]; first time call is init entry point
00001734 FF1E[9003]                 call    far [MagicBackdoor]
                                                                   ; with a standard device driver
                                                                   ; init packet at es:bx
00001738 731D                       jnb     short no_driver_version_fail ; skip if not a version failure
0000173A B80600                     mov     ax,6                   ; DS_INTERNAL_REVISION_6 ; (Stacker ?)
                                                                   ; tell it what revision we expect
                                    ;call   far [cs:MagicBackdoor]
0000173D FF1E[9003]                 call    far [MagicBackdoor]
00001741 7314                       jnb     short no_driver_version_fail

                                ;   In this case, we're going to display a message

                                    ;push   cs
                                    ;pop    ds
                                    ; 13/04/2024
                                    ; ds = cs
00001743 BA[DA53]                   mov     dx,baddblspace         ; "Required system component is not instal"...
00001746 E80B33                     call    print                  ; display the message

                                ;   point backdoor call back to safe far return

                                fail_driver_load:
                                    ;mov    [cs:MagicBackdoor+2],cs
                                    ;mov    word [cs:MagicBackdoor],NullBackdoor
00001749 8C0E[9203]                 mov     [MagicBackdoor+2],cs
0000174D C706[9003][9403]           mov     word [MagicBackdoor],NullBackdoor
                                pre_exit_err:
```

```
28132 00001753 B84000                    mov     ax,40h                  ; SYSPRE_BADFILE_ERROR
28133                                                                    ; (problem loading dblspace.bin)
28134 00001756 C3                        retn
28135
28136                                no_driver_version_fail:
28137 00001757 09C0                      or      ax,ax                   ; error code returned?
28138 00001759 75EE                      jnz     short fail_driver_load
28139
28140                                magic_is_resident:
28141                                    ; 13/04/2024
28142                                    ; ds = cs
28143                                    ;mov    ax,[cs:break_addr]
28144 0000175B A1[7D03]                  mov     ax,[break_addr]
28145 0000175E E8B4FB                    call    ParaRound               ; convert to paragraphs
28146                                    ;add    ax,[cs:break_addr+2]
28147                                    ;mov    [cs:DevBrkAddr+2],ax
28148                                    ;mov    word [cs:DevBrkAddr],0; store normalized end here
28149 00001761 0306[7F03]                add     ax,[break_addr+2]
28150 00001765 A3[3F24]                  mov     [DevBrkAddr+2],ax
28151 00001768 C706[3D24]0000            mov     word [DevBrkAddr],0
28152 0000176E BB0400                    mov     bx,4                    ; inquire how many paragraphs it wants
28153                                    ;call   far [cs:MagicBackdoor]
28154 00001771 FF1E[9003]                call    far [MagicBackdoor]
28155                                    ;mov    bx,[cs:ALLOCLIM]        ; get top of free memory
28156 00001775 8B1E[A502]                mov     bx,[ALLOCLIM]
28157 00001779 29C3                      sub     bx,ax                   ; see how much we'll lower it
28158                                    ;cmp    bx,[cs:DevBrkAddr+2]    ; is there that much room free?
28159 0000177B 3B1E[3F24]                cmp     bx,[DevBrkAddr+2]
28160 0000177F 7212                      jb      short cant_move_driver
28161                                    ;;sub   [cs:ALLOCLIM],ax        ; (mov [cs:ALLOCLIM],bx)
28162                                    ;mov    [cs:ALLOCLIM],bx ; Retro DOS v5.0 ; 08/04/2024
28163                                    ; 13/04/2024
28164 00001781 891E[A502]                mov     [ALLOCLIM],bx
28165                                    ;mov    es,[cs:ALLOCLIM]
28166 00001785 8E06[A502]                mov     es,[ALLOCLIM]
28167 00001789 BB0600                    mov     bx,6                    ; tell the driver to move itself
28168                                    ;call   far [cs:MagicBackdoor]
28169 0000178C FF1E[9003]                call    far [MagicBackdoor]
28170                                    ;mov    [cs:DevBrkAddr+2],ax    ; save end of low stub
28171 00001790 A3[3F24]                  mov     [DevBrkAddr+2],ax       ; save end of low stub
28172
28173                                cant_move_driver:
28174                                    ;mov    ax,[cs:DevBrkAddr+2]    ; get terminate segment
28175 00001793 A1[3F24]                  mov     ax,[DevBrkAddr+2]
28176                                    ;cmp    ax,[cs:DevLoadEnd]      ; terminate size TOO big?
28177 00001796 3B06[3724]                cmp     ax,[DevLoadEnd]
28178 0000179A 77B7                      ja      short pre_exit_err      ; error out if so
28179
28180                                ;------ deal with block device drivers
28181
28182                                _isblock:                           ; if no units found,erase the device
28183                                    ; 13/04/2024
28184                                    ; ds = cs
28185                                    ;mov    al,[cs:unitcount]
28186 0000179C A0[7C03]                  mov     al,[unitcount]
28187 0000179F 08C0                      or      al,al
28188 000017A1 74B0                      jz      short pre_exit_err
28189 000017A3 30E4                      xor     ah,ah
28190                                    ;lds    si,[cs:DevEntry]        ; set ds:si to header
28191 000017A5 C536[3924]                lds     si,[DevEntry]
28192 000017A9 88440A                    mov     [si+10],al              ; mov [si+SYSDEV.NAME],al
28193                                                                    ; number of units in name field
28194                                                                    ; device drivers are *supposed*
28195                                                                    ; to do this for themselves.
28196 000017AC 89C1                      mov     cx,ax
28197 000017AE 2EC43E[6D02]              les     di,[cs:DOSINFO]                 ; es:di point to dos info
28198 000017B3 268A6520                  mov     ah,[es:di+20h]          ; [es:di+SYSI_NUMIO]
28199                                                                    ; get number of devices
28200 000017B7 88E2                      mov     dl,ah
28201 000017B9 00C4                      add     ah,al                   ; check for too many devices
28202 000017BB 80FC1A                    cmp     ah,26                   ; 'A' - 'Z' is 26 devices
28203 000017BE 7793                      ja      short pre_exit_err
28204 000017C0 2E800E[6919]02            or      byte [cs:setdevmarkflag],2
28205 000017C6 E83D1F                    call    DevSetBreak
28206                                    ;jnc    short _ok_block
28207                                    ;jmp    pre_exit_err
28208                                    ; 13/04/2024
28209 000017C9 7288                      jc      short pre_exit_err  ; ds <> cs
28210
28211                                _ok_block:
28212 000017CB 26886520                  mov     [es:di+20h],ah          ; [es:di+SYSI_NUMIO] ; update the amount
28213
28214 000017CF 2EC51E[8103]               lds     bx,[cs:bpb_addr]        ; point to bpb array (*)
28215 000017D4 30F6                      xor     dh,dh
28216
28217                                _perunit:
28218 000017D6 2EC42E[6D02]              les     bp,[cs:DOSINFO]
28219 000017DB 26C46E00                  les     bp,[es:bp+0]            ; [es:bp.sysi_dpb]
28220                                                                    ; get first dpb
28221                                                                    ; [es:bp+SysInitvars.SYSI_DPB] ; [es:bp+0]
28222                                _scandpb:
28223 000017DF 26837E19FF                cmp     word [es:bp+19h],0FFFFh ; -1 ; [es:bp.dpb_next_dpb]
28224 000017E4 7406                      jz      short _foundpb
28225 000017E6 26C46E19                  les     bp,[es:bp+19h]          ; les bp,[es:bp.dpb_next_dpb]
28226                                                                    ; [es:bp+DPB.NEXT_DPB]
28227 000017EA EBF3                      jmp     short _scandpb
28228
28229                                ;   We've found the end of the DPB chain. Now extend it.
28230
28231                                _foundpb:
28232 000017EC 2EA1[3D24]                mov     ax,[cs:DevBrkAddr]
28233 000017F0 26894619                  mov     [es:bp+19h],ax          ; [es:bp.dpb_next_dpb] ; DPB.NEXT_DPB
28234 000017F4 2EA1[3F24]                mov     ax,[cs:DevBrkAddr+2]
28235 000017F8 2689461B                  mov     [es:bp+1Bh],ax          ; [es:bp.dpb_next_dpb+2] ; DPB.NEXT_DPB+2
28236 000017FC 2EC42E[3D24]              les     bp,[cs:DevBrkAddr]
28237 00001801 26C74619FFFF              mov     word [es:bp+19h],0FFFFh ; -1
28238 00001807 26C64618FF                mov     byte [es:bp+18h],0FFh ; [es:bp.dpb_first_access],-1
28239                                                                    ; DPB.FIRST_ACCESS
28240 0000180C 2E8306[3D24]3D            add     word [cs:DevBrkAddr],61 ; DPBSIZ ; 3Dh
28241 00001812 E8D01E                    call    RoundBreakAddr
28242 00001815 8B37                      mov     si,[bx]                 ; ds:si points to bpb (*)
28243                                                                    ; (mov si,[bx] ..and then.. add bx,2)
28244                                                                    ; Note: If unit count > 1,bx points to a BPB in the BPB array,
28245                                                                    ; the array address is in [bpb_addr] (*)
28246                                                                    ; Erdogan Tan - 07/07/2023
28247 00001817 26885600                  mov     [es:bp+0],dl                    ; mov word [es:bp.dpb_drive],dx
28248                                                                    ; [es:bp+DPB.DRIVE],dl
28249 0000181B 26887601                  mov     [es:bp+1],dh                    ; [es:bp+DPB.UNIT],dh
28250 0000181F 52                        push    dx
28251 00001820 51                        push    cx
28252 00001821 BA5241                    mov     dx,4152h                ; DX = signature 4152h ('AR') for FAT32 extended BPB/DPB
28253 00001824 31C9                      xor     cx,cx   ; 0
28254 00001826 26894E1D                  mov     [es:bp+1Dh],cx          ; DPB.NEXT_FREE ; last allocated cluster #
28255 0000182A 394C0B                    cmp     [si+0Bh],cx             ; BPB.fatsecs16 ; [si+A_BPB.BPB_SECTORSPERFAT]
```

```
28256 0000182D 7514                     jnz     short _setdpb           ; FAT DPB (33 bytes)
28257                                                                   ; FAT32 DPB (61 bytes)
28258 0000182F 26894E39                 mov     [es:bp+39h],cx          ; DPB.FAT32_NXTFREE = 0
28259 00001833 26894E3B                 mov     [es:bp+3Bh],cx          ; DPB.FAT32_NXTFREE+2 = 0
28260 00001837 49                       dec     cx                      ; 0FFFFh ; -1
28261 00001838 26894E1F                 mov     [es:bp+1Fh],cx          ; DPB.FREE_CNT (-1 = unknown)
28262 0000183C 26894E21                 mov     [es:bp+21h],cx          ; DPB.FREE_CNT+2 (-1 = unknown)
28263 00001840 B95845                   mov     cx,4558h                ; CX = signature 4558h ('EX') for FAT32 extended BPB/DPB
28264
28265                                 _setdpb:
28266 00001843 B453                     mov     ah,53h          ; SETDPB ; hidden system call
28267 00001845 CD21                     int     21h             ; DOS - 2+ internal - TRANSLATE BIOS PARAMETER BLOCK
28268                                                           ; DS:SI -> BPB (BIOS Parameter Block)
28269                                                           ; ES:BP -> buffer for DOS Drive Parameter Block
28270                                                           ; (if CX=4558h & DX=4152h,FAT32 Extended DPB will be set)
28271 00001847 59                       pop     cx
28272 00001848 5A                       pop     dx
28273 00001849 268B4602                 mov     ax,[es:bp+2]            ; [es:bp.dpb_sector_size]  ; [es:bp+DPB.SECTOR_SIZE]
28274 0000184D 06                       push    es
28275 0000184E 2EC43E[6D02]             les     di,[cs:DOSINFO]
28276 00001853 263B4510                 cmp     ax,[es:di+10h]          ; [es:di.sysi_maxsec] ; [es:di+SysInitvars.SYSI_MAXSEC]
28277 00001857 07                       pop     es
28278 00001858 7604                     jbe     short _iblk_1
28279                                   ; 13/04/2024
28280                                   ; ds <> cs
28281 0000185A B84000                   mov     ax,40h                  ; SYSPRE_BADFILE_ERROR ; (pre_exit_err)
28282                                                                   ; (problem loading dblspace.bin)
28283 0000185D C3                       retn
28284
28285                                 _iblk_1:
28286 0000185E 1E                       push    ds
28287 0000185F 2EC506[3924]             lds     ax,[cs:DevEntry]
28288 00001864 26894613                 mov     [es:bp+13h],ax          ; [es:bp+DPB.DRIVER_ADDR]
28289 00001868 268C5E15                 mov     [es:bp+15h],ds          ; [es:bp+DPB.DRIVER_ADDR+2]
28290 0000186C 1F                       pop     ds
28291 0000186D FEC2                     inc     dl                      ; increment drive number
28292 0000186F FEC6                     inc     dh                      ; increment unit number
28293 00001871 43                       inc     bx
28294 00001872 43                       inc     bx                      ; point to next BPB
28295                                                                   ; (in the BPB array) (*) -add bx,2-
28296 00001873 49                       dec     cx                      ; loop _foundpb
28297 00001874 7403                     jz      short _linkit
28298 00001876 E973FF                   jmp     _foundpb
28299
28300                                 _linkit:
28301 00001879 0E                       push    cs
28302 0000187A 1F                       pop     ds
28303 0000187B E825F5                   call    TempCDS                 ; set cds for new drives
28304                                   ; 13/04/2024
28305                                   ; (DS may not be same with CS here)
28306 0000187E 2EC43E[6D02]             les     di,[cs:DOSINFO]                 ; es:di = dos table (SysInitVars)
28307 00001883 268B4522                 mov     ax,[es:di+22h]          ; [es:di+SYSI_DEV] ; dx:cx = head of list
28308 00001887 268B5D24                 mov     bx,[es:di+24h]          ; [es:di+SYSI_DEV+2]
28309 0000188B 2EC536[3924]             lds     si,[cs:DevEntry]        ; ds:si = device location
28310 00001890 8904                     mov     [si],ax                 ; link in the driver
28311 00001892 895C02                   mov     [si+2],bx
28312 00001895 26897522                 mov     [es:di+22h],si          ; [es:di+SYSI_DEV] ; set head of list in dos
28313 00001899 268C5D24                 mov     [es:di+24h],ds          ; [es:di+SYSI_DEV+2]
28314 0000189D E8881E                   call    DevBreak                ; mark successful install
28315                                   ; 13/04/2024
28316                                   ; ds = cs
28317                                   ;mov    cx,[cs:DevBrkAddr+2]    ; pass it a work buffer
28318                                   ;mov    dx,[cs:ALLOCLIM]        ; address in cx (segment)
28319 000018A0 8B0E[3F24]               mov     cx,[DevBrkAddr+2]
28320 000018A4 8B16[A502]               mov     dx,[ALLOCLIM]
28321 000018A8 29CA                     sub     dx,cx                   ; for len dx (paragraphs)
28322 000018AA B80055                   mov     ax,5500h                ; we're shuffle aware,but don't move
28323                                                                   ; any drives at this point.
28324 000018AD BB0200                   mov     bx,2                    ; switch what we can now
28325                                   ;call   far [cs:MagicBackdoor]
28326 000018B0 FF1E[9003]               call    far [MagicBackdoor]
28327                                 pre_exit:                         ; no errors!
28328 000018B4 31C0                     xor     ax,ax                   ; zf=1
28329                                 no_magic:  ; 13/04/2024
28330 000018B6 C3                       retn
28331
28332                                 ; =============== S U B R O U T I N E =======================================
28333
28334                                 ; 08/04/2024 - Retro DOS v5.0
28335                                 ; (PCDOS 7.1 IBMBIO.COM - SYSINIT:1B9Fh)
28336
28337                                 ;***MagicPostload -- called to clean up and make sure Magic is final placed
28338
28339                                 MagicPostload:
28340                                     ; 13/04/3024
28341                                     ; ds = cs
28342 000018B7 E89C00                   call    get_dblspace_version  ; is it there?
28343 000018BA 75FA                     jnz     short no_magic        ; done if not
28344 000018BC F7C20080                 test    dx,8000h              ; is it already permanent?
28345 000018C0 74F4                     jz      short no_magic        ; no,done if so (not in final position)
28346 000018C2 BBFFFF                   mov     bx,0FFFFh ; -1        ; how much space does it want?
28347 000018C5 B8114A                   mov     ax,4A11h              ; multMagicdrv
28348                                                                 ; DBLSPACE.BIN - GET RELOCATION SIZE
28349 000018C8 CD2F                     int     2Fh                   ; get paragraphs into ax
28350 000018CA 40                       inc     ax                    ; extra 2 paragraphs for the stub
28351 000018CB 40                       inc     ax                    ; ((tiny_stub_end-tiny_stub_start)+15)/16
28352                                                                 ; (18+15)/16 = 2
28353                                   ;mov    [cs:DevSize],ax             ; store that (**)
28354                                   ;mov    byte [cs:DeviceHi],0  ; not to be loaded in UMB
28355                                   ;mov    [cs:bpb_addr+2],cs    ; pass name so that
28356                                   ;                             ; arena header can be set
28357                                   ;mov    word [cs:bpb_addr],MagicDDName ; "\DBLSPACE.BIN"
28358                                   ; 13/04/2024
28359                                   ; ds = cs
28360 000018CC A3[3324]                 mov     [DevSize],ax
28361 000018CF C606[5124]00             mov     byte [DeviceHi],0
28362 000018D4 8C0E[8303]               mov     [bpb_addr+2],cs
28363 000018D8 C706[8103][AB16]         mov     word [bpb_addr],MagicDDName
28364
28365 000018DE E8112F                   call    round                 ; normalize memhi:memlo
28366 000018E1 E8811C                   call    InitDevLoad           ; set up sub-arena,DevLoadAddr,
28367                                                                 ; DevLoadEnd,and DevEntry
28368                                                                 ; gets arena name from bpb_addr
28369                                   ; 13/04/2024
28370                                   ; ds = cs
28371                                   ;mov    es,[cs:DevLoadAddr]   ; (**) (InitDevload sets this)
28372 000018E4 8E06[3524]               mov     es,[DevLoadAddr]
28373
28374                                 ;   First, move a little header in place so that this looks
28375                                 ;   to the mem command like a legitimate driver load. Otherwise,
28376                                 ;   it will display garbage for the device name
28377
28378 000018E8 31FF                     xor     di,di                 ; move a little header in place
28379                                                                 ; so that this looks to the mem command
```

```
                                                        ; like a legitimate driver load
000018EA BE[C816]               mov     si,tiny_stub_start
                                ;mov    cx,18           ; (tiny_stub_end-tiny_stub_start)
000018ED B91200                 mov     cx,tiny_stub_end-tiny_stub_start
000018F0 F3A4                   rep movsb               ; move it!
000018F2 8CC0                   mov     ax,es           ; advance es appropriately
000018F4 40                     inc     ax              ; add ax,((tiny_stub_end-tiny_stub_start)+15)/16
000018F5 40                     inc     ax
000018F6 8EC0                   mov     es,ax
000018F8 BBFEFF                 mov     bx,0FFFEh ; -2  ; final placement!
000018FB B8114A                 mov     ax,4A11h        ; multMagicdrv
000018FE CD2F                   int     2Fh             ; DBLSPACE.BIN - RELOCATE
                                                        ; es = segment to which to relocate DBLSPACE.BIN
                                ;mov    ax,[cs:DevLoadAddr] ; (**)
                                ;add    ax,[cs:DevSize]         ; calculate seg after DD load
                                ;mov    [cs:DevBrkAddr+2],ax  ; save as ending address!
                                ;mov    word [cs:DevBrkAddr],0
                                ; 13/04/2024
                                ; ds = cs
00001900 A1[3524]               mov     ax,[DevLoadAddr]
00001903 0306[3324]             add     ax,[DevSize]
00001907 A3[3F24]               mov     [DevBrkAddr+2],ax
0000190A C706[3D24]0000         mov     word [DevBrkAddr],0

00001910 E8F31D                 call    DevSetBreak     ; go ahead and alloc mem for device
                                ;call   DevBreak
                            ;no_magic:
                                ;retn
                                ; 13/04/2024 - Retro DOS v5.0
00001913 E9121E                 jmp     DevBreak

                            ; =============== S U B R O U T I N E ===================================

                            ; 08/04/2024 - Retro DOS v5.0
                            ; (PCDOS 7.1 IBMBIO.COM - SYSINIT:1C08h)

                            ;***MagicSetCdss -- disable CDSs for still unmounted DblSpace drives
                            ;   entry:
                            ;      CDSs are now persistent and in their final place

                            MagicSetCdss:
                                ; 13/04/2024 - Retro DOS v5.0
                                ; ds = cs
00001916 E83D00                 call    get_dblspace_version  ; is it there?
00001919 753A                   jnz     short magic_set_exit  ; done if not
                                                        ; cl = first DblSpace drive in ASCII
                                                        ; ch = number of DblSpace drive letters
0000191B 2EC536[6D02]           lds     si,[cs:DOSINFO]         ; point to DOS data area (SysInitVars)
00001920 C57416                 lds     si,[si+16h]     ; lds si,[si+SYSI_CDS] ; fetch CDSs
00001923 B458                   mov     ah,88           ; curdirLen
00001925 80E941                 sub     cl,'A'          ; make it zero based.
00001928 88C8                   mov     al,cl           ; get first DblSpace drive letter
0000192A F6E4                   mul     ah              ; find first DblSpace CDS
0000192C 01C6                   add     si,ax           ; cds pointer
0000192E 88CA                   mov     dl,cl           ; save for drive testing loop
00001930 88E9                   mov     cl,ch           ; get DblSpace drive count into cx
00001932 30ED                   xor     ch,ch

                            ;   We know cx > 0, or else the driver wouldn't have stayed resident

                            magic_set_cdss_1:
00001934 51                     push    cx
00001935 52                     push    dx
00001936 1E                     push    ds
00001937 56                     push    si
00001938 B8114A                 mov     ax,4A11h        ; multMagicdrv
0000193B BB0100                 mov     bx,1            ; MD_DRIVE_MAP ; inquire drive map
0000193E CD2F                   int     2Fh             ; DBLSPACE.BIN - "GetDriveMapping"
                                                        ; see if this is an unused DblSpace drive
00001940 5E                     pop     si
00001941 1F                     pop     ds
00001942 5A                     pop     dx
00001943 59                     pop     cx
00001944 38DA                   cmp     dl,bl           ; if mapped to itself,it is vacant
00001946 7504                   jnz     short magic_set_cdss_2; skip if used
00001948 806444BF               and     byte [si+44h],0BFh    ; Retro DOS v5.0 ; 08/04/2024
                                ;and    word [si+43h],0BFFFh
                                                        ; reset the bit in flags (curdir_inuse bit)
                                                        ; [si+curdir_list.cdir_flags],~curdir_inuse ; word
                                                        ; (.. [si+1+curdir_list.cdir_flags],0BFh ; byte)
                            magic_set_cdss_2:
0000194C 83C658                 add     si,88           ; curdirLen
0000194F FEC2                   inc     dl              ; next drive
00001951 E2E1                   loop    magic_set_cdss_1
                                ; 13/04/2024
00001953 0E                     push    cs
00001954 1F                     pop     ds

                            magic_set_exit:
00001955 C3                     retn

                            ; =============== S U B R O U T I N E ===================================

                            ; 08/04/2024 - Retro DOS v5.0
                            ; (PCDOS 7.1 IBMBIO.COM - SYSINIT:1C47h)

                            get_dblspace_version:
00001956 B8114A                 mov     ax,4A11h        ; multMagicdrv
                                                        ; DBLSPACE.BIN - "GetVersion" - INSTALLATION CHECK
                                                        ; (BX = 0)
00001959 31DB                   xor     bx,bx           ; MD_VERSION = 0
0000195B CD2F                   int     2Fh             ; Return:
                                                        ;  AX = 0000h (successful)
                                                        ;  BX = 444Dh ("DM")
                                                        ;  CL = first drive letter used by DBLSPACE (41h = A:)
                                                        ;  CH = number of drive letters used by DBLSPACE
                                                        ;  DX = internal DBLSPACE.BIN version number (bits 14-0)
                                                        ;  bit 15 set if DBLSPACE.BIN has not yet been relocated
                                                        ;  to final position in memory (i.e. DBLSPACE.SYS /MOVE)
0000195D 09C0                   or      ax,ax           ; ax = 0 (successful,zf=1)
0000195F C3                     retn

                            ; ---------------------------------------------------------------------------
                            ; SYSCONF.ASM - MSDOS 6.0 - 1991
                            ; ---------------------------------------------------------------------------
                            ; 27/03/2019 - Retro DOS v4.0

                            ;MULTI_CONFIG      equ 1

                            HIGH_FIRST  equ 080h            ; from ARENA.INC - modifier for
                                                                ; allocation strategy call

                            ;have_install_cmd equ 00000001b   ; config.sys has install= commands
                            ;has_installed       equ 00000010b       ; sysinit_base installed.
```

```
28504
28505                           default_filenum equ 8
28506
28507                           ;stacksw    equ true              ; include switchable hardware stacks
28508
28509                           ; external variable defined in ibmbio module for multi-track
28510
28511                           ;multrk_on  equ 10000000b         ;user spcified mutitrack=on,or system turns
28512                                                             ; it on after handling config.sys file as a
28513                                                             ; default value,if multrk_flag = multrk_off1.
28514                           ;multrk_off1       equ 00000000b         ;initial value. no "multitrack=" command entered.
28515                           ;multrk_off2       equ 00000001b         ;user specified multitrack=off.
28516
28517                           ; if stacksw
28518
28519                           ; internal stack parameters
28520
28521                           ;entrysize  equ 8
28522
28523                           ;mincount   equ 8
28524                           ;defaultcount      equ 9
28525                           ;maxcount   equ 64
28526
28527                           ;minsize    equ 32
28528                           ;defaultsize       equ 128
28529                           ;maxsize    equ 512
28530
28531                           DOS_FLAG_OFFSET    equ 86h
28532
28533                           ;ifdef MULTI_CONFIG
28534                           ;
28535                           ;   config_envlen must immediately precede config_wrkseg, because they
28536                           ;   may be loaded as a dword ptr
28537
28538                           ; 30/12/2022 - Retro DOS v4.2 (Modified MSDOS 6.21 IO.SYS)
28539                           ; 25/10/2022
28540 00001960 0000            config_envlen:     dw  0                 ; when config_wrkseg is being used as
28541                                                                    ;  a scratch env, this is its length
28542 00001962 0000            config_wrkseg:     dw  0                 ; config work area (above confbot)
28543                                                                    ;  segment of work area
28544
28545 00001964 00              config_cmd: db  0                 ; current config cmd
28546                                                             ;  (with CONFIG_OPTION_QUERY bit intact)
28547 00001965 00              config_multi:      db  0                  ; non-zero if multi-config config.sys
28548
28549                           ;endif ; MULTI_CONFIG
28550
28551 00001966 00              multdeviceflag:    db  0
28552
28553 00001967 0000            devmark_addr:      dw  0                 ;segment address for devmark.
28554
28555 00001969 00              setdevmarkflag: db  0                    ;flag used for devmark
28556
28557                           ; 30/12/2022
28558                           ; 12/12/2022
28559 0000196A 00              driver_units:      db  0                 ;total unitcount for driver
28560
28561                           ; 12/12/2022
28562                           ;ems_stub_installed:
28563                           ;          db  0
28564
28565                           ; 12/12/2022
28566                           ;align 2
28567
28568                           badparm_ptr:       ; label dword
28569 0000196B 0000            badparm_off:       dw  0
28570 0000196D 0000            badparm_seg:       dw  0
28571
28572                           ;****************************************************************************
28573                           ;take care of config.sys file.
28574                           ;system parser data and code.
28575                           ;****************************************************************************
28576
28577                           ;*********************************************************************
28578                           ; parser options set for msbio sysconf module
28579                           ;*********************************************************************
28580                           ;
28581                           ;**** default assemble swiches definition **************************
28582
28583                           ;farsw      equ 0          ; near call expected
28584                           ;datesw     equ 0          ; check date format
28585                           ;timesw     equ 0          ; check time format
28586                           ;filesw     equ 1          ; check file specification
28587                           ;capsw      equ 0          ; perform caps if specified
28588                           ;cmpxsw     equ 0          ; check complex list
28589                           ;numsw      equ 1          ; check numeric value
28590                           ;keysw      equ 0          ; support keywords
28591                           ;swsw       equ 1          ; support switches
28592                           ;val1sw     equ 1          ; support value definition 1
28593                           ;val2sw     equ 0          ; support value definition 2
28594                           ;val3sw     equ 1          ; support value definition 3
28595                           ;drvsw      equ 1          ; support drive only format
28596                           ;qussw      equ 0          ; support quoted string format
28597
28598                           ; psdata_seg equ cs
28599
28600                               ;.xlist
28601                               ;include parse.asm           ;together with psdata.inc
28602                               ;.list
28603
28604                           ; PSDATA.INC - MSDOS 6.0 - 1991
28605                           ; ============================================================================
28606                           ; 27/03/2019 - Retro DOS v4.0
28607
28608                           ; 30/03/2019
28609                           ; VERSION.INC (MSDOS 6.0)
28610                           ; Set DBCS Blank constant
28611
28612                           ; ifndef DBCS
28613                           DB_SPACE EQU 2020h
28614                           DB_SP_HI EQU 20h
28615                           DB_SP_LO EQU 20h
28616                           ; else
28617
28618                           ;*********************************************************************
28619                           ; Parser include file
28620                           ;*********************************************************************
28621
28622                           ;**** Equation field
28623                           ;-------- Character code definition
28624
28625                           _$P_DBSP1      equ  DB_SP_HI       ;AN000; 1st byte of DBCS blank
28626                           _$P_DBSP2      equ  DB_SP_LO       ;AN000; 2nd byte of DBCS blank
28627                           _$P_Period     equ  "."             ;AN020;
```

```
                                _$P_Slash       equ "/"             ;AN020;
                                _$P_Space       equ " "             ;AN000; SBCS blank
                                _$P_Comma       equ ","             ;AN000;
                                _$P_Switch      equ "/"             ;AN000;
                                _$P_Keyword     equ "="             ;AN000;
                                _$P_Colon       equ ":"             ;AN000;
                                _$P_Plus        equ "+"             ;AN000;
                                _$P_Minus       equ "-"             ;AN000;
                                _$P_Rparen      equ ")"             ;AN000;
                                _$P_Lparen      equ "("             ;AN000;
                                ;_$P_SQuote         equ  "'"                    ;AN025; deleted
                                _$P_DQuote      equ '"'             ;AN000;
                                _$P_NULL        equ 0               ;AN000;
                                _$P_TAB                 equ 9               ;AN000;
                                _$P_CR                  equ 0Dh             ;AN000;
                                _$P_LF                  equ 0Ah             ;AN000;
                                _$P_ASCII80     equ 80h             ;AN000; ASCII 80h character code

                                ;-------- Masks
                                _$P_Make_Lower          equ 20h             ;AN000; make lower case character
                                _$P_Make_Upper          equ 0FFh-_$P_Make_Lower ;AN000; make upper case character

                                ;-------- DOS function call related equs

                                _$P_DOS_Get_CDI         equ 3800h           ;AN000; get country dependent information
                                                                            ; by this call, following information
                                struc _$P_CDI
 00000000 ????                   .DateF: resw 1
 00000002 ??????????             .Money: resb 5
 00000007 ????                   .1000:      resb 2
 00000009 ????                   .Dec:       resb 2
 0000000B ????                   .DateS: resb 2
 0000000D ????                   .TimeS: resb 2
 0000000F ??                         resb 1
 00000010 ??                         resb 1
 00000011 ??                     .TimeF: resb 1
 00000012 ????????                   resw 2
 00000016 ????                       resb 2
 00000018 <res Ah>                   resw 5
                                 .size:
                                endstruc

                                _$P_Date_MDY            equ 0               ;AN000;
                                _$P_Date_DMY            equ 1               ;AN000;
                                _$P_Date_YMD            equ 2               ;AN000;
                                ;-------------
                                _$P_DOS_GetEV           equ 6300h           ;AN000; get DBCS EV call
                                                                            ;AN000; DS:SI will points to DBCS EV
                                ;-------------
                                _$P_DOS_Get_TBL         equ 65h             ;AN000; get uppercase table call
                                                                            ;AN000; following parameters are set
                                                                            ;AN000; to get casemap table.
                                _$P_DOSTBL_Def          equ -1              ;AN000; get default
                                _$P_DOSTBL_BL           equ 5               ;AN000; buffer length for Tbl pointer
                                _$P_DOSTBL_File         equ 4               ;AN000; get file uppercase table
                                _$P_DOSTBL_Char         equ 2               ;AN000; get character uppercase table
                                                                            ; By this call following information
                                                                            ; is returned.
                                struc _$P_DOS_TBL
 00000000 ??                     .InfoID: resb 1                 ;AN000; information id for the table
 00000001 ????                   .Off:       resw 1              ;AN000; offset address of the table
 00000003 ????                   .Seg:       resw 1              ;AN000; segment address of the table
                                endstruc

                                ; ---------------------------------------------------------------------------
                                ; PARMS     LABEL   BYTE
                                ;           DW      PARMSX
                                ;           DB      2               ; NUMBER OF STRINGS (0, 1, 2)
                                ;           DB      length          ; LENGTH OF THE NEXT LIST, 0 IF NONE
                                ;           DB      " .. "           ; EXTRA DELIMITER LIST,
                                ;                                   ; TYPICAL ARE ";", "="
                                ;                                   ; "," & WHITESPACE ALWAYS
                                ;           DB      length          ; LENGTH OF THE NEXT LIST, 0 IF NONE
                                ;           DB      " .. "           ; EXTRA END OF LINE LIST, CR, LF OR 0 ALWAYS
                                ; ---------------------------------------------------------------------------

                                ;-------------------------------- PARMS block structure
                                struc _$P_PARMS_Blk
 00000000 ????                   .PARMSX_Address:  resw 1        ;AN000; Address of PARMSX
 00000002 ??                     .Num_Extra:         resb 1              ;AN000; Number of extra stuff
 00000003 ??                     .Len_Extra_Delim: resb 1        ;AN000; Length of extra delimiter
                                endstruc

                                _$P_Len_PARMS           equ 4               ;AN000;
                                _$P_I_Use_Default   equ     0               ;AN000; no extra stuff specified
                                _$P_I_Have_Delim    equ     1               ;AN000; extra delimiter specified
                                _$P_I_Have_EOL          equ 2               ;AN000; extra EOL specified

                                ; ---------------------------------------------------------------------------
                                ; PARMSX    LABEL   BYTE
                                ;           DB      minp,maxp       ; MIN, MAX POSITIONAL OPERANDS ALLOWED
                                ;           DW      CONTROL         ; DESCRIPTION OF POSITIONAL 1
                                ;           :                       ; REPEATS maxp-1 TIMES
                                ;           DB      maxs            ; # OF SWITCHES
                                ;           DW      CONTROL         ; DESCRIPTION OF SWITCH 1
                                ;           :                       ; REPEATS maxs-1 TIMES
                                ;           DB      maxk            ; # OF KEYWORD
                                ;           DW      CONTROL         ; DESCRIPTION OF KEYWORD 1
                                ;           :                       ; REPEATS maxk-1 TIMES
                                ; ---------------------------------------------------------------------------

                                ;-------------------------------- PARMSX block structure
                                struc _$P_PARMSX_Blk                ;AN000;
 00000000 ??                     .MinP: resb 1                      ;AN000; Minimum positional number
 00000001 ??                     .MaxP:     resb 1                  ;AN000; Maximum positional number
 00000002 ????                   .1st_Control: resw 1               ;AN000; Address of the 1st CONTROL block
                                endstruc

                                ; ---------------------------------------------------------------------------
                                ; << Control field definition  >>
                                ;
                                ;
                                ;CONTROL   LABEL   BYTE
                                ;       DW      MATCH_FLAGS     ; CONTROLS TYPE MATCHED
                                ;                               ; 8000H=NUMERIC VALUE, (VALUE LIST WILL BE CHECKED)
                                ;                               ; 4000H=SIGNED NUMERIC VALUE (VALUE LIST WILL BE CHECKED)
                                ;                               ; 2000H=SIMPLE STRING(VALUE LIST WILL BE CHECKED)
                                ;                               ; 1000H=DATE STRING (VALUE LIST WON'T BE CHECKED)
                                ;                               ; 0800H=TIME STRING (VALUE LIST WON'T BE CHECKED)
                                ;                               ; 0400H=COMPLEX LIST (VALUE LIST WON'T BE CHECKED)
                                ;                               ; 0200H=FILE SPEC (VALUE LIST WON'T BE CHECKED)
                                ;                               ; 0100H=DRIVE ONLY (VALUE LIST WON'T BE CHECKED)
                                ;                               ; 0080H=QUOTED STRING (VALUE LIST WON'T BE CHECKED)
                                ;                               ; 0010H=IGNORE ":" AT END IN MATCH
```

```
28752                           ;                               ; 0002H=REPEATS ALLOWED
28753                           ;                               ; 0001H=OPTIONAL
28754                           ;       DW      FUNCTION_FLAGS
28755                           ;                               ; 0001H=CAP RESULT BY FILE TABLE
28756                           ;                               ; 0002H=CAP RESULT BY CHAR TABLE
28757                           ;                               ; 0010H=REMOVE ":" AT END
28758                           ; (tm10)                        ; 0020H=colon is not necessary for switch
28759                           ;       DW      RESULT          ; RESULT BUFFER
28760                           ;       DW      VALUES          ; VALUE LISTS
28761                           ;       DB      nid             ; NUMBER OF KEYWORD/SWITCH SYNONYMS IN FOLLOWING LIST
28762                           ;       DB      "...",0          ; IF n >0, KEYWORD 1
28763                           ;       :
28764                           ;
28765                           ;Note:
28766                           ;     - The MATCH_FLAG is bit significant. You can set, for example, TIME bit and
28767                           ;       DATE bit simalteniously.
28768                           ;
28769                           ;       The parser examins each bit along with the following priority.
28770                           ;
28771                           ;       COMPLEX -> DATE -> TIME -> NUMERIC VAL -> SIGNED NUMERIC VAL -> DRIVE ->
28772                           ;       FILE SPEC -> SIMPLE STRING.
28773                           ;
28774                           ;     - When the FUNCTION_FLAG is 0001 or 0002, the STRING pointed to by a pointer
28775                           ;       in the result buffer is capitalized.
28776                           ;
28777                           ;     - Match_Flags 0001H and 0002H have meaning only for the positional.
28778                           ;
28779                           ;     - The "...",0 (bottom most line) does require '=' or '/'. When you need a
28780                           ;       switch, for example, '/A', then STRING points to;
28781                           ;
28782                           ;                   DB    1         ; number of following synonyms
28783                           ;                   DB   '/A',0
28784                           ;
28785                           ;       When you need a keyword, for example, 'CODEPAGE=', then "...",0 will be;
28786                           ;
28787                           ;                   DB    1         ; number of following synonyms
28788                           ;                   DB   'CODEPAGE=',0
28789                           ;
28790                           ;     - "..." must consist of upper case characters only because the parser
28791                           ;       performs pattern matching after converting input to upper case (by
28792                           ;       using the current country upper case table)
28793                           ;
28794                           ;     - One "..." can contain only one switch or keyword. If you need, for
28795                           ;       example /A and /B, the format will be;
28796                           ;
28797                           ;                   DB    2         ; number of following synonyms
28798                           ;                   DB    '/A',0
28799                           ;                   DB    '/B',0
28800                           ; ---------------------------------------------------------------------------
28801
28802                           ;**** Match_Flags
28803
28804                           _$P_Num_Val    equ  8000h          ;AN000; Numeric Value
28805                           _$P_SNum_Val          equ  4000h          ;AN000; Signed numeric value
28806                           _$P_Simple_S          equ  2000h          ;AN000; Simple string
28807                           _$P_Date_S     equ  1000h          ;AN000; Date string
28808                           _$P_Time_S     equ  0800h          ;AN000; Time string
28809                           _$P_Cmpx_S     equ  0400h          ;AN000; Complex string
28810                           _$P_File_Spc          equ  0200h          ;AN000; File Spec
28811                           _$P_Drv_Only          equ  0100h          ;AN000; Drive Only
28812                           _$P_Qu_String         equ  0080h          ;AN000; Quoted string
28813                           _$P_Ig_Colon          equ  0010h          ;AN000; Ignore colon at end in match
28814                           _$P_Repeat     equ  0002h          ;AN000; Repeat allowed
28815                           _$P_Optional          equ  0001h          ;AN000; Optional
28816
28817                           ;**** Function flags
28818
28819                           _$P_CAP_File          equ  0001h          ;AN000; CAP result by file table
28820                           _$P_CAP_Char          equ  0002h          ;AN000; CAP result by character table
28821                           _$P_Rm_Colon          equ  0010h          ;AN000; Remove ":" at the end
28822                           _$P_colon_is_not_necessary equ 0020h      ;AN000;(tm10) /+10 and /+:10
28823
28824                           ;-------------------------------- Control block structure
28825                           struc _$P_Control_Blk
28826 00000000 ????              .Match_Flag:       resw 1          ;AN000; Controls type matched
28827 00000002 ????              .Function_Flag: resw 1             ;AN000; Function should be taken
28828 00000004 ????              .Result_Buf:       resw 1          ; Result buffer address
28829 00000006 ????              .Value_List:       resw 1          ;AN000; Value list address
28830 00000008 ??                .nid:              resb 1          ;AN000; # of keyword/SW synonyms
28831 00000009 ??                .KEYorSW:   resb 1          ;AN000; keyword or sw
28832                           endstruc
28833
28834                           ; ---------------------------------------------------------------------------
28835                           ; << Value List Definition >>
28836                           ;
28837                           ;VALUES     LABEL   BYTE
28838                           ;           DB      nval                ; NUMBER OF VALUE DEFINITIONS (0 - 3)
28839                           ;          +-
28840                           ;           | DB      nrng              ; NUMBER OF RANGES
28841                           ;           | +DB     ITEM_TAG          ; RETURN VALUE IF RANGE MATCHED
28842                           ;           | +DD     X,Y               ; RANGE OF VALUES
28843                           ;           | :
28844                           ;           | DB      nnval             ; NUMBER OF CHOICES
28845                           ;           | +DB     ITEM_TAG          ; RETURN VALUE IF NUMBER CHOICE MATCHED
28846                           ;           | +DD     VALUE             ; SPECIFIC CHOICE IF NUMBER
28847                           ;           | :
28848                           ;           | DB      nstrval           ; NUMBER OF CHOICES
28849                           ;           | +DB     ITEM_TAG          ; RETURN VALUE IF STRING CHOICE MATCHED
28850                           ;           | +DW     STRING            ; SPECIFIC CHOICE IF STING
28851                           ;          +- :
28852                           ;
28853                           ;STRING     DB      "...",0             ; ASCIIZ STRING IMAGE
28854                           ;
28855                           ;Note:
28856                           ;     - ITEM_TAG must not be 0FFH, which will be used in the result buffer
28857                           ;       when no choice lists are provided.
28858                           ;
28859                           ;     - STRING must consist of upper case characters only because the parser
28860                           ;       performs pattern matching after converting input to upper case (by
28861                           ;       using the current country upper case table)
28862                           ; ---------------------------------------------------------------------------
28863
28864                           _$P_nval_None         equ  0            ;AN000; no value list ID
28865                           _$P_nval_Range        equ  1            ;AN000; range list ID
28866                           _$P_nval_Value        equ  2            ;AN000; value list ID
28867                           _$P_nval_String       equ  3            ;AN000; string list ID
28868                           _$P_Len_Range         equ  9            ;AN000; Length of a range choice(two DD plus one DB)
28869                           _$P_Len_Value         equ  5            ;AN000; Length of a value choice(one DD plus one DB)
28870                           _$P_Len_String        equ  3            ;AN000; Length of a string choice(one DW plus one DB)
28871                           _$P_No_nrng    equ  0             ;AN000; (tm07) no nrng. nnval must not be 0.
28872
28873                           struc _$P_Val_List
28874 00000000 ??                .NumofList: resb 1                  ;AN000; number of following choice
28875 00000001 ????              .Val_XL:    resw 1                  ;AN000; lower word of value
```

```
28876 00000003 ????                     .Val_XH:    resw 1                  ;AN000; higher word of value
28877 00000005 ????                     .Val_YL:    resw 1                  ;AN000; lower word of another value
28878 00000007 ????                     .Val_YH:    resw 1                  ;AN000; higher word of another value
28879                                   endstruc
28880
28881                                   ; ---------------------------------------------------------------------------
28882                                   ; << Result Buffer Definition  >>
28883                                   ;
28884                                   ;RESULT     LABEL   BYTE                ; BELOW FILLED IN FOR DEFAULTS
28885                                   ;           DB      type                ; TYPE RETURNED: 0=RESERVED,
28886                                   ;                                       ;       1=NUMBER, 2=LIST INDEX,
28887                                   ;                                       ;       3=STRING, 4=COMPLEX,
28888                                   ;                                       ;       5=FILESPEC, 6=DRIVE
28889                                   ;                                       ;       7=DATE, 8=TIME
28890                                   ;                                       ;       9=QUOTED STRING
28891                                   ;           DB      ITEM_TAG            ; MATCHED ITEM TAG
28892                                   ;
28893                                   ;           dw      synonym@            ; es:@ points to found SYNONYM if provided.
28894                                   ;
28895                                   ;             +-
28896                                   ;         | DD      n                   ; VALUE IF NUMBER
28897                                   ;         | or
28898                                   ;         |  DW     i                   ; INDEX (OFFSET) INTO VALUE LIST
28899                                   ;         |                             ; (ES presents Segment address)
28900                                   ;         | or
28901                                   ;         |  DD     STRING              ; OFFSET OF STRING VALUE
28902                                   ;         | or
28903                                   ;         |  DB     drv                 ; DRIVE NUMBER (1-A, 2-B,..., 26-Z)
28904                                   ;         | or
28905                                   ;         |  DW     YEAR      ;(1980-2099)  IN CASE OF DATE
28906                                   ;         |  DB     MONTH     ;(1-12)      Note: Range check is not performed.
28907                                   ;         |  DB     DATE      ;(1-31)           0 is filled when the corresponding field was not specified.
28908                                   ;         | or
28909                                   ;         |  DB     HOUR      ;(0-23)       IN CASE OF TIME
28910                                   ;         |  DB     MINUTES    ;(0-59)      Note: Range check is not performed .
28911                                   ;         |  DB     SECONDS    ;(0-59)           0 is filled when the corresponding field was not specified .
28912                                   ;         |  DB     HUNDREDTHS ;(0-99)
28913                                   ;        +-
28914                                   ;
28915                                   ;
28916                                   ;Note: ITEM_TAG is 0FFH when the caller does not specify the choice
28917                                   ;      list.
28918                                   ;
28919                                   ;      YEAR: If the input value for the year is less than 100, parser
28920                                   ;        adds 1900 to it.  For example, when 87 is input to parser for
28921                                   ;        the year value, he returns 1987.
28922                                   ; ---------------------------------------------------------------------------
28923
28924                                   ;-------------------------------- Result block structure
28925                                   struc _$P_Result_Blk
28926 00000000 ??                       .Type:        resb 1            ;AN000; Type returned
28927 00000001 ??                       .Item_Tag:    resb 1            ;AN000; Matched item tag
28928 00000002 ????                     .SYNONYM_Ptr: resw 1            ;AN000; pointer to Synonym list returned
28929 00000004 ????????                 .Picked_Val:  resb 4            ;AN000; value
28930                                   endstruc
28931
28932                                   ;---------------------------------
28933                                   ;**** values for the type field in the result block
28934
28935                                   _$P_EOL               equ 0             ;AN000; End of line
28936                                   _$P_Number    equ 1               ;AN000; Number
28937                                   _$P_List_Idx          equ 2             ;AN000; List Index
28938                                   _$P_String    equ 3               ;AN000; String
28939                                   _$P_Complex   equ 4               ;AN000; Complex
28940                                   _$P_File_Spec         equ 5             ;AN000; File Spec
28941                                   _$P_Drive     equ 6               ;AN000; Drive
28942                                   _$P_Date_F    equ 7               ;AN000; Date
28943                                   _$P_Time_F    equ 8               ;AN000; Time
28944                                   _$P_Quoted_String  equ    9             ;AN000; Quoted String
28945
28946                                   _$P_No_Tag    equ 0FFh            ;AN000; No ITEM_TAG found
28947
28948                                   ;**** Return code
28949                                   ;
28950                                   ; following return code will be returned in the AX register.
28951
28952                                   _$P_No_Error          equ 0             ;AN000; No error
28953                                   _$P_Too_Many          equ 1             ;AN000; Too many operands
28954                                   _$P_Op_Missing        equ 2             ;AN000; Required operand missing
28955                                   _$P_Not_In_SW         equ 3             ;AN000; Not in switch list provided
28956                                   _$P_Not_In_Key        equ 4             ;AN000; Not in keyword list provided
28957                                   _$P_Out_Of_Range   equ    6             ;AN000; Out of range specified
28958                                   _$P_Not_In_Val        equ 7             ;AN000; Not in value list provided
28959                                   _$P_Not_In_Str        equ 8             ;AN000; Not in string list provided
28960                                   _$P_Syntax    equ 9               ;AN000; Syntax error
28961                                   _$P_RC_EOL    equ -1              ;AN000; End of command line
28962
28963                                   ; DATA - Retro DOS v4.0 - 27/03/2019
28964
28965                                   ; MSDOS 6.2 IO.SYS SYSINIT:179Ch
28966
28967                                   ; 14/04/2024 - Retro DOS v5.0
28968                                   ; PCDOS 7.1 IBMBIO.COM - SYSINIT:1C62h
28969
28970                                   ;********************** Local Data ************************************
28971 0000196F 0000                   _$P_ORDINAL:          dw  0             ;AN000; Operand ordinal save area
28972 00001971 0000                   _$P_RC:               dw  0             ;AN000; Return code from parser
28973 00001973 0000                   _$P_SI_Save:          dw  0             ;AN000; Pointer of command buffer
28974 00001975 0000                   _$P_DX:               dw  0             ;AN000; Return result buffer address
28975 00001977 00                     _$P_Terminator:       db  0             ;AN000; Terminator code (ASCII)
28976 00001978 0000                   _$P_DBCSEV_OFF:       dw  0             ;AN000; Offset of DBCS EV
28977 0000197A 0000                   _$P_DBCSEV_SEG:       dw  0             ;AN000; Segment of DBCS EV
28978 0000197C 0000                   _$P_Flags:    dw  0             ;AN000; Parser internal flags
28979                                   %define _$P_Flags1 _$P_Flags            ;AN038; to reference first byte flags
28980                                   %define _$P_Flags2 _$P_Flags+1          ;AN038; to reference second byte flags only
28981
28982                                   ;in second byte of _$P_Flags, referenced as _$P_Flags2:
28983                                   _$P_equ               equ 01h         ;AN000; "=" packed in string buffet
28984                                   _$P_Neg               equ 02h         ;AN000; Negative value
28985                                   _$P_Time12    equ 04h         ;AN000; set when PM is specified
28986                                   _$P_Key_Cmp   equ 08h         ;AN000; set when keyword compare
28987                                   _$P_SW_Cmp    equ 10h         ;AN000; set when switch compare
28988                                   _$P_Extra     equ 20h         ;AN000; set when extra delimiter found
28989                                   _$P_SW                equ 40h         ;AN000; set when switch found (tm08)
28990                                   _$P_Signed    equ 80h         ;AN000; signed numeric specified
28991
28992                                   ;in first byte of _$P_Flags, referenced as _$P_Flags1:
28993                                   _$P_time12am          equ 01h         ;AN038; set when AM is specified on time
28994                                   _$P_TIME_AGAIN        equ 02h         ;AN039; SET WHEN READY TO RE-PARSE TIME
28995
28996 0000197E 0000                   _$P_SaveSI_Cmpx:   dw     0            ;AN000; save si for later use by complex
28997 00001980 0000                   _$P_KEYorSW_Ptr:   dw     0            ;AN000; points next to "=" or ":" code
28998 00001982 0000                   _$P_Save_EOB:         dw  0             ;AN000; save pointer to EOB
28999 00001984 0000                   _$P_Found_SYNONYM: dw     0            ;AN000; es:@ points to found synonym
```

```
29000
29001 00001986 00<rep 80h>            _$P_STRING_BUF:       times 128 db 0     ;AN000; Pick a operand from command line
29002                                 _$P_STRING_BUF_END equ    $              ;AN000;
29003
29004                                 ; 25/10/2022
29005                                 ; (MSDOS 5.0 IO.SYS, SYSINIT:16F8h)
29006
29007 00001A06 FF                     _$P_Char_CAP_Ptr:  db     0FFh           ;AN000; info id
29008 00001A07 0000                                      dw   0              ;AN000; offset of char case map table
29009 00001A09 0000                                      dw   0              ;AN000; segment of char case map table
29010                                 ; 25/10/2022
29011                                 ;IF CAPSW
29012                                 ;_$P_File_CAP_Ptr: db     0FFh           ;AN000; info id
29013                                 ;                  dw   0              ;AN000; offset of file case map table
29014                                 ;                  dw   0              ;AN000; segment of file case map table
29015                                 ;ENDIF
29016
29017                                 ; (tm06) IF FileSW                    ;AN000;(Check if file spec is supported)
29018                                 ;
29019
29020                                 ;M029
29021                                 ;!!!WARNING!!!
29022                                 ; In routine SYSPARSE (parse.asm), _$P_FileSp_Char is reinitialized using
29023                                 ;hardcoded strings. If the chars in the string are changed here, corresponding
29024                                 ;changes need to be made in SYSPARSE
29025
29026                                 ;IF FileSW+DrvSW                      ;AN000;(Check if file spec is supported)
29027
29028                                 ; 25/10/2022
29029                                 ; (MSDOS 5.0 IO.SYS, SYSINIT:16FDh)
29030
29031 00001A0B 5B5D7C3C3E2B3D3B22   _$P_FileSp_Char       db  '[]|<>+=;"'     ;AN000; delimitter of file spec
29032                                 _$P_FileSp_Len        equ $-_$P_FileSp_Char ;AN000;
29033
29034                                 ;ENDIF                                    ;AN000;(of FileSW)
29035
29036                                 ; delimiter parsing
29037                                 _$P_colon_period   equ    01h           ;AN032; check for colon & period
29038                                 _$P_period_only       equ 02h           ;AN032; check only for period
29039
29040                                 ;filespec error flag
29041 00001A14 00                     _$P_err_flag:         db  0             ;AN033; flag set if filespec parsing error
29042                                                                         ;AN033;  was detected.
29043                                 _$P_error_filespec equ    01h           ;AN033; mask to set flag
29044
29045
29046                                 ; PARSE.ASM - MSDOS 6.0 - 1991
29047                                 ; ============================================================================
29048                                 ; 27/03/2019 - Retro DOS v4.0
29049
29050                                 ;****************************************************************************
29051                                 ;
29052                                 ; SysParse;
29053                                 ;
29054                                 ;  Function : Parser Entry
29055                                 ;
29056                                 ;  Input: DS:SI -> command line
29057                                 ;     ES:DI -> parameter block
29058                                 ;     cs -> psdata.inc
29059                                 ;     CX = operand ordinal
29060                                 ;
29061                                 ;     Note:  ES is the segment containing all the control blocks defined
29062                                 ;            by the caller, except for the DOS COMMAND line parms, which
29063                                 ;            is in DS.
29064                                 ;
29065                                 ;  Output: CY = 1   error of caller, means invalid parameter block or
29066                                 ;               invalid value list. But this parser does NOT implement
29067                                 ;               this feature. Therefore CY always zero.
29068                                 ;
29069                                 ;       CY = 0   AX = return code
29070                                 ;                BL = terminated delimiter code
29071                                 ;                CX = new operand ordinal
29072                                 ;                SI = set past scaned operand
29073                                 ;                DX = selected result buffer
29074                                 ;
29075                                 ; Use:      _$P_Skip_Delim, _$P_Chk_EOL, _$P_Chk_Delim, _$P_Chk_DBCS
29076                                 ;   _$P_Chk_Swtch, _$P_Chk_Pos_Control, _$P_Chk_Key_Control
29077                                 ;   _$P_Chk_Sw_Control, _$P_Fill_Result
29078                                 ;
29079                                 ; Vars: _$P_Ordinal(RW), _$P_RC(RW), _$P_SI_Save(RW), _$P_DX(R), _$P_Terminator(R)
29080                                 ;   _$P_SaveSI_Cmpx(W), _$P_Flags(RW), _$P_Found_SYNONYM(R), _$P_Save_EOB(W)
29081                                 ;
29082                                 ;-------- Modification History ----------------------------------------
29083                                 ;
29084                                 ;  4/04/87 : Created by K. K,
29085                                 ;  4/28/87 : _$P_Val_YH assemble error (tm01)
29086                                 ;       : JMP SHORT assemble error (tm02)
29087                                 ;  5/14/87 : Someone doesn't want to include psdata (tm03)
29088                                 ;  6/12/87 : _$P_Bridge is missing when TimeSw equ 0 and (CmpxSw equ 1 or
29089                                 ;         DateSW equ 1)           (tm04)
29090                                 ;  6/12/87 : _$P_SorD_Quote is missing when QusSw equ 0 and CmpxSW equ 1
29091                                 ;                                 (tm05) in PSDATA.INC
29092                                 ;  6/12/87 : _$P_FileSp_Char and _$P_FileSP_Len are missing
29093                                 ;         when FileSW equ 0 and DrvSW equ 1 (tm06) in PSDATA.INC
29094                                 ;  6/18/87 : $VAL1 and $VAL3, $VAL2 and $VAL3 can be used in the same
29095                                 ;         value-list block        (tm07)
29096                                 ;  6/20/87 : Add _$P_SW to check if there's an omiting parameter after
29097                                 ;         switch (keyword) or not. If there is, backup si for next call
29098                                 ;         (tm08)
29099                                 ;  6/24/87 : Complex Item checking does not work correctly when CmpSW equ 1
29100                                 ;         and DateSW equ 0 and TimeSW equ 0 (tm09)
29101                                 ;  6/24/87 : New function flag _$P_colon_is_not_necessary for switch
29102                                 ;         /+15 and /+:15 are allowed for user (tm10)
29103                                 ;  6/29/87 : ECS call changes DS register but it causes the address problem
29104                                 ;         in user's routines. _$P_Chk_DBCS (tm11)
29105                                 ;  7/10/87 : Switch with no_match flag (0x0000H) does not work correctly
29106                                 ;                                 (tm12)
29107                                 ;  7/10/87 : Invalid switch/keyword does not work correctly
29108                                 ;                                 (tm13)
29109                                 ;  7/10/87 : Drive_only breaks 3 bytes after the result buffer
29110                                 ;                                 (tm14)
29111                                 ;  7/12/87 : Too_Many_Operands sets DX=0 as the PARSE result
29112                                 ;                                 (tm15)
29113                                 ;  7/24/87 : Negative lower bound on numeric ranges cause trouble
29114                                 ;  7/24/87 : Quoted strings being returned with quotes.
29115
29116                                 ;  7/28/87 : Kerry S (;AN018;)
29117                                 ;         Non optional value on switch (match flags<>0 and <>1) not flagged
29118                                 ;         as an error when missing.       Solution: return error 2.  Modules
29119                                 ;         affected: _$P_Chk_SW_Control.
29120
29121                                 ;  7/29/87 : Kerry S (;AN019;)
29122                                 ;         Now allow the optional bit in match flags for switches.  This
29123                                 ;         allows the switch to be encountered with a value or without a
```

```
29124                           ;       value and no error is returned.
29125                           ;
29126
29127                           ;  8/28/87 : Ed K, Kerry S (;AN020;)
29128                           ;  9/14/87   In PROC _$P_Get_DecNum, when checking for field separators
29129                           ;       within a date response, instead of checking just for the one
29130                           ;       character defined by the COUNTRY DEPENDENT INFO, check for
29131                           ;       all three chars, "-", "/", and ".". Change _$P_Chk_Switch to allow
29132                           ;       slashes in date strings when DateSw (assembler switch) is set.
29133
29134                           ;  9/1/87  : Kerry S (;AN021)
29135                           ;       In PROC _$P_String_Comp, when comparing the switch or keyword on
29136                           ;       the command line with the string in the control block the
29137                           ;       comparing was stopping at a colon (switch) or equal (keyword)
29138                           ;       on the command line and assuming a match.      This allowed a shorter
29139                           ;       string on the command line than in the synonym list in the control
29140                           ;       block.  I put in a test for a null in the control block so the
29141                           ;       string in the control block must be the same length as the string
29142                           ;       preceeding the colon or equal on the command line.
29143
29144                           ;  8/28/87 : Kerry S (;AN022;)
29145                           ;       All references to data in PSDATA.INC had CS overrides.  This caused
29146                           ;       problems for people who included it themselves in a segment other
29147                           ;       than CS.  Added switch to allow including PSDATA.INC in any
29148                           ;       segment.
29149
29150                           ;  9/16/87 : Ed K (;AN023;) PTM1040
29151                           ;       in _$P_set_cdi PROC, it assumes CS points to psdata. Change Push CS
29152                           ;       into PUSH cs.  In _$P_Get_DecNum PROC, fix AN020
29153                           ;       forced both TIME and DATE to use the delims, "-","/",".".
29154                           ;       Created FLag, in _$P_time_Format PROC, to request the delim in
29155                           ;       BL be used if TIME is being parsed.
29156
29157                           ;  9/24/87 : Ed K
29158                           ;       Removed the include to STRUC.INC.      Replaced the STRUC macro
29159                           ;       invocations with their normally expanded code; made comments
29160                           ;       out of the STRUC macro invocation statements to maintain readability.
29161
29162                           ;  9/24/87 : Ed K (;AN024;) PTM1222
29163                           ;       When no CONTROL for a keyword found, tried to fill in RESULT
29164                           ;       pointed to by non-existant CONTROL.
29165
29166                           ; 10/15/87 : Ed K (;AN025;) PTM1672
29167                           ;       A quoted text string can be framed only by double quote.  Remove
29168                           ;       support to frame quoted text string with single quote.
29169                           ;       (apostrophe) _$P_SorD_Quote is removed from PSDATA.INC.
29170                           ;       _$P_SQuote EQU also removed from PSDATA.INC.  Any references to
29171                           ;       single quote in PROC prologues are left as is for history reasons.
29172
29173                           ;       This fixes another bug, not mentioned in p1672, in that two
29174                           ;       quote chars within a quoted string is supposed to be reported as
29175                           ;       one quote character, but is reported as two quotes.  This changed
29176                           ;       two instructions in PROC _$P_Quoted_Str.
29177
29178                           ;       Also fixed are several JMP that caused a NOP, these changed to
29179                           ;       have the SHORT operator to avoid the unneeded NOP.
29180
29181                           ;       The code and PSDATA.INC have been aligned for ease of reading.
29182
29183                           ; 10/26/87 : Ed K (;AN026;) PTM2041, DATE within SWITCH, BX reference to
29184                           ;       psdata buffer should have cs.
29185
29186                           ; 10/27/87 : Ed K (;AN027;) PTM2042 comma between keywords implies
29187                           ;       positional missing.
29188
29189                           ; 11/06/87 : Ed K (;AN028;) PTM 2315 Parser should not use line feed
29190                           ;       as a line delimiter, should use carriage return.
29191                           ;       Define switch: LFEOLSW, if on, accept LF as end of line char.
29192
29193                           ; 11/11/87 : Ed K (;AN029;) PTM 1651 GET RID OF WHITESPACE AROUND "=".
29194
29195                           ; 11/18/87 : Ed K (;AN030;) PTM 2551 If filename is just "", then
29196                           ;       endless loop since SI is returned still pointing to start
29197                           ;       of that parm.
29198
29199                           ; 11/19/87 : Ed K (;AN031;) PTM 2585 date & time getting bad values.
29200                           ;       Vector to returned string has CS instead of cs, but
29201                           ;       when tried to fix it on previous version, changed similar
29202                           ;       but wrong place.
29203
29204                           ; 12/09/87 : Bill L (;AN032;) PTM 2772 colon and period are now valid
29205                           ;       delimiters between hours, minutes, seconds for time. And period
29206                           ;       and comma are valid delimiters between seconds and 100th second.
29207
29208                           ; 12/14/87 : Bill L (;AN033;) PTM 2722 if illegal delimiter characters
29209                           ;       in a filespec, then flag an error.
29210
29211                           ; 12/22/87 : Bill L (;AN034;)        All local data to parser is now
29212                           ;       indexed off of the cs equate instead of the DS register.
29213                           ;       Using this method, DS can point to the segment of PSP or to psdata
29214                           ;  -->          local parser data. Why were some references to local data changed
29215                           ;       to do this before, but not all ?????
29216
29217                           ; 02/02/88 : Ed K (;AC035;) INSPECT utility, suggests optimizations.
29218
29219                           ; 02/05/88 : Ed K (;AN036;) P3372-UPPERCASE TRANSLATION, cs HOSED.
29220                           ;
29221                           ; 02/08/88 : Ed K (;AN037;) P3410-AVOID POP OF CS, CHECK BASESW FIRST.
29222
29223                           ; 02/19/88 : Ed K (;AN038;) p3524 above noon and "am" should be error
29224
29225                           ; 02/23/88 : Ed K (;AN039;) p3518 accept "comma" and "period" as decimal
29226                           ;       separator in TIME before hundredths field.
29227                           ;
29228                           ; 08/09/90 : SA    M005    Prevented parser from recognizing '=' signs within
29229                           ;                  strings as keywords.
29230                           ;
29231                           ;*****************************************************************************
29232
29233                           ;IF FarSW                          ;AN000;(Check if need far return)
29234                           ;SysParse proc far                 ;AN000;
29235                           ;ELSE                                      ;AN000;
29236                           ;SysParse proc near                        ;AN000;
29237                           ;ENDIF                                     ;AN000;(of FarSW)
29238
29239                           ; 27/03/2019 - Retro DOS v4.0
29240                           ; (MSDOS 6.21 IO.SYS - SYSINIT:1842h)
29241
29242                           ; 25/10/2022 - Retro DOS v4.0
29243                           ; (MSDOS 5.0 IO.SYS - SYSINIT:1707h)
29244
29245                           ; 06/09/2023 - Retro DOS v4.2 IO.SYS Optimization (& Retro DOS v5.0 pre-work)
29246                           ; (PCDOS 7.1 IBMBIO.COM - SYSINIT:1D08h)
29247
```

```
                                SysParse:
                                    ; 06/09/2023 - Retro DOS v4.2 IO.SYS (Optimization)
                                    ; dx = 0
00001A15 1E                         push    ds ; *!*
00001A16 0E                         push    cs
00001A17 1F                         pop     ds

                                    ;mov    word [cs:_$P_Flags],0 ;AC034; Clear all internal flags
                                    ;cld                          ;AN000; confirm forward direction
                                    ;mov    word [cs:_$P_ORDINAL],cx ;AC034; save operand ordinal
                                    ;mov    word [cs:_$P_RC],_$P_No_Error ;AC034; Assume no error
                                    ;mov    word [cs:_$P_Found_SYNONYM],0 ;AC034; initalize synonym pointer
                                    ;
                                    ;mov    word [cs:_$P_DX],0    ;AC034; (tm15)

                                    ; 06/09/2023
00001A18 8916[7C19]                 mov     [_$P_Flags],dx ; 0    ;AC034; Clear all internal flags
00001A1C FC                         cld                           ;AN000; confirm forward direction
00001A1D 890E[6F19]                 mov     [_$P_ORDINAL],cx      ;AC034; save operand ordinal
00001A21 8916[7119]                 mov     [_$P_RC],dx ; $P_No_Error ;AC034; Assume no error
00001A25 8916[8419]                 mov     [_$P_Found_SYNONYM],dx; 0 ;AC034; initalize synonym pointer
00001A29 8916[7519]                 mov     [_$P_DX],dx ; 0               ;AC034; (tm15)

                                ;M029 -- Begin changes
                                ; The table of special chars _$P_FileSp_Char should be initialized on every
                                ;entry to SysParse. This is in the non-checksum region and any program that
                                ;corrupts this table but does not corrupt the checksum region will leave
                                ;command.com parsing in an inconsistent state.
                                ; NB: The special characters string has been hardcoded here. If any change
                                ;is made to it in psdata.inc, a corresponding change needs to be made here.

                                ;IF FileSW + DrvSW
                                    ; 14/04/2024 (NASM syntax BugFix) .. '][' (MASM) -> '[]' (NASM)

                                    ;mov    word [cs:_$P_FileSp_Char], '[]'
                                    ;mov    word [cs:_$P_FileSp_Char+2], '|<'
                                    ;mov    word [cs:_$P_FileSp_Char+4], '>+'
                                    ;mov    word [cs:_$P_FileSp_Char+6], '=;'

                                    ; 14/04/2024
                                    ; 06/09/2023
00001A2D C706[0B1A]5B5D             mov     word [_$P_FileSp_Char], '[]'  ; mov word [_$P_FileSp_Char],5D5Bh
00001A33 C706[0D1A]7C3C             mov     word [_$P_FileSp_Char+2], '|<'       ; 3C7Ch
00001A39 C706[0F1A]3E2B             mov     word [_$P_FileSp_Char+4], '>+'       ; 2B3Eh
00001A3F C706[111A]3D3B             mov     word [_$P_FileSp_Char+6], '=;'       ; 3B3Dh
                                ;ENDIF
                                    ; 06/09/2023
00001A45 1F                         pop     ds ; *!*

                                ;M029 -- End of changes

00001A46 E87806                     call    _$P_Skip_Delim        ;AN000; Move si to 1st non white space
00001A49 7313                       jnc     short _$P_Start               ;AN000; If EOL is not encountered, do parse
                                ;--------------------------- End of Line
00001A4B B8FFFF                     mov     ax,_$P_RC_EOL         ;AN000; set exit code to -1
00001A4E 53                         push    bx                    ;AN000;
                                    ;mov    bx,[es:di+_$P_PARMS_Blk.PARMSX_Address]
                                                                  ;AN000; Get the PARMSX address to
00001A4F 268B1D                     mov     bx,[es:di]
                                    ;cmp    cl,[es:bx+_$P_PARMSX_Blk.MinP]
                                                                  ;AN000; check ORDINAL to see if the minimum
00001A52 263A0F                     cmp     cl,[es:bx]
00001A55 7303                       jae     short _$P_Fin         ;AN000; positional found.

00001A57 B80200                     mov     ax,_$P_Op_Missing     ;AN000; If no, set exit code to missing operand
                                _$P_Fin:                          ;AN000;
00001A5A 5B                         pop     bx                    ;AN000;
00001A5B E90A01                     jmp     _$P_Single_Exit               ;AN000; return to the caller
                                ;---------------------------
                                _$P_Start:                        ;AN000;
00001A5E 2E8936[7E19]               mov     [cs:_$P_SaveSI_Cmpx],si ;AN000;AC034; save ptr to command line for later use by complex,
00001A63 53                         push    bx                    ;AN000; quoted string or file spec.
00001A64 57                         push    di                    ;AN000;
00001A65 55                         push    bp                    ;AN000;
                                    ;;lea   bx,[cs:_$P_STRING_BUF] ;AC034; set buffer to copy from command string
                                    ; 02/11/2022
                                    ;lea    bx,[_$P_STRING_BUF]
                                    ; 07/09/2023
00001A66 BB[8619]                   mov     bx,_$P_STRING_BUF
00001A69 2EF606[7D19]20             test    byte [cs:_$P_Flags2],_$P_Extra ;AC034; 3/9 extra delimiter encountered ?
00001A6F 7543                       jnz     short _$P_Pack_End    ;AN000; 3/9 if yes, no need to copy

                                _$P_Pack_Loop:                            ;AN000;
00001A71 AC                         lodsb                         ;AN000; Pick a operand from buffer
00001A72 E8F106                     call    _$P_Chk_Switch        ;AN000; Check switch character
00001A75 723C                       jc      short _$P_Pack_End_BY_EOL ;AN020; if carry set found delimiter type slash, need backup si, else
continue

00001A77 E86906                     call    _$P_Chk_EOL           ;AN000; Check EOL character
00001A7A 7437                       je      short _$P_Pack_End_BY_EOL ;AN000; need backup si

00001A7C E89906                     call    _$P_Chk_Delim         ;AN000; Check delimiter
00001A7F 7518                       jne     short _$P_PL01                ;AN000; If no, process next byte

00001A81 2EF606[7D19]20             test    byte [cs:_$P_Flags2],_$P_Extra ;AC034; 3/9 If yes and white spec,
                                ; (tm08)jne short _$P_Pack_End    ;AN000; 3/9 then
00001A87 7505                       jnz     short _$P_Pack_End_backup_si ;AN000; (tm08)

00001A89 E83506                     call    _$P_Skip_Delim        ;AN000; skip subsequent white space,too
00001A8C EB26                       jmp     short _$P_Pack_End    ;AN000; finish copy by placing NUL at end

                                _$P_Pack_End_backup_si:                   ;AN000; (tm08)
00001A8E 2EF606[7D19]41             test    byte [cs:_$P_Flags2],_$P_SW+_$P_equ ;AN000;AC034;  (tm08)
00001A94 741E                       jz      short _$P_Pack_End    ;AN000; (tm08)

00001A96 4E                         dec     si                    ;AN000; (tm08)
00001A97 EB1B                       jmp     short _$P_Pack_End    ;AN025; (tm08)

                                _$P_PL01:                         ;AN000;
00001A99 2E8807                     mov     [cs:bx],al            ;AN000; move byte to STRING_BUF
00001A9C 3C3D                       cmp     al,_$P_Keyword  ;'='  ;AN000; if it is equal character,
00001A9E 7506                       jne     short _$P_PL00                ;AN000; then

00001AA0 2E800E[7D19]01             or      byte [cs:_$P_Flags2],_$P_equ ;AC034; remember it in flag
                                _$P_PL00:                         ;AN000;
00001AA6 43                         inc     bx                    ;AN000; ready to see next byte
00001AA7 E8D506                     call    _$P_Chk_DBCS          ;AN000; was it 1st byte of DBCS ?
00001AAA 73C5                       jnc     short _$P_Pack_Loop   ;AN000; if no, process to next byte

00001AAC AC                         lodsb                         ;AN000; if yes, store
00001AAD 2E8807                     mov     [cs:bx],al            ;AN000;    2nd byte of DBCS
00001AB0 43                         inc     bx                    ;AN000; update pointer
00001AB1 EBBE                       jmp     short _$P_Pack_Loop   ;AN000; process to next byte
```

```
29371                                   _$P_Pack_End_BY_EOL:                    ;AN000;
29372 00001AB3 4E                           dec     si                      ;AN000; backup si pointer
29373                                   _$P_Pack_End:                           ;AN000;
29374 00001AB4 2E8936[7319]                 mov     [cs:_$P_SI_Save],si     ;AC034; save next pointer, SI
29375                                       ; 07/09/2023
29376                                       ;mov    byte [cs:bx],_$P_NULL ;AN000; put NULL at the end
29377 00001AB9 30E4                         xor     ah,ah ; 0 ; *
29378 00001ABB 2E8827                       mov     [cs:bx],ah ; _$P_NULL ;AN000; put NULL at the end
29379                                       ;
29380 00001ABE 2E891E[8219]                 mov     [cs:_$P_Save_EOB],bx    ;AC034; 3/17/87 keep the address for later use of complex
29381                                       ;mov    bx,[es:di+_$P_PARMS_Blk.PARMSX_Address] ;AN000; get PARMSX address
29382 00001AC3 268B1D                       mov     bx,[es:di]
29383                                       ;;lea   si,[cs:_$P_STRING_BUF];AC034;
29384                                       ; 02/11/2022
29385                                       ;lea    si,[_$P_STRING_BUF]
29386                                       ; 07/09/2023
29387 00001AC6 BE[8619]                     mov     si,_$P_STRING_BUF
29388 00001AC9 2E803C2F                     cmp     byte [cs:si],_$P_Switch ;AN000; the operand begins w/ switch char ?
29389 00001ACD 7440                         je      short _$P_SW_Manager  ;AN000; if yes, process as switch
29390
29391 00001ACF 2E803C22                     cmp     byte [cs:si],_$P_DQuote       ;M005;is it a string?
29392 00001AD3 7408                         je      short _$P_Positional_Manager ;M005;if so, process as one!
29393
29394 00001AD5 2EF606[7D19]01               test    byte [cs:_$P_Flags2],_$P_equ ;AC034; the operand includes equal char ?
29395 00001ADB 7552                         jnz     short _$P_Key_Manager ;AN000; if yes, process as keyword
29396
29397                                   _$P_Positional_Manager:                 ;AN000; else process as positional
29398 00001ADD 268A4701                     mov     al,[es:bx+_$P_PARMSX_Blk.MaxP] ;AN000; get maxp
29399                                       ; 07/09/2023
29400                                       ;xor    ah,ah                   ;AN000; ax = maxp
29401 00001AE1 2E3906[6F19]                 cmp     [cs:_$P_ORDINAL],ax     ;AC034; too many positional ?
29402 00001AE6 7312                         jae     short _$P_Too_Many_Error ;AN000; if yes, set exit code to too many
29403
29404 00001AE8 2EA1[6F19]                   mov     ax,[cs:_$P_ORDINAL]     ;AC034; see what the current ordinal
29405 00001AEC D1E0                         shl     ax,1                    ;AN000; ax = ax*2
29406 00001AEE 43                           inc     bx                      ;AC035; add '2' to
29407 00001AEF 43                           inc     bx                      ;AC035;  BX reg
29408                                                                       ;AN000; now bx points to 1st CONTROL
29409 00001AF0 01C3                         add     bx,ax                   ;AN000; now bx points to specified CONTROL address
29410 00001AF2 268B1F                       mov     bx,[es:bx]              ;AN000; now bx points to specified CONTROL itself
29411 00001AF5 E87200                       call    _$P_Chk_Pos_Control     ;AN000; Do process for positional
29412 00001AF8 EB53                         jmp     short _$P_Return_to_Caller ;AN000; and return to the caller
29413
29414                                   _$P_Too_Many_Error:                     ;AN000;
29415 00001AFA 2EC706[7119]0100             mov     word [cs:_$P_RC],_$P_Too_Many ;AC034; set exit code
29416 00001B01 EB4A                         jmp     short _$P_Return_to_Caller ;AN000; and return to the caller
29417
29418                                       ; 07/09/2023 - Retro DOSD v4.2 IO.SYS (Optimization)
29419                                       ; (PCDOS 7.1 IBMBIO.COM - SYSINIT:1E06h)
29420                                   get_maxp:
29421                                       ;mov    al,[es:bx+1]
29422 00001B03 268A4701                     mov     al,[es:bx+_$P_PARMSX_Blk.MaxP] ;AN000; get maxp
29423                                       ; 07/09/2023
29424                                       ; ah=0 ; *
29425                                       ;xor    ah,ah ; 0               ;AN000; ax = maxp
29426 00001B07 30ED                         xor     ch,ch ; **
29427 00001B09 40                           inc     ax                      ;AN000;
29428 00001B0A D1E0                         shl     ax,1                    ;AN000; ax = (ax+1)*2
29429 00001B0C 01C3                         add     bx,ax                   ;AN000; now bx points to maxs
29430 00001B0E C3                           retn
29431
29432                                   _$P_SW_Manager:                         ;AN000;
29433                                       ; 07/09/2023
29434                                       ;mov    al,[es:bx+_$P_PARMSX_Blk.MaxP] ;AN000; get maxp
29435                                       ;xor    ah,ah                   ;AN000; ax = maxp
29436                                       ;inc    ax                      ;AN000;
29437                                       ;shl    ax,1                    ;AN000; ax = (ax+1)*2
29438                                       ;add    bx,ax                   ;AN000; now bx points to maxs
29439 00001B0F E8F1FF                       call    get_maxp ; 07/09/2023
29440
29441 00001B12 268A0F                       mov     cl,[es:bx]              ;AN000;
29442                                       ; 07/09/2023
29443                                       ;xor    ch,ch ; ** (ch=0)       ;AN000; cx = maxs
29444                                       ;or     cx,cx                   ;AN000; at least one switch ?
29445                                       ;jz     short _$P_SW_Not_Found        ;AN000;
29446                                       ; 07/07/2023
29447 00001B15 E30F                         jcxz    _$P_SW_Not_Found        ; no
29448
29449 00001B17 43                           inc     bx                      ;AN000; now bx points to 1st CONTROL address
29450
29451                                   _$P_SW_Mgr_Loop:                        ;AN000;
29452 00001B18 53                           push    bx                      ;AN000;
29453 00001B19 268B1F                       mov     bx,[es:bx]              ;AN000; bx points to Switch CONTROL itself
29454 00001B1C E8A900                       call    _$P_Chk_SW_Control      ;AN000; do process for switch
29455 00001B1F 5B                           pop     bx                      ;AN000;
29456 00001B20 732B                         jnc     short _$P_Return_to_Caller ;AN000; if the CONTROL is for the switch, exit
29457
29458 00001B22 43                           inc     bx                      ;AC035; add '2' to
29459 00001B23 43                           inc     bx                      ;AC035;  BX reg
29460                                                                       ;AN000; else bx points to the next CONTROL
29461 00001B24 E2F2                         loop    _$P_SW_Mgr_Loop               ;AN000; and loop
29462
29463                                   _$P_SW_Not_Found:                       ;AN000;
29464 00001B26 2EC706[7119]0300             mov     word [cs:_$P_RC],_$P_Not_In_SW ;AC034; here no CONTROL for the switch has
29465 00001B2D EB1E                         jmp     short _$P_Return_to_Caller ;AN000; not been found, means error.
29466
29467                                   _$P_Key_Manager:                        ;AN000;
29468                                       ; 07/09/2023
29469                                       ;mov    al,[es:bx+_$P_PARMSX_Blk.MaxP] ;AN000; get maxp
29470                                       ;xor    ah,ah                   ;AN000; ax = maxp
29471                                       ;inc    ax                      ;AN000;
29472                                       ;shl    ax,1                    ;AN000; ax = (ax+1)*2
29473                                       ;add    bx,ax                   ;AN000; now bx points to maxs
29474 00001B2F E8D1FF                       call    get_maxp ; 07/09/2023
29475
29476 00001B32 268A07                       mov     al,[es:bx]              ;AN000;
29477 00001B35 30E4                         xor     ah,ah ; 0               ;AN000; ax = maxs
29478 00001B37 D1E0                         shl     ax,1                    ;AN000;
29479 00001B39 40                           inc     ax                      ;AN000; ax = ax*2+1
29480 00001B3A 01C3                         add     bx,ax                   ;AN000; now bx points to maxk
29481 00001B3C 268A0F                       mov     cl,[es:bx]              ;AN000;
29482                                       ; 07/09/2023
29483                                       ;xor    ch,ch ; ** (ch=0)       ;AN000; cx = maxk
29484                                       ;or     cx,cx                   ;AN000; at least one keyword ?
29485                                       ;jz     short _$P_Key_Not_Found       ;AN000;
29486                                       ; 07/07/2023
29487 00001B3F E305                         jcxz    _$P_Key_Not_Found       ; no
29488
29489 00001B41 43                           inc     bx                      ;AN000; now bx points to 1st CONTROL
29490
29491                                   _$P_Key_Mgr_Loop:                       ;AN000;
29492                                       ; 07/09/2023
29493                                       ; ('_$P_Chk_Key_Control' contains only 'stc' instruction)
29494                                       ; (always returns with cf=1)
```

```
                                    ;push   bx                      ;AN000;
                                    ;mov    bx,[es:bx]              ;AN000; bx points to keyword CONTROL itself
                                    ;call   _$P_Chk_Key_Control     ;AN000; do process for keyword
                                    ;pop    bx                      ;AN000;
                                    ;jnc    short _$P_Return_to_Caller ;AN000; if the CONTROL is for the keyword, exit
                                    ; 07/09/2023
                                    ; cf=1 (after 'call _$P_Chk_Key_Control')

00001B42 43                         inc     bx                      ;AC035; add '2' to
00001B43 43                         inc     bx                      ;AC035;  BX reg
                                                                    ;AN000; else bx points to the next CONTROL
00001B44 E2FC                       loop    _$P_Key_Mgr_Loop        ;AN000; and loop

                                _$P_Key_Not_Found:                  ;AN000;
00001B46 2EC706[7119]0400           mov     word [cs:_$P_RC],_$P_Not_In_Key ;AC034; here no CONTROL for the keyword has
                                _$P_Return_to_Caller:                   ;AN000;
00001B4D 5D                         pop     bp                      ;AN000;
00001B4E 5F                         pop     di                      ;AN000;
00001B4F 5B                         pop     bx                      ;AN000;
00001B50 2E8B0E[6F19]               mov     cx,[cs:_$P_ORDINAL]     ;AC034; return next ordinal
00001B55 2EA1[7119]                 mov     ax,[cs:_$P_RC]          ;AC034; return exit code
00001B59 2E8B36[7319]               mov     si,[cs:_$P_SI_Save]     ;AC034; return next operand pointer
00001B5E 2E8B16[7519]               mov     dx,[cs:_$P_DX]          ;AC034; return result buffer address
00001B63 2E8A1E[7719]               mov     bl,[cs:_$P_Terminator]  ;AC034; return delimiter code found
                                _$P_Single_Exit:                    ;AN000;
00001B68 F8                         clc                             ;AN000;
00001B69 C3                         retn                            ;AN000;

                                ;******************************************************************************
                                ; _$P_Chk_Pos_Control
                                ;
                                ; Function: Parse CONTROL block for a positional
                                ;
                                ; Input:     ES:BX -> CONTROL block
                                ;         cs:SI -> _$P_STRING_BUF
                                ;
                                ; Output:    None
                                ;
                                ; Use:       _$P_Fill_Result, _$P_Check_Match_Flags
                                ;
                                ; Vars: _$P_Ordinal(W), _$P_RC(W)
                                ;******************************************************************************

                                _$P_Chk_Pos_Control:
00001B6A 50                         push    ax                      ;AN000;
                                    ;mov    ax,[es:bx+_$P_Control_Blk.Match_Flag] ;AN000;
00001B6B 268B07                     mov     ax,[es:bx]
                                    ; 12/12/2022
00001B6E A802                       test    al,_$P_Repeat
                                    ;test   ax,_$P_Repeat           ;AN000; repeat allowed ?
00001B70 7505                       jnz     short _$P_CPC00                 ;AN000; then do not increment ORDINAL

00001B72 2EFF06[6F19]               inc     word [cs:_$P_ORDINAL] ;AC034; update the ordinal
                                _$P_CPC00:                          ;AN000;
00001B77 2E803C00                   cmp     byte [cs:si],_$P_NULL ;AN000; no data ?
00001B7B 7517                       jne     short _$P_CPC01                 ;AN000;

                                    ; 12/12/2022
00001B7D A801                       test    al,_$P_Optional
                                    ;test   ax,_$P_Optional                 ;AN000; yes, then is it optional ?
00001B7F 7509                       jnz     short _$P_CPC02                 ;AN000;

00001B81 2EC706[7119]0200           mov     word [cs:_$P_RC],_$P_Op_Missing ;AC034; no, then error 3/17/87
00001B88 EB0D                       jmp     short _$P_CPC_Exit      ;AN000;

                                _$P_CPC02:                          ;AN000;
00001B8A 50                         push    ax                      ;AN000;
                                    ;mov    al,_$P_String           ;AN000; if it is optional return NULL
                                    ;mov    ah,_$P_No_Tag           ;AN000; no item tag indication
                                    ; 07/07/2023
00001B8B B803FF                     mov     ax,(_$P_No_Tag<<8)|_$P_String
00001B8E E89600                     call    _$P_Fill_Result                 ;AN000;
00001B91 58                         pop     ax                      ;AN000;
00001B92 EB03                       jmp     short _$P_CPC_Exit      ;AN000;

                                _$P_CPC01:                          ;AN000;
00001B94 E81101                     call    _$P_Check_Match_Flags ;AN000;
                                _$P_CPC_Exit:                               ;AN000;
00001B97 58                         pop     ax                      ;AN000;
00001B98 C3                         retn                            ;AN000;

                                ;******************************************************************************
                                ; _$P_Chk_Key_Control
                                ;
                                ; Function: Parse CONTROL block for a keyword
                                ;
                                ; Input:     ES:BX -> CONTROL block
                                ;         cs:SI -> _$P_STRING_BUF
                                ;
                                ; Output:    CY = 1 : not match
                                ;
                                ; Use:       _$P_Fill_Result, _$P_Search_KEYorSW, _$P_Check_Match_Flags
                                ;
                                ; Vars: _$P_RC(W), _$P_SaveSI_Cmpx(W), _$P_KEYorSW_Ptr(R), _$P_Flags(W)
                                ;******************************************************************************

                                ; 07/09/2023
                                ;_$P_Chk_Key_Control:
                                ;   stc                             ;AN000; this logic works when the KeySW
                                ;   retn                            ;AN000; is reset.

                                ;******************************************************************************
                                ; _$P_Search_KEYorSW:
                                ;
                                ; Function: Seach specified keyword or switch from CONTROL
                                ;
                                ; Input:     ES:BX -> CONTROL block
                                ;         cs:SI -> _$P_STRING_BUF
                                ;
                                ; Output:    CY = 1 : not match
                                ;
                                ; Use:       _$P_String_Comp, _$P_MoveBP_NUL, _$P_Found_SYNONYM
                                ;******************************************************************************

                                    ; 25/10/2022 - Retro DOS v4.0
                                    ; (MSDOS 5.0 IO.SYS - SYSINIT:18B6h)

                                _$P_Search_KEYorSW:                         ;AN000;
00001B99 55                         push    bp                      ;AN000;
00001B9A 51                         push    cx                      ;AN000;
00001B9B 268A4F08                   mov     cl,[es:bx+_$P_Control_Blk.nid] ;AN000; Get synonym count
00001B9F 30ED                       xor     ch,ch                   ;AN000; and set it to cx
                                    ;or     cx,cx                   ;AN000; No synonyms specified ?
                                    ;jz     short _$P_KEYorSW_Not_Found ;AN000; then indicate not found by CY
```

```
                                    ; 07/07/2023
00001BA1 E30D                       jcxz    _$P_KEYorSW_Not_Found

                                    ;lea    bp,[es:bx+_$P_Control_Blk.KEYorSW] ;AN000; BP points to the 1st synonym
                                    ; 25/10/2022
00001BA3 8D6F09                     lea     bp,[bx+_$P_Control_Blk.KEYorSW]
                                    ;lea    bp,[bx+9]
                                _$P_KEYorSW_Loop:               ;AN000;
00001BA6 E8B503                     call    _$P_String_Comp             ;AN000; compare string in buffer w/ the synonym
00001BA9 7308                       jnc     short _$P_KEYorSW_Found     ;AN000; If match, set it to synonym pointer

00001BAB E80E00                     call    _$P_MoveBP_NUL      ;AN000; else, bp points to the next string
00001BAE E2F6                       loop    _$P_KEYorSW_Loop    ;AN000; loop nid times
                                _$P_KEYorSW_Not_Found:                  ;AN000;
00001BB0 F9                         stc                         ;AN000; indicate not found in synonym list
00001BB1 EB06                       jmp     short _$P_KEYorSW_Exit;AN000; and exit

                                _$P_KEYorSW_Found:              ;AN000;
00001BB3 2E892E[8419]               mov     [cs:_$P_Found_SYNONYM],bp ;AC034; set synonym pointer
00001BB8 F8                         clc                         ;AN000; indicate found
                                _$P_KEYorSW_Exit:               ;AN000;
00001BB9 59                         pop     cx                  ;AN000;
00001BBA 5D                         pop     bp                  ;AN000;
00001BBB C3                         retn                        ;AN000;

                                ;****************************************************************************
                                ; _$P_MoveBP_NUL
                                ;****************************************************************************

                                _$P_MoveBP_NUL:
                                _$P_MBP_Loop:                               ;AN000;
                                    ; 11/12/2022
00001BBC 26807E0000                 cmp     byte [es:bp],_$P_NULL ;AN000; Increment BP that points
                                    ; 25/10/2022 (MSDOS 5.0 IO.SYS compatibility)
                                    ; (SYSINIT:18DBh)
                                    ;cmp     byte [es:bp+0],0
00001BC1 7403                       je      short _$P_MBP_Exit    ;AN000; to the synomym list

00001BC3 45                         inc     bp                  ;AN000; until
00001BC4 EBF6                       jmp     short _$P_MBP_Loop    ;AN000; NULL encountered.

                                _$P_MBP_Exit:                               ;AN000;
00001BC6 45                         inc     bp                  ;AN000; bp points to next to NULL
00001BC7 C3                         retn                        ;AN000;

                                ;****************************************************************************
                                ; _$P_Chk_SW_Control
                                ;
                                ; Function: Parse CONTROL block for a switch
                                ;
                                ; Input:     ES:BX -> CONTROL block
                                ;        cs:SI -> _$P_STRING_BUF
                                ;
                                ; Output:    CY = 1 : not match
                                ;
                                ; Use:       _$P_Fill_Result, _$P_Search_KEYorSW, _$P_Check_Match_Flags
                                ;
                                ; Vars:  _$P_SaveSI_Cmpx(W), _$P_KEYorSW_Ptr(R), _$P_Flags(W)
                                ;****************************************************************************

                                _$P_Chk_SW_Control:

                                ;IF SwSW                            ;AN000;(Check if switch is supported)
                                    ;or     byte [cs:_$P_Flags+1],10h
00001BC8 2E800E[7D19]10             or      byte [cs:_$P_Flags2],_$P_SW_Cmp ;AC034; Indicate switch for later string comparison
00001BCE E8C8FF                     call    _$P_Search_KEYorSW      ;AN000; Search the switch in the CONTROL block
00001BD1 7248                       jc      short _$P_Chk_SW_Err0 ;AN000; not found, then try next CONTROL

                                    ;and    [cs:_$P_Flags+],0EFh
00001BD3 2E8026[7D19]EF             and     byte [cs:_$P_Flags2],0FFh-_$P_SW_Cmp
                                                                    ;AC034; reset the indicator previously set
00001BD9 50                         push    ax                      ;AN000;             /switch:
00001BDA 2EA1[8019]                 mov     ax,[cs:_$P_KEYorSW_Ptr] ;AC034;             ^       ^
00001BDE 29F0                       sub     ax,si                   ;AN000;  SI    KEYorSW
00001BE0 2E0106[7E19]               add     [cs:_$P_SaveSI_Cmpx],ax     ;AC034; update for complex list
00001BE5 58                         pop     ax                      ;AN000;

00001BE6 2E8B36[8019]               mov     si,[cs:_$P_KEYorSW_Ptr] ;AC034; set si at the end or colon
00001BEB 2E803C00                   cmp     byte [cs:si],_$P_NULL ;AN000; any data after colon
00001BEF 7525                       jne     short _$P_CSW00             ;AN000; if yes, process match flags

00001BF1 2E807CFF3A                 cmp     byte [cs:si-1],_$P_Colon ;AN000; if no, the switch terminated by colon ?
00001BF6 7509                       jne     short _$P_Chk_if_data_required ;AN000; if yes,

00001BF8 2EC706[7119]0900           mov     word [cs:_$P_RC],_$P_Syntax ;AC034; return syntax error
00001BFF EB1C                       jmp     short _$P_Chk_SW_Exit ;AN000;

                                _$P_Chk_if_data_required:       ;AN018; no data, no colon
                                    ;cmp    word [es:bx+_$P_Control_Blk.Match_Flag],0
00001C01 26833F00                   cmp     word [es:bx],0      ;AN018; should have data? zero match flag means switch followed by nothing is
OK
00001C05 7416                       je      short _$P_Chk_SW_Exit ;AN018; match flags not zero so should have something if optional bit is not
on

                                    ;test   word [es:bx+_$P_Control_Blk.Match_Flag],_$P_Optional
                                    ; 02/11/2022 (MSDOS 5.0 IO.SYS SYINIT compatibility)
                                    ;test   word [es:bx],1
                                    ; 12/12/2022
                                    ;test   word [es:bx],_$P_Optional ;AN019; see if no value is valid
00001C07 26F60701                   test    byte [es:bx],_$P_Optional
00001C0B 7510                       jnz     short _$P_Chk_SW_Exit ;AN019; if so, then leave, else yell

00001C0D 2EC706[7119]0200           mov     word [cs:_$P_RC],_$P_Op_Missing ;AC034; return required operand missing
00001C14 EB07                       jmp     short _$P_Chk_SW_Exit ;AN018;

                                _$P_CSW00:                      ;AN000;
00001C16 E88F00                     call    _$P_Check_Match_Flags ;AN000; process match flag
00001C19 F8                         clc                         ;AN000; indicate match
                                    ;jmp    short _$P_Chk_SW_Single_Exit ;AN000;
                                    ; 12/12/2022
00001C1A C3                         retn

                                _$P_Chk_SW_Err0:                ;AN000;
00001C1B F9                         stc                         ;AN000; not found in switch synonym list
                                    ;jmp    short _$P_Chk_SW_Single_Exit ;AN000;
                                    ; 12/12/2022
00001C1C C3                         retn

                                _$P_Chk_SW_Exit:                ;AN000;
00001C1D 50                         push    ax                  ;AN000;
                                    ;mov    al,_$P_String       ;AN000;
                                    ;mov    ah,_$P_No_Tag       ;AN000;
                                    ; 07/07/2023
00001C1E B803FF                     mov     ax,(_$P_No_Tag<<8)|_$P_String
```

```
29741 00001C21 E80300                   call    _$P_Fill_Result             ;AN000; set result buffer
29742 00001C24 58                       pop     ax                  ;AN000;
29743 00001C25 F8                       clc                         ;AN000;
29744                                 _$P_Chk_SW_Single_Exit:                 ;AN000;
29745 00001C26 C3                       retn                        ;AN000;
29746                                 ;ELSE                                   ;AN000;(of IF SwSW)
29747                                 ;   stc                         ;AN000; this logic works when the SwSW
29748                                 ;   retn                        ;AN000; is reset.
29749
29750                                 ;******************************************************************************
29751                                 ; _$P_Fill_Result
29752                                 ;
29753                                 ; Function: Fill the result buffer
29754                                 ;
29755                                 ; Input:    AH = Item tag
29756                                 ;       AL = type
29757                                 ;            AL = 1: CX,DX has 32bit number (CX = high)
29758                                 ;            AL = 2: DX has index(offset) into value list
29759                                 ;            AL = 6: DL has driver # (1-A, 2-B, ... , 26 - Z)
29760                                 ;            AL = 7: DX has year, CL has month and CH has date
29761                                 ;            AL = 8: DL has hours, DH has minutes, CL has seconds,
29762                                 ;                    amd CH has hundredths
29763                                 ;            AL = else: cs:SI points to returned string buffer
29764                                 ;       ES:BX -> CONTROL block
29765                                 ;
29766                                 ; Output:   None
29767                                 ;
29768                                 ; Use:      _$P_Do_CAPS_String, _$P_Remove_Colon, _$P_Found_SYNONYM
29769                                 ;
29770                                 ; Vars: _$P_DX(W)
29771                                 ;******************************************************************************
29772
29773                                 _$P_Fill_Result:
29774 00001C27 57                       push    di                  ;AN000;
29775 00001C28 268B7F04                 mov     di,[es:bx+_$P_Control_Blk.Result_Buf]
29776                                                                 ;AN000; di points to result buffer
29777 00001C2C 2E893E[7519]             mov     [cs:_$P_DX],di      ;AC034; set returned result address
29778                                   ;mov    [es:di+_$P_Result_Blk.Type],al ;AN000; store type
29779                                   ;mov    [es:di+_$P_Result_Blk.Item_Tag],ah ;AN000; store item tag
29780                                   ; 07/09/2023
29781                                   ;mov    [es:di+_$P_Result_Blk.Type], ax
29782 00001C31 268905                   mov     [es:di],ax          ; store type (al) and item tag (ah)
29783
29784 00001C34 50                       push    ax                  ;AN000;
29785 00001C35 2EA1[8419]               mov     ax,[cs:_$P_Found_SYNONYM] ;AC034; if yes,
29786 00001C39 26894502                 mov     [es:di+_$P_Result_Blk.SYNONYM_Ptr],ax
29787                                                                 ;AN000;   then set it to the result
29788 00001C3D 58                       pop     ax                  ;AN000;
29789                                 _$P_RLT04:                      ;AN000;
29790 00001C3E 3C01                     cmp     al,_$P_Number       ;AN000; if number
29791 00001C40 750A                     jne     short _$P_RLT00             ;AN000;
29792
29793                                 _$P_RLT02:                      ;AN000;
29794 00001C42 26895504                 mov     [es:di+_$P_Result_Blk.Picked_Val],dx ;AN000; then store 32bit
29795 00001C46 26894D06                 mov     [es:di+_$P_Result_Blk.Picked_Val+2],cx ;AN000; number
29796 00001C4A EB5A                     jmp     short _$P_RLT_Exit  ;AN000;
29797
29798                                 _$P_RLT00:                      ;AN000;
29799 00001C4C 3C02                     cmp     al,_$P_List_Idx             ;AN000; if list index
29800 00001C4E 7506                     jne     short _$P_RLT01             ;AN000;
29801
29802 00001C50 26895504                 mov     [es:di+_$P_Result_Blk.Picked_Val],dx
29803                                                                 ;AN000; then store list index
29804 00001C54 EB50                     jmp     short _$P_RLT_Exit  ;AN000;
29805
29806                                 _$P_RLT01:                      ;AN000;
29807 00001C56 3C07                     cmp     al,_$P_Date_F       ;AN000; Date format ?
29808 00001C58 74E8                     je      short _$P_RLT02             ;AN000;
29809
29810 00001C5A 3C08                     cmp     al,_$P_Time_F       ;AN000; Time format ?
29811 00001C5C 74E4                     je      short _$P_RLT02             ;AN000;
29812
29813 00001C5E 3C06                     cmp     al,_$P_Drive        ;AN000; drive format ?
29814 00001C60 7506                     jne     short _$P_RLT03             ;AN000;
29815
29816 00001C62 26885504                 mov     [es:di+_$P_Result_Blk.Picked_Val],dl ;AN000; store drive number
29817 00001C66 EB3E                     jmp     short _$P_RLT_Exit  ;AN000;
29818
29819                                 _$P_RLT03:                      ;AN000;
29820 00001C68 3C04                     cmp     al,_$P_Complex      ;AN000; complex format ?
29821 00001C6A 750F                     jne     short _$P_RLT05             ;AN000;
29822
29823 00001C6C 2EA1[7E19]               mov     ax,[cs:_$P_SaveSI_Cmpx] ;AC034; then get pointer in command buffer
29824 00001C70 40                       inc     ax                  ;AN000; skip left Parentheses
29825 00001C71 26894504                 mov     [es:di+_$P_Result_Blk.Picked_Val],ax ;AN000; store offset
29826 00001C75 268C5D06                 mov     [es:di+_$P_Result_Blk.Picked_Val+2],ds ;AN000; store segment
29827 00001C79 EB2B                     jmp     short _$P_RLT_Exit  ;AN000;
29828
29829                                 _$P_RLT05:                      ;AN000;
29830                                 ;-----------------------  AL = 3, 5, or 9
29831 00001C7B 26897504                 mov     [es:di+_$P_Result_Blk.Picked_Val],si
29832                                                                 ;AN000; store offset of STRING_BUF
29833 00001C7F 268C4D06                 mov     [es:di+_$P_Result_Blk.Picked_Val+2],cs
29834                                                                 ;AN031; store segment of STRING_BUF
29835 00001C83 50                       push    ax                  ;AN000;
29836 00001C84 26F6470201               test    byte [es:bx+_$P_Control_Blk.Function_Flag],_$P_CAP_File
29837                                                                 ;AN000; need CAPS by file table?
29838 00001C89 7404                     jz      short _$P_RLT_CAP00 ;AN000;
29839
29840 00001C8B B004                     mov     al,_$P_DOSTBL_File  ;AN000; use file upper case table
29841 00001C8D EB09                     jmp     short _$P_RLT_CAP02 ;AN000;
29842
29843                                 _$P_RLT_CAP00:                          ;AN000;
29844 00001C8F 26F6470202               test    byte [es:bx+_$P_Control_Blk.Function_Flag],_$P_CAP_Char
29845                                                                 ;AN000; need CAPS by char table ?
29846 00001C94 7405                     jz      short _$P_RLT_CAP01 ;AN000;
29847
29848 00001C96 B002                     mov     al,_$P_DOSTBL_Char  ;AN000; use character upper case table
29849                                 _$P_RLT_CAP02:                          ;AN000;
29850 00001C98 E8DF00                   call    _$P_Do_CAPS_String  ;AN000;  process CAPS along the table
29851                                 _$P_RLT_CAP01:                          ;AN000;
29852 00001C9B 58                       pop     ax                  ;AN000;
29853 00001C9C 26F6470210               test    byte [es:bx+_$P_Control_Blk.Function_Flag],_$P_Rm_Colon
29854                                                                 ;AN000; removing colon at end ?
29855 00001CA1 7403                     jz      short _$P_RLT_Exit  ;AN000;
29856
29857 00001CA3 E8AE00                   call    _$P_Remove_Colon    ;AN000; then process it.
29858                                 _$P_RLT_Exit:                           ;AN000;
29859 00001CA6 5F                       pop     di                  ;AN000;
29860 00001CA7 C3                       retn                        ;AN000;
29861
29862                                 ;******************************************************************************
29863                                 ; _$P_Check_Match_Flags
29864                                 ;
```

```
                                ; Function:  Check the mutch_flags and make the exit code and set the
                                ;        result buffer
                                ;
                                ;       Check for types in this order:
                                ;          Complex
                                ;          Date
                                ;          Time
                                ;          Drive
                                ;          Filespec
                                ;          Quoted String
                                ;          Simple String
                                ;
                                ; Input:     cs:SI -> _$P_STRING_BUF
                                ;        ES:BX -> CONTROL block
                                ;
                                ; Output:    None
                                ;
                                ; Use:           _$P_Value, P$_SValue, _$P_Simple_String, _$P_Date_Format
                                ;        _$P_Time_Format, _$P_Complex_Format, _$P_File_Foemat
                                ;        _$P_Drive_Format
                                ;*************************************************************************
                                
                                    ; 25/10/2022 - Retro DOS v4.0
                                    ; (MSDOS 5.0 IO.SYS - SYSINIT:19CFh)
                                
                                    ; 14/04/2024 - Retro DOS v5.0
                                    ; (PCDOS 7.1 IBMBIO.COM - SYSINIT:1FC3h)
                                
                                    ; 12/12/2022
                                _$P_Check_Match_Flags:
00001CA8 2EC606[141A]00             mov     byte [cs:_$P_err_flag],_$P_NULL
                                                                    ;AN033;AC034;; clear filespec error flag.
00001CAE 50                         push    ax                      ;AN000;
                                    ;mov    ax,[es:bx+_$P_Control_Blk.Match_Flag]
00001CAF 268B07                     mov     ax,[es:bx]              ;AN000; load match flag(16bit) to ax
00001CB2 09C0                       or      ax,ax                   ;AC035; test ax for zero
00001CB4 7517                       jnz     short _$P_Mat           ;AN000; (tm12)
00001CB6 50                         push    ax                      ;AN000; (tm12)
00001CB7 53                         push    bx                      ;AN000; (tm12)
00001CB8 52                         push    dx                      ;AN000; (tm12)
00001CB9 57                         push    di                      ;AN000; (tm12)
00001CBA 2EC706[7119]0900           mov     word [cs:_$P_RC],_$P_Syntax ;AC034; (tm12)
                                    ;mov    ah,_$P_No_Tag           ;AN000; (tm12)
                                    ;mov    al,_$P_String           ;AN000; (tm12)
                                    ; 07/07/2023
00001CC1 B803FF                     mov     ax,(_$P_No_Tag<<8)|_$P_String
00001CC4 E860FF                     call    _$P_Fill_Result                 ;AN000; (tm12)
00001CC7 5F                         pop     di                      ;AN000; (tm12)
00001CC8 5A                         pop     dx                      ;AN000; (tm12)
00001CC9 5B                         pop     bx                      ;AN000; (tm12)
00001CCA 58                         pop     ax                      ;AN000; (tm12)
                                    ; 12/12/2022
                                    ;jmp    short _$P_Bridge        ;AC035; (tm12)
                                    ; 12/12/2022
                                ;_$P_Mat:                           ;AN000; (tm12)
                                    ;jmp    short _$P_Match03       ;AN025; (tm09)
                                _$P_Bridge:
00001CCB EB6E                       jmp     short _$P_Match_Exit    ;AN000; (tm02)
                                
                                    ; 02/11/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
                                    ; (SYSINIT:19F9h)
                                    ; 12/12/2022
                                    ;nop    ; db 90h
                                
                                ; 12/12/2022
                                _$P_Mat:
                                _$P_Match03:                                ;AN000;
                                    ;test   ax,_$P_Num_Val ; 8000h;AN000; Numeric value
                                    ; 07/07/2023
00001CCD F6C480                     test    ah,(_$P_Num_Val>>8) ; 80h
00001CD0 7412                       jz      short _$P_Match04       ;AN000;
                                
00001CD2 2EC706[7119]0000           mov     word [cs:_$P_RC],_$P_No_Error ;AC034; assume no error
00001CD9 E81E01                     call    _$P_Value               ;AN000; do process
00001CDC 2E833E[7119]09             cmp     word [cs:_$P_RC],_$P_Syntax ;AC034; if error, examine the next type
00001CE2 7557                       jne     short _$P_Match_Exit    ;AN000;
                                _$P_Match04:                                ;AN000;
                                    ;test   ax,_$P_SNum_Val ; 4000h         ;AN000; Signed numeric value
                                    ; 07/07/2023
00001CE4 F6C440                     test    ah,(_$P_SNum_Val>>8) ; 40h
00001CE7 7412                       jz      short _$P_Match05       ;AN000;
                                
00001CE9 2EC706[7119]0000           mov     word [cs:_$P_RC],_$P_No_Error ;AC034; assume no error
00001CF0 E8E300                     call    _$P_SValue              ;AN000; do process
00001CF3 2E833E[7119]09             cmp     word [cs:_$P_RC],_$P_Syntax ;AC034; if error, examine the next type
00001CF9 7540                       jne     short _$P_Match_Exit    ;AN000;
                                _$P_Match05:                                ;AN000;
                                    ;test   ax,_$P_Drv_Only ; 100h;AN000; Drive only
                                    ; 07/07/2023
00001CFB F6C401                     test    ah,(_$P_Drv_Only>>8) ; 1
00001CFE 7415                       jz      short _$P_Match06       ;AN000;
                                
00001D00 2EC706[7119]0000           mov     word [cs:_$P_RC],_$P_No_Error ;AC034; assume no error
00001D07 E8F202                     call    _$P_File_Format                 ;AN000; 1st, call file format
00001D0A E87203                     call    _$P_Drive_Format        ;AN000; check drive format, next
00001D0D 2E833E[7119]09             cmp     word [cs:_$P_RC],_$P_Syntax ;AC034; if error, examine the next type
00001D13 7526                       jne     short _$P_Match_Exit    ;AN000;
                                _$P_Match06:                                ;AN000;
                                    ;test   ax,_$P_File_Spc ; 200h;AN000; File spec
                                    ; 07/07/2023
00001D15 F6C402                     test    ah,(_$P_File_Spc>>8) ; 2
00001D18 7412                       jz      short _$P_Match07       ;AN000;
                                
00001D1A 2EC706[7119]0000           mov     word [cs:_$P_RC],_$P_No_Error ;AC034; assume no error
00001D21 E8D802                     call    _$P_File_Format                 ;AN000; do process
00001D24 2E833E[7119]09             cmp     word [cs:_$P_RC],_$P_Syntax ;AC034; if error, examine the next type
00001D2A 750F                       jne     short _$P_Match_Exit    ;AN000;
                                _$P_Match07:                                ;AN000;
                                    ;test   ax,_$P_Simple_S         ; 2000h ;AN000; Simple string
                                    ; 07/07/2023
00001D2C F6C420                     test    ah,(_$P_Simple_S>>8) ; 20h
00001D2F 740A                       jz      short _$P_Match09       ;AN000;
                                
00001D31 2EC706[7119]0000           mov     word [cs:_$P_RC],_$P_No_Error ;AC034; assume no error
00001D38 E8BA01                     call    _$P_Simple_String       ;AN000; do process
                                _$P_Match09:                                ;AN000;
                                _$P_Match_Exit:                             ;AN000;
00001D3B 2E833E[141A]01             cmp     word [cs:_$P_err_flag],_$P_error_filespec ;AC034; bad filespec ?
00001D41 750F                       jne     short _$P_Match2_Exit ;AN033; no, continue
00001D43 2E833E[7119]00             cmp     word [cs:_$P_RC],_$P_No_Error ;AN033;AC034;; check for other errors ?
00001D49 7507                       jne     short _$P_Match2_Exit ;AN033; no, continue
00001D4B 2EC706[7119]0900           mov     word [cs:_$P_RC],_$P_Syntax ;AN033;AC034;; set error flag
                                _$P_Match2_Exit:                    ;AN033;
00001D52 58                         pop     ax                      ;AN000;
```

```
00001D53 C3                         retn                            ;AN000;
                                    ;*****************************************************************************
                                    ; _$P_Remove_Colon;
                                    ;
                                    ; Function: Remove colon at end
                                    ;
                                    ; Input:    cs:SI points to string buffer to be examineed
                                    ;
                                    ; Output:   None
                                    ;
                                    ; Use:      _$P_Chk_DBCS
                                    ;*****************************************************************************
                                    _$P_Remove_Colon:
00001D54 50                             push    ax                      ;AN000;
00001D55 56                             push    si                      ;AN000;
                                    _$P_RCOL_Loop:                              ;AN000;
00001D56 2E8A04                         mov     al,[cs:si]              ;AN000; get character
00001D59 08C0                           or      al,al                   ;AN000; end of string ?
00001D5B 741A                           jz      short _$P_RCOL_Exit     ;AN000; if yes, just exit

00001D5D 3C3A                           cmp     al,_$P_Colon            ;AN000; is it colon ?
00001D5F 750D                           jne     short _$P_RCOL00        ;AN000;

00001D61 2E807C0100                     cmp     byte [cs:si+1],_$P_NULL ;AN000; if so, next is NULL ?
00001D66 7506                           jne     short _$P_RCOL00        ;AN000; no, then next char

00001D68 2EC60400                       mov     byte [cs:si],_$P_NULL   ;AN000; yes, remove colon
00001D6C EB09                           jmp     short _$P_RCOL_Exit     ;AN000; and exit.

                                    _$P_RCOL00:                             ;AN000;
00001D6E E80E04                         call    _$P_Chk_DBCS            ;AN000; if not colon, then check if
00001D71 7301                           jnc     short _$P_RCOL01        ;AN000; DBCS leading byte.

00001D73 46                             inc     si                      ;AN000; if yes, skip trailing byte
                                    _$P_RCOL01:                             ;AN000;
00001D74 46                             inc     si                      ;AN000; si points to next byte
00001D75 EBDF                           jmp     short _$P_RCOL_Loop     ;AN000; loop until NULL encountered

                                    _$P_RCOL_Exit:                              ;AN000;
00001D77 5E                             pop     si                      ;AN000;
00001D78 58                             pop     ax                      ;AN000;
00001D79 C3                             retn                            ;AN000;

                                    ;*****************************************************************************
                                    ; _$P_Do_CAPS_String;
                                    ;
                                    ; Function: Perform capitalization along with the file case map table
                                    ;       or character case map table.
                                    ;
                                    ; Input:    AL = 2 : Use character table
                                    ;       AL = 4 : Use file table
                                    ;       cs:SI points to string buffer to be capitalized
                                    ;
                                    ; Output:   None
                                    ;
                                    ; Use:      _$P_Do_CAPS_Char, _$P_Chk_DBCS
                                    ;*****************************************************************************

                                    _$P_Do_CAPS_String:
00001D7A 56                             push    si                      ;AN000;
00001D7B 52                             push    dx                      ;AN000;
00001D7C 88C2                           mov     dl,al                   ;AN000; save info id

                                    _$P_DCS_Loop:                               ;AN000;
00001D7E 2E8A04                         mov     al,[cs:si]              ;AN000; load charater and
00001D81 E8FB03                         call    _$P_Chk_DBCS            ;AN000; check if DBCS leading byte
00001D84 720C                           jc      short _$P_DCS00                 ;AN000; if yes, do not need CAPS

00001D86 08C0                           or      al,al                   ;AN000; end of string ?
00001D88 740C                           jz      short _$P_DCS_Exit      ;AN000; then exit.

00001D8A E80C00                         call    _$P_Do_CAPS_Char        ;AN000; Here a SBCS char need to be CAPS
00001D8D 2E8804                         mov     [cs:si],al              ;AN000; stored upper case char to buffer
00001D90 EB01                           jmp     short _$P_DCS01                 ;AN000; process next
                                    _$P_DCS00:                              ;AN000;
00001D92 46                             inc     si                      ;AN000; skip DBCS leading and trailing byte
                                    _$P_DCS01:                              ;AN000;
00001D93 46                             inc     si                      ;AN000; si point to next byte
00001D94 EBE8                           jmp     short _$P_DCS_Loop      ;AN000; loop until NULL encountered
                                    _$P_DCS_Exit:                               ;AN000;
00001D96 5A                             pop     dx                      ;AN000;
00001D97 5E                             pop     si                      ;AN000;
00001D98 C3                             retn

                                    ;*****************************************************************************
                                    ; _$P_Do_CAPS_Char;
                                    ;
                                    ; Function: Perform capitalization along with the file case map table
                                    ;       or character case map table.
                                    ;
                                    ; Input:    DL = 2 : Use character table
                                    ;       DL = 4 : Use file table
                                    ;       AL = character to be capitalized
                                    ;
                                    ; Output:   None
                                    ;
                                    ; Use:      INT 21h /w AH=65h
                                    ;*****************************************************************************

                                    _$P_Do_CAPS_Char:
00001D99 3C80                           cmp     al,_$P_ASCII80 ;80h     ;AN000; need upper case table ?
00001D9B 730B                           jae     short _$P_DCC_Go        ;AN000;

00001D9D 3C61                           cmp     al,"a"                    ;AN000; if no,
00001D9F 7234                           jb      short _$P_CAPS_Ret      ;AN000;   check if  "a" <= AL <= "z"

00001DA1 3C7A                           cmp     al,"z"                    ;AN000;
00001DA3 7730                           ja      short _$P_CAPS_Ret      ;AN000;   if yes, make CAPS

00001DA5 24DF                           and     al,_$P_Make_Upper ;0DFh ;AN000;   else do nothing.
                                        ;jmp    short _$P_CAPS_Ret      ;AN000;
                                        ; 07/07/2023
00001DA7 C3                             retn

                                    _$P_DCC_Go:                             ;AN000;
00001DA8 53                             push    bx                      ;AN000;
00001DA9 06                             push    es                      ;AN000;
00001DAA 57                             push    di                      ;AN000;

                                        ;;lea   di,[cs:_$P_Char_CAP_Ptr] ;AC034; or use char CAPS table ?
                                        ;lea    di,[_$P_Char_CAP_Ptr]
                                        ; 07/09/2023
```

```
30113 00001DAB BF[061A]                mov     di,_$P_Char_CAP_Ptr
30114                                  _$P_DCC00:                              ;AN000;
30115 00001DAE 2E3815                  cmp     [cs:di],dl                      ;AN000; already got table address ?
30116 00001DB1 7415                    je      short _$P_DCC01                 ;AN000; if no,
30117
30118                                  ;In this next section, ES will be used to pass a 5 byte workarea to INT 21h,
30119                                  ; the GET COUNTYRY INFO call. This usage of ES is required by the function
30120                                  ; call, regardless of what base register is currently be defined as cs.
30121
30122 00001DB3 50                      push    ax                              ;AN000; get CAPS table thru DOS call
30123 00001DB4 51                      push    cx                              ;AN000;
30124 00001DB5 52                      push    dx                              ;AN000;
30125
30126 00001DB6 0E                      push    cs                              ;AC036; pass current base seg into
30127                                                                          ;(Note: this used to push CS.  BUG...
30128 00001DB7 07                      pop     es                              ;AN000;   ES reg, required for
30129                                                                          ;get extended country information
30130                                  ;mov    al,dl ; function                ;AN000; upper case table
30131                                  ; 07/07/2023
30132 00001DB8 92                      xchg    ax,dx
30133 00001DB9 B465                    mov     ah,_$P_DOS_Get_TBL ; 65h ;AN000; get extended CDI
30134 00001DBB BBFFFF                  mov     bx,_$P_DOSTBL_Def ; -1;AN000; get active CON
30135 00001DBE B90500                  mov     cx,_$P_DOSTBL_BL ; 5    ;AN000; buffer length
30136                                  ;mov    dx,_$P_DOSTBL_Def               ;AN000; get for default code page
30137                                  ; 07/07/2023
30138 00001DC1 89DA                    mov     dx,bx ; 0FFFFh
30139                                                                          ;DI already set to point to buffer
30140 00001DC3 CD21                    int     21h                             ;AN000; es:di point to buffer that
30141                                                                          ;now has been filled in with info
30142 00001DC5 5A                      pop     dx                              ;AN000;
30143 00001DC6 59                      pop     cx                              ;AN000;
30144 00001DC7 58                      pop     ax                              ;AN000;
30145
30146                                  _$P_DCC01:                              ;AN000;
30147
30148                                  ;In this next section, ES will be used as the base of the XLAT table, provided
30149                                  ; by the previous GET COUNTRY INFO DOS call.  This usage of ES is made
30150                                  ; regardless of which base reg is currently the cs reg.
30151
30152                                  ; 14/04/2024
30153                                  ;mov    bx,[cs:di+_$P_DOS_TBL.Off] ;AN000; get offset of table
30154                                  ;mov    es,[cs:di+_$P_DOS_TBL.Seg] ;AN000; get segment of table
30155                                  ; 07/07/2023
30156 00001DC8 2EC45D01                les     bx,[cs:di+_$P_DOS_TBL.Off]
30157 00001DCC 43                      inc     bx                              ;AC035; add '2' to
30158 00001DCD 43                      inc     bx                              ;AC035;  BX reg
30159                                                                          ;AN000; skip length field
30160 00001DCE 2C80                    sub     al,_$P_ASCII80 ; 80h            ;AN000; make char to index
30161                                  ;xlat   es:[bx]                         ;AN000; perform case map
30162 00001DD0 26                      es
30163 00001DD1 D7                      xlat
30164 00001DD2 5F                      pop     di                              ;AN000;
30165 00001DD3 07                      pop     es                              ;AN000;
30166 00001DD4 5B                      pop     bx                              ;AN000;
30167                                  _$P_CAPS_Ret:                                   ;AN000;
30168 00001DD5 C3                      retn                                    ;AN000;
30169
30170                                  ;****************************************************************************
30171                                  ; _$P_Value / _$P_SValue
30172                                  ;
30173                                  ; Function:  Make 32bit value from cs:SI and see value list
30174                                  ;        and make result buffer.
30175                                  ;        _$P_SValue is an entry point for the signed value
30176                                  ;        and this will simply call _$P_Value after the handling
30177                                  ;        of the sign character, "+" or "-"
30178                                  ;
30179                                  ; Input:     cs:SI -> _$P_STRING_BUF
30180                                  ;        ES:BX -> CONTROL block
30181                                  ;
30182                                  ; Output:    None
30183                                  ;
30184                                  ; Use:      _$P_Fill_Result, _$P_Check_OVF
30185                                  ;
30186                                  ; Vars: _$P_RC(W), _$P_Flags(RW)
30187                                  ;****************************************************************************
30188
30189                                      ; 26/10/2022 - Retro DOS v4.0
30190                                      ; (MSDOS 5.0 IO.SYS - SYSINIT:1B0Bh)
30191
30192                                      ; 07/07/2023 - Retro DOS v4.2 IO.SYS (optimization)
30193                                      ; (MSDOS 6.21 IO.SYS - SYSINIT:1C46h)
30194                                  _$P_SValue:                             ;AN000; when signed value here
30195 00001DD6 50                      push    ax                              ;AN000;
30196 00001DD7 2E800E[7D19]80          or      byte [cs:_$P_Flags2],_$P_Signed ;AC034; indicate a signed numeric
30197 00001DDD 2E8026[7D19]FD          and     byte [cs:_$P_Flags2],0FFh-_$P_Neg ;AC034; assume positive value
30198                                  ;and    byte [cs:_$P_Flags2],~_$P_Neg ; 07/07/2023
30199 00001DE3 2E8A04                  mov     al,[cs:si]                      ;AN000; get sign
30200 00001DE6 3C2B                    cmp     al,_$P_Plus                     ;AN000; "+" ?
30201 00001DE8 740A                    je      short _$P_SVal00                ;AN000;
30202
30203 00001DEA 3C2D                    cmp     al,_$P_Minus                    ;AN000; "-" ?
30204 00001DEC 7507                    jne     short _$P_SVal01                ;AN000; else
30205
30206 00001DEE 2E800E[7D19]02          or      byte [cs:_$P_Flags2],_$P_Neg ;AC034; set this is negative value
30207                                  _$P_SVal00:                             ;AN000;
30208 00001DF4 46                      inc     si                              ;AN000; skip sign char
30209                                  _$P_SVal01:                             ;AN000;
30210 00001DF5 E80200                  call    _$P_Value                       ;AN000; and process value
30211 00001DF8 58                      pop     ax                              ;AN000;
30212 00001DF9 C3                      retn
30213
30214                                  ;****************************************************************************
30215
30216                                      ; 07/07/2023 - Retro DOS v4.2 IO.SYS (optimization)
30217                                      ; (MSDOS 6.21 IO.SYS - SYSINIT:1C6Ah)
30218
30219                                      ; 26/10/2022
30220                                  _$P_Value:                              ;AN000;
30221 00001DFA 50                      push    ax                              ;AN000;
30222 00001DFB 51                      push    cx                              ;AN000;
30223 00001DFC 52                      push    dx                              ;AN000;
30224 00001DFD 56                      push    si                              ;AN000;
30225 00001DFE 31C9                    xor     cx,cx                           ;AN000; cx = higher 16 bits
30226 00001E00 31D2                    xor     dx,dx                           ;AN000; dx = lower 16 bits
30227 00001E02 53                      push    bx                              ;AN000; save control pointer
30228                                  _$P_Value_Loop:                                 ;AN000;
30229 00001E03 2E8A04                  mov     al,[cs:si]                      ;AN000; get character
30230 00001E06 08C0                    or      al,al                           ;AN000; end of line ?
30231 00001E08 7438                    jz      short _$P_Value00               ;AN000;
30232
30233 00001E0A E8DC00                  call    _$P_0099                        ;AN000; make asc(0..9) to bin(0..9)
30234 00001E0D 722F                    jc      short _$P_Value_Err0            ;AN000;
30235
30236 00001E0F 30E4                    xor     ah,ah                           ;AN000;
```

```
30237 00001E11 89C5                     mov     bp,ax                   ;AN000; save binary number
30238
30239                                   ; 08/07/2023 - Retro DOS v4.2 IO.SYS (optimization)
30240                                   ; Ref: Disassembled PCDOS 7.1 IBMBIO.COM SYSINIT code
30241                                   ;                               Erdogan Tan - July 2023
30242                                   %if 0
30243                                       shl     dx,1                ;AN000; to have 2*x
30244                                       rcl     cx,1                ;AN000; shift left w/ carry
30245                                       call    _$P_Check_OVF       ;AN000; Overflow occurred ?
30246                                       jc      short _$P_Value_Err0  ;AN000; then error, exit
30247
30248                                       mov     bx,dx               ;AN000; save low(2*x)
30249                                       mov     ax,cx               ;AN000; save high(2*x)
30250                                       shl     dx,1                ;AN000; to have 4*x
30251                                       rcl     cx,1                ;AN000; shift left w/ carry
30252                                       call    _$P_Check_OVF       ;AN000; Overflow occurred ?
30253                                       jc      short _$P_Value_Err0  ;AN000; then error, exit
30254
30255                                       shl     dx,1                ;AN000; to have 8*x
30256                                       rcl     cx,1                ;AN000; shift left w/ carry
30257                                       call    _$P_Check_OVF       ;AN000; Overflow occurred ?
30258                                       jc      short _$P_Value_Err0  ;AN000; then error, exit
30259
30260                                       add     dx,bx               ;AN000; now have 10*x
30261                                       adc     cx,ax               ;AN000; 32bit ADD
30262                                       call    _$P_Check_OVF       ;AN000; Overflow occurred ?
30263                                       jc      short _$P_Value_Err0  ;AN000; then error, exit
30264
30265                                       add     dx,bp               ;AN000; Add the current one degree decimal
30266                                       adc     cx,0                ;AN000; if carry, add 1 to high 16bit
30267                                       call    _$P_Check_OVF       ;AN000; Overflow occurred ?
30268                                       jc      short _$P_Value_Err0  ;AN000; then error, exit
30269
30270                                       inc     si                  ;AN000; update pointer
30271                                       jmp     short _$P_Value_Loop  ;AN000; loop until NULL encountered
30272                                   ;_$P_Value_Err0:
30273                                   %endif
30274                                   ;****
30275                                   %if 1
30276                                   ; 08/07/2023 - Retro DOS v4.2 IO.SYS (optimization)
30277                                       ; (PCDOS 7.1 IBMBIO.COM - SYSINIT:2130h)
30278
30279                                       ; 14/04/2024 - Retro DOS v5.0
30280                                       ;xor    ah,ah
30281                                       ;mov    bp,ax                   ; save binary number
30282
30283 00001E13 E81C00                     call    _$P_Value_2x_OVF        ; multiply cx:dx by 2 and then check overflow
30284 00001E16 89D3                       mov     bx,dx                   ; ax:bx = 2*(cx:dx)
30285 00001E18 89C8                       mov     ax,cx
30286 00001E1A E81500                     call    _$P_Value_2x_OVF        ; multiply cx:dx by 2 and then check overflow
30287 00001E1D E81200                     call    _$P_Value_2x_OVF        ; multiply cx:dx by 2 and then check overflow
30288 00001E20 01DA                       add     dx,bx                   ; 8*(cx:dx)+2*(cx:dx) = 10*(cx:dx)
30289 00001E22 11C1                       adc     cx,ax
30290 00001E24 E80F00                     call    _$P_Value_Chk_Add_OVF
30291 00001E27 01EA                       add     dx,bp                   ; Add the current one degree decimal
30292                                                                     ; if carry, add 1 to high 16bit
30293 00001E29 83D100                     adc     cx,0
30294 00001E2C E80700                     call    _$P_Value_Chk_Add_OVF ; Overflow occurred ?
30295                                                                     ; then error, exit (without return here)
30296 00001E2F 46                         inc     si                      ; update pointer
30297 00001E30 EBD1                       jmp     short _$P_Value_Loop
30298
30299                                   _$P_Value_2x_OVF:
30300 00001E32 D1E2                       shl     dx,1                    ; to have 2*x
30301 00001E34 D1D1                       rcl     cx,1                    ; shift left w/ carry
30302                                   _$P_Value_Chk_Add_OVF:
30303 00001E36 E89E00                     call    _$P_Check_OVF           ; check overflow (for the last shift or add)
30304 00001E39 7201                       jc      short _$P_Value_OVF
30305 00001E3B C3                         retn
30306                                   _$P_Value_OVF:
30307 00001E3C 44                         inc     sp                      ; skip "call" return address to the caller
30308 00001E3D 44                         inc     sp
30309
30310                                   ;_$P_Value_Err0:
30311                                   %endif
30312                                   ;****
30313
30314                                   _$P_Value_Err0:                             ;AN000;
30315 00001E3E 5B                         pop     bx                      ;AN000;
30316 00001E3F E98300                     jmp     _$P_Value_Err           ;AN000; Bridge
30317                                   ;
30318                                   _$P_Value00:                                ;AN000;
30319 00001E42 5B                         pop     bx                      ;AN000; restore control pointer
30320 00001E43 2EF606[7D19]02             test    byte [cs:_$P_Flags2],_$P_Neg ;AC034; here cx,dx = 32bit value
30321 00001E49 740A                       jz      short _$P_Value01       ;AN000; was it negative ?
30322
30323 00001E4B F7D1                       not     cx                      ;AN000; +
30324 00001E4D F7D2                       not     dx                      ;AN000; |- Make 2's complement
30325 00001E4F 83C201                     add     dx,1                    ;AN000; |
30326 00001E52 83D100                     adc     cx,0                    ;AN000; +
30327
30328                                   _$P_Value01:                                ;AN000; / nval = 0
30329 00001E55 268B7706                   mov     si,[es:bx+_$P_Control_Blk.Value_List] ;AN000; si points to value list
30330 00001E59 268A04                     mov     al,[es:si]              ;AN000; get nval
30331                                     ; 07/09/2023
30332                                     ;cmp    al,_$P_nval_None ; 0    ;AN000; no value list ?
30333                                     ;;*jne  short _$P_Value02       ;AN000;
30334                                     ;;* 07/07/2023
30335                                     ;je     short _$P_Value05
30336                                     ; 07/09/2023
30337 00001E5C 08C0                       or      al,al
30338 00001E5E 7459                       jz      short _$P_Value05 ; _$P_nval_None
30339
30340                                     ;mov    al,_$P_Number           ;AN000; Set type
30341                                     ;mov    ah,_$P_No_Tag           ;AN000; No ITEM_TAG set
30342                                     ; 07/07/2023
30343                                     ;*mov   ax,(_$P_No_Tag<<8)|_$P_Number
30344                                     ;*jmp   short _$P_Value_Exit    ;AN000;
30345
30346                                     ; 26/10/2022 (MSDOS 5.0 IO.SYS, SYSINIT compatibility)
30347                                     ; (SYSINIT:1BA5h)
30348                                     ; 12/12/2022
30349                                     ;nop    ; db  90h
30350
30351                                   _$P_Value02:                                ;AN000; / nval = 1
30352                                   ;IF Val1SW                          ;AN000;(Check if value list id #1 is supported)
30353                                   ;(tm07) cmp al,_$P_nval_Range       ;AN000; have range list ?
30354                                   ;(tm07) jne short _$P_Value03       ;AN000;
30355
30356 00001E60 46                         inc     si                      ;AN000;
30357 00001E61 268A04                     mov     al,[es:si]              ;AN000; al = number of range
30358
30359                                     ; 07/09/2023
30360                                     ;cmp    al,_$P_No_nrng          ;AN000; (tm07)
```

```
30361                                   ;je     short _$P_Value03     ;AN000; (tm07)
30362 00001E64 08C0                     or      al,al
30363 00001E66 745D                     jz      short _$P_Value03 ; _$P_No_nrng
30364
30365 00001E68 46                       inc     si                    ;AN000; si points to 1st item_tag
30366                                 _$P_Val02_Loop:                         ;AN000;
30367 00001E69 2EF606[7D19]80           test    byte [cs:_$P_Flags2],_$P_Signed ;AC034;
30368 00001E6F 751E                     jnz     short _$P_Val02_Sign  ;AN000;
30369
30370 00001E71 263B4C03                 cmp     cx,[es:si+_$P_Val_List.Val_XH] ;AN000; comp cx with XH
30371 00001E75 7234                     jb      short _$P_Val02_Next  ;AN000;
30372 00001E77 7706                     ja      short _$P_Val_In      ;AN000;
30373
30374 00001E79 263B5401                 cmp     dx,[es:si+_$P_Val_List.Val_XL] ;AN000; comp dx with XL
30375 00001E7D 722C                     jb      short _$P_Val02_Next  ;AN000;
30376
30377                                 _$P_Val_In:                     ;AN000;
30378 00001E7F 263B4C07                 cmp     cx,[es:si+_$P_Val_List.Val_YH] ;AN000; comp cx with YH (tm01)
30379 00001E83 7726                     ja      short _$P_Val02_Next  ;AN000;
30380 00001E85 7237                     jb      short _$P_Val_Found   ;AN000;
30381
30382 00001E87 263B5405                 cmp     dx,[es:si+_$P_Val_List.Val_YL] ;AN000; comp dx with YL
30383 00001E8B 771E                     ja      short _$P_Val02_Next  ;AN000;
30384
30385 00001E8D EB2F                     jmp     short _$P_Val_Found   ;AN000;
30386
30387                                 _$P_Val02_Sign:                         ;AN000;
30388 00001E8F 263B4C03                 cmp     cx,[es:si+_$P_Val_List.Val_XH]       ;AN000; comp cx with XH
30389 00001E93 7C16                     jl      short _$P_Val02_Next  ;AN000;
30390 00001E95 7F06                     jg      short _$P_SVal_In     ;AN000;
30391
30392 00001E97 263B5401                 cmp     dx,[es:si+_$P_Val_List.Val_XL]       ;AN000; comp dx with XL
30393 00001E9B 7C0E                     jl      short _$P_Val02_Next  ;AN000;
30394
30395                                 _$P_SVal_In:                            ;AN000;
30396 00001E9D 263B4C07                 cmp     cx,[es:si+_$P_Val_List.Val_YH]       ;AN000; comp cx with YH
30397 00001EA1 7F08                     jg      short _$P_Val02_Next  ;AN000;
30398
30399 00001EA3 7C19                     jl      short _$P_Val_Found   ;AN000;
30400
30401 00001EA5 263B5405                 cmp     dx,[es:si+_$P_Val_List.Val_YL]       ;AN000; comp dx with YL
30402                                   ;jg     short _$P_Val02_Next  ;AN000;
30403                                   ;jmp    short _$P_Val_Found   ;AN000;
30404                                   ; 07/07/2023
30405 00001EA9 7E13                     jng     short _$P_Val_Found
30406
30407                                 _$P_Val02_Next:                         ;AN000;
30408 00001EAB 83C609                   add     si,_$P_Len_Range      ;AN000;
30409 00001EAE FEC8                     dec     al                    ;AN000; loop nrng times in AL
30410 00001EB0 75B7                     jne     short _$P_Val02_Loop  ;AN000;
30411                                                                 ; / Not found
30412 00001EB2 2EC706[7119]0600         mov     word [cs:_$P_RC],_$P_Out_of_Range ;AC034;
30413                                   ;mov    al,_$P_Number         ;AN000;
30414                                   ;mov    ah,_$P_No_Tag         ;AN000; No ITEM_TAG set
30415                                 _$P_Value05:              ;* 07/07/2023
30416                                   ; 07/07/2023
30417 00001EB9 B801FF                   mov     ax,(_$P_No_Tag<<8)|_$P_Number
30418 00001EBC EB11                     jmp     short _$P_Value_Exit  ;AN000;
30419
30420                                 _$P_Val_Found:                          ;AN000;
30421 00001EBE B001                     mov     al,_$P_Number         ;AN000;
30422 00001EC0 268A24                   mov     ah,[es:si]            ;AN000; found ITEM_TAG set
30423 00001EC3 EB0A                     jmp     short _$P_Value_Exit  ;AN000;
30424
30425                                 _$P_Value03:                            ;AN000; / nval = 2
30426
30427                                 ;IF Val2SW                      ;AN000;(Check if value list id #2 is supported)
30428                                 ;;;;cmp     al,$P_nval_Value      ; have match list ? ASSUME nval=2,
30429                                 ;;;;jne     $P_Value04            ; even if it is 3 or more.
30430                                 ;(tm07) inc si                    ;AN000;
30431                                 ;(tm07) mov al,es:[si]            ;AN000; al = nrng
30432                                 ;   mov     ah,$P_Len_Range       ;AN000;
30433                                 ;   mul     ah                    ;AN000;  Skip nrng field
30434                                 ;   inc     ax                    ;AN000;
30435                                 ;   add     si,ax                 ;AN000; si points to nnval
30436                                 ;   mov     al,es:[si]            ;AN000; get nnval
30437                                 ;   inc     si                    ;AN000; si points to 1st item_tag
30438                                 ;$P_Val03_Loop:                         ;AN000;
30439                                 ;   cmp     cx,es:[si+$P_Val_XH]  ;AN000; comp cx with XH
30440                                 ;   jne     $P_Val03_Next         ;AN000;
30441                                 ;
30442                                 ;   cmp     dx,es:[si+$P_Val_XL]  ;AN000; comp dx with XL
30443                                 ;   je      $P_Val_Found          ;AN000;
30444                                 ;
30445                                 ;$P_Val03_Next:                         ;AN000;
30446                                 ;   add     si,$P_Len_Value       ;AN000; points to next value choice
30447                                 ;   dec     al                    ;AN000; loop nval times in AL
30448                                 ;   jne     $P_Val03_Loop         ;AN000;
30449                                 ;                                 ;AN000; / Not found
30450                                 ;   mov     psdata_seg:$P_RC,$P_Not_in_Val ;AC034;
30451                                 ;   mov     al,$P_Number          ;AN000;
30452                                 ;   mov     ah,$P_No_Tag          ;AN000; No ITEM_TAG set
30453                                 ;   jmp     short $P_Value_Exit   ;AN000;
30454                                 ;
30455                                 ;ENDIF                                  ;AN000;(of Val2SW)
30456                                 ;$P_Value04:
30457
30458                                 _$P_Value_Err:                          ;AN000;
30459 00001EC5 2EC706[7119]0900         mov     word [cs:_$P_RC],_$P_Syntax ;AC034;
30460                                   ;mov    al,_$P_String         ;AN000; Set type
30461                                   ;mov    ah,_$P_No_Tag         ;AN000; No ITEM_TAG set
30462                                   ; 07/09/2023
30463                                   ; 07/07/2023
30464 00001ECC B803FF                   mov     ax,(_$P_No_Tag<<8)|_$P_String
30465                                 _$P_Value_Exit:                         ;AN000;
30466 00001ECF E855FD                   call    _$P_Fill_Result               ;AN000;
30467 00001ED2 5E                       pop     si                    ;AN000;
30468 00001ED3 5A                       pop     dx                    ;AN000;
30469 00001ED4 59                       pop     cx                    ;AN000;
30470 00001ED5 58                       pop     ax                    ;AN000;
30471 00001ED6 C3                       retn                          ;AN000;
30472
30473                                 ; 28/03/2019 - Retro DOS v4.0
30474
30475                                 ;--------------------------------------------------------------------------
30476                                 ; _$P_Check_OVF
30477                                 ;
30478                                 ; Function:  Check if overflow is occurred with consideration of
30479                                 ;        signed or un-signed numeric value
30480                                 ;
30481                                 ; Input:     Flag register
30482                                 ;
30483                                 ; Output:    CY = 1  :    Overflow
30484                                 ;
```

```
                                ; Vars:     _$P_Flags(R)
                                ;****************************************************************************
                                
                                    ; 26/10/2022
                                _$P_Check_OVF:
00001ED7 9C                         pushf                           ;AN000;
00001ED8 2EF606[7D19]02             test   byte [cs:_$P_Flags2],_$P_Neg ;AC034; is it negative value ?
00001EDE 7502                       jnz    short _$P_COVF               ;AN000; if no, check overflow
                                
00001EE0 9D                         popf                            ;AN000; by the CY bit
00001EE1 C3                         retn                            ;AN000;
                                
                                _$P_COVF:                           ;AN000;
00001EE2 9D                         popf                            ;AN000; else,
00001EE3 7002                       jo     short _$P_COVF00         ;AN000; check overflow by the OF
                                
00001EE5 F8                         clc                             ;AN000; indicate it with CY bit
00001EE6 C3                         retn                            ;AN000; CY=0 means no overflow
                                
                                _$P_COVF00:                         ;AN000;
00001EE7 F9                         stc                             ;AN000; and CY=1 means overflow
00001EE8 C3                         retn                            ;AN000;
                                
                                ;****************************************************************************
                                ; _$P_0099;
                                ;
                                ; Function:  Make ASCII 0-9 to Binary 0-9
                                ;
                                ; Input:     AL = character code
                                ;
                                ; Output:    CY = 1 : AL is not number
                                ;         CY = 0 : AL contains binary value
                                ;****************************************************************************
                                
                                ; 08/07/2023 - Retro DOS v4.2 IO.SYS (optimization)
                                %if 0
                                _$P_0099:
                                    cmp    al,"0"                   ;AN000;
                                    ;jb    short _$P_0099Err        ;AN000; must be 0 =< al =< 9
                                    ; 12/12/2022
                                    jb     short _$P_0099Err2  ; cf=1
                                
                                    cmp    al,"9"                   ;AN000;
                                    ja     short _$P_0099Err        ;AN000; must be 0 =< al =< 9
                                
                                    sub    al,"0"                   ;AN000; make char -> bin
                                    ; 12/12/2022
                                    ; cf=0
                                    ;clc                            ;AN000; indicate no error
                                    retn                            ;AN000;
                                
                                _$P_0099Err:                               ;AN000;
                                    stc                             ;AN000; indicate error
                                _$P_0099Err2: ; 12/12/2022
                                    retn                            ;AN000;
                                %endif
                                
                                ; 08/07/2023 - Retro DOS v4.2 IO.SYS (optimization)
                                %if 1
                                _$P_0099:
00001EE9 3C30                       cmp    al,"0"                   ; cmp al,30h
00001EEB 7207                       jb     short _$P_0099Err        ; must be 0 =< al =< 9
00001EED 3C3A                       cmp    al,"9"+1                 ; cmp al,3Ah
00001EEF F5                         cmc                             ; cf=0 -> cf=1
00001EF0 7202                       jb     short _$P_0099Err
00001EF2 2C30                       sub    al,"0"  ; sub al,30h     ; make char -> bin
                                    ; cf=0
                                _$P_0099Err:       ; cf=1
00001EF4 C3                         retn
                                %endif
                                
                                ;****************************************************************************
                                ; _$P_Simple_String
                                ;
                                ; Function:  See value list for the simple string
                                ;         and make result buffer.
                                ;
                                ; Input:     cs:SI -> _$P_STRING_BUF
                                ;         ES:BX -> CONTROL block
                                ;
                                ; Output:    None
                                ;
                                ; Use:      _$P_Fill_Result, _$P_String_Comp
                                ;
                                ; Vars: _$P_RC(W)
                                ;****************************************************************************
                                
                                _$P_Simple_String:
00001EF5 50                         push   ax                       ;AN000;
00001EF6 53                         push   bx                       ;AN000;
00001EF7 52                         push   dx                       ;AN000;
00001EF8 57                         push   di                       ;AN000;
00001EF9 268B7F06                   mov    di,[es:bx+_$P_Control_Blk.Value_List] ;AN000; di points to value list
00001EFD 268A05                     mov    al,[es:di]               ;AN000; get nval
00001F00 08C0                       or     al,al                    ;AN000; no value list ?
00001F02 7504                       jnz    short _$P_Sim00                 ;AN000; then
                                
00001F04 B4FF                       mov    ah,_$P_No_Tag            ;AN000; No ITEM_TAG set
00001F06 EB4C                       jmp    short _$P_Sim_Exit       ;AN000; and set result buffer
                                
                                _$P_Sim00:                          ;AN000;
                                ;IF Val3SW+KeySW                    ;AN000;(Check if keyword or value list id #3 is supported)
00001F08 3C03                       cmp    al,_$P_nval_String       ;AN000; String choice list provided ?
00001F0A 753F                       jne    short _$P_Sim01                 ;AN000; if no, syntax error
                                
00001F0C 47                         inc    di                       ;AN000;
00001F0D 268A05                     mov    al,[es:di]               ;AN000; al = nrng
00001F10 B409                       mov    ah,_$P_Len_Range         ;AN000;
00001F12 F6E4                       mul    ah                       ;AN000; Skip nrng field
00001F14 40                         inc    ax                       ;AN000; ax = (nrng*9)+1
00001F15 01C7                       add    di,ax                    ;AN000; di points to nnval
00001F17 268A05                     mov    al,[es:di]               ;AN000; get nnval
00001F1A B405                       mov    ah,_$P_Len_Value         ;AN000;
00001F1C F6E4                       mul    ah                       ;AN000; Skip nnval field
00001F1E 40                         inc    ax                       ;AN000; ax = (nnval*5)+1
00001F1F 01C7                       add    di,ax                    ;AN000; di points to nstrval
00001F21 268A05                     mov    al,[es:di]               ;AN000; get nstrval c
00001F24 47                         inc    di                       ;AC035; add '2' to
00001F25 47                         inc    di                       ;AC035;  DI reg
                                                                    ;AN000; di points to 1st string in list
                                _$P_Sim_Loop:                               ;AN000;
00001F26 268B2D                     mov    bp,[es:di]               ;AN000; get string pointer
00001F29 E83200                     call   _$P_String_Comp                  ;AN000; compare it with operand
00001F2C 7312                       jnc    short _$P_Sim_Found      ;AN000; found on list ?
```

```
30609
30610 00001F2E 83C703                    add     di,_$P_Len_String ; 3 ;AN000; if no, point to next choice
30611 00001F31 FEC8                      dec     al                    ;AN000; loop nstval times in AL
30612 00001F33 75F1                      jne     short _$P_Sim_Loop    ;AN000;
30613                                                                  ;AN000; / Not found
30614 00001F35 2EC706[7119]0800          mov     word [cs:_$P_RC],_$P_Not_In_Str ;AC034;
30615 00001F3C B4FF                      mov     ah,_$P_No_Tag         ;AN000; No ITEM_TAG set
30616 00001F3E EB14                      jmp     short _$P_Sim_Exit    ;AN000;
30617
30618                                  _$P_Sim_Found:                          ;AN000;
30619 00001F40 268A65FF                  mov     ah,[es:di-1]          ;AN000; set item_tag
30620 00001F44 B002                      mov     al,_$P_List_Idx               ;AN000;
30621 00001F46 268B15                    mov     dx,[es:di]            ;AN000; get address of STRING
30622 00001F49 EB0B                      jmp     short _$P_Sim_Exit0   ;AN000;
30623                                  ;ENDIF                                  ;AN000;(of Val3SW+KeySW)
30624                                  _$P_Sim01:                      ;AN000;
30625 00001F4B 2EC706[7119]0900          mov     word [cs:_$P_RC],_$P_Syntax ;AC034;
30626 00001F52 B4FF                      mov     ah,_$P_No_Tag         ;AN000; No ITEM_TAG set
30627                                  _$P_Sim_Exit:                           ;AN000;
30628 00001F54 B003                      mov     al,_$P_String         ;AN000; Set type
30629                                  _$P_Sim_Exit0:                          ;AN000;
30630 00001F56 E8CEFC                    call    _$P_Fill_Result               ;AN000;
30631 00001F59 5F                        pop     di                    ;AN000;
30632 00001F5A 5A                        pop     dx                    ;AN000;
30633 00001F5B 5B                        pop     bx                    ;AN000;
30634 00001F5C 58                        pop     ax                    ;AN000;
30635 00001F5D C3                        retn                          ;AN000;
30636
30637                                  ;******************************************************************************
30638                                  ; _$P_String_Comp:
30639                                  ;
30640                                  ; Function:  Compare two string
30641                                  ;
30642                                  ; Input:     cs:SI -> 1st string
30643                                  ;        ES:BP -> 2nd string  (Must be upper case)
30644                                  ;        ES:BX -> CONTROL block
30645                                  ;
30646                                  ; Output:    CY = 1 if not match
30647                                  ;
30648                                  ; Use:      _$P_Chk_DBCS, _$P_Do_CAPS_Char
30649                                  ;
30650                                  ; Vars: _$P_KEYor_SW_Ptr(W), _$P_Flags(R). _$P_KEYorSW_Ptr
30651                                  ;******************************************************************************
30652
30653                                  _$P_String_Comp:
30654 00001F5E 50                        push    ax                    ;AN000;
30655 00001F5F 55                        push    bp                    ;AN000;
30656 00001F60 52                        push    dx                    ;AN000;
30657 00001F61 56                        push    si                    ;AN000;
30658 00001F62 B202                      mov     dl,_$P_DOSTBL_Char    ;AN000; use character case map table
30659                                  _$P_SCOM_Loop:                          ;AN000;
30660 00001F64 2E8A04                    mov     al,[cs:si]            ;AN000; get command character
30661 00001F67 E81502                    call    _$P_Chk_DBCS          ;AN000; DBCS ?
30662 00001F6A 723A                      jc      short _$P_SCOM00      ;AN000; yes,DBCS
30663
30664 00001F6C E82AFE                    call    _$P_Do_CAPS_Char      ;AN000; else, upper case map before comparison
30665                                  ;IF KeySW+SwSW                          ;AN000;(Check if keyword or switch is supported)
30666 00001F6F 2EF606[7D19]08            test    byte [cs:_$P_Flags2],_$P_Key_Cmp ;AC034; keyword search ?
30667 00001F75 740D                      jz      short _$P_SCOM04      ;AN000;
30668
30669 00001F77 3C3D                      cmp     al,_$P_Keyword        ;AN000; "=" is delimiter
30670 00001F79 751F                      jne     short _$P_SCOM03      ;AN000; IF "=" on command line AND  (bp+1=> char after the "=" in synonym
list)
30671
30672 00001F7B 26807E0100                cmp     byte [es:bp+1],_$P_NULL ;AN021;  at end of keyword string in the control block THEN
30673 00001F80 756D                      jne     short _$P_SCOM_Differ ;AN021;
30674
30675 00001F82 EB13                      jmp     short _$P_SCOM05      ;AN000; keyword found in synonym list
30676
30677                                  _$P_SCOM04:                             ;AN000;
30678 00001F84 2EF606[7D19]10            test    byte [cs:_$P_Flags2],_$P_SW_Cmp ;AC034; switch search ?
30679 00001F8A 740E                      jz      short _$P_SCOM03      ;AN000;
30680
30681 00001F8C 3C3A                      cmp     al,_$P_Colon          ;AN000; ":" is delimiter, at end of switch on command line
30682 00001F8E 750A                      jne     short _$P_SCOM03      ;AN000; continue compares
30683
30684                                    ; 26/10/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
30685                                    ;cmp    byte [es:bp+0],_$P_NULL
30686                                    ; 11/12/2022
30687 00001F90 26807E0000                cmp     byte [es:bp],_$P_NULL ;AN021; IF at end of switch on command AND
30688 00001F95 7558                      jne     short _$P_SCOM_Differ ;AN021;   at end of switch string in the control block THEN
30689
30690                                  _$P_SCOM05:                             ;AN000;   found a match
30691 00001F97 46                        inc     si                    ;AN000; si points to just after "=" or ":"
30692 00001F98 EB58                      jmp     short _$P_SCOM_Same   ;AN000; exit
30693
30694                                  _$P_SCOM03:                             ;AN000;
30695                                  ;ENDIF                                  ;AN000;(of KeySW+SwSW)
30696                                    ; 26/10/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
30697                                    ;cmp    al,[es:bp+0]
30698                                    ; 11/12/2022
30699 00001F9A 263A4600                  cmp     al,[es:bp]            ;AN000; compare operand w/ a synonym
30700 00001F9E 751B                      jne     short _$P_SCOM_Differ0        ;AN000; if different, check ignore colon option
30701
30702 00001FA0 08C0                      or      al,al                 ;AN000; end of line
30703 00001FA2 744E                      jz      short _$P_SCOM_Same   ;AN000; if so, exit
30704
30705                                    ; 12/12/2022
30706                                    ;inc    si                    ;AN000; update operand pointer
30707                                    ;inc    bp                    ;AN000;    and synonym pointer
30708                                    ; 26/10/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
30709 00001FA4 EB11                      jmp     short _$P_SCOM01      ;AN000; loop until NULL or "=" or ":" found in case
30710
30711                                  _$P_SCOM00:                             ;AN000; Here al is DBCS leading byte
30712                                    ; 26/10/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
30713                                    ;cmp    al,[es:bp+0]
30714                                    ; 11/12/2022
30715 00001FA6 263A4600                  cmp     al,[es:bp]            ;AN000; compare leading byte
30716 00001FAA 7543                      jne     short _$P_SCOM_Differ ;AN000; if not match, say different
30717
30718 00001FAC 46                        inc     si                    ;AN000; else, load next byte
30719 00001FAD 2E8A04                    mov     al,[cs:si]            ;AN000; and
30720 00001FB0 45                        inc     bp                    ;AN000;
30721                                    ; 26/10/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
30722                                    ;cmp    al,[es:bp+0]
30723                                    ; 11/12/2022
30724 00001FB1 263A4600                  cmp     al,[es:bp]            ;AN000; compare 2nd byte
30725 00001FB5 7538                      jne     short _$P_SCOM_Differ ;AN000; if not match, say different, too
30726
30727                                    ; 26/10/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
30728                                    ; 12/12/2022
30729                                  _$P_SCOM01:
30730 00001FB7 46                        inc     si                    ;AN000; else update operand pointer
30731 00001FB8 45                        inc     bp                    ;AN000;                and synonym pointer
```

```
                                ;_$P_SCOM01:                            ;AN000;
00001FB9 EBA9                       jmp     short _$P_SCOM_Loop   ;AN000; loop until NULL or "=" or "/" found in case

                                _$P_SCOM_Differ0:               ;AN000;
                                ;IF SwSW                        ;AN000;(tm10)
00001FBB 2EF606[7D19]40             test    byte [cs:_$P_Flags2],_$P_SW ;AC034;(tm10)
00001FC1 740E                       jz      short _$P_not_applicable ;AN000;(tm10)

                                    ; 26/10/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
                                    ;test   word [es:bx+_$P_Control_Blk.Function_Flag],_$P_colon_is_not_necessary ;AN000;(tm10)
                                    ; 12/12/2022
00001FC3 26F6470220                 test    byte [es:bx+_$P_Control_Blk.Function_Flag],_$P_colon_is_not_necessary
00001FC8 7407                       jz      short _$P_not_applicable ;AN000;(tm10)

                                    ; 26/10/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
                                    ;cmp    byte [es:bp+0],_$P_NULL
                                    ; 11/12/2022
00001FCA 26807E0000                 cmp     byte [es:bp],_$P_NULL ;AN000;(tm10)
                                ;(deleted ;AN025;) jne short _$P_not_applicable ;AN000;(tm10)
00001FCF 7421                       je      short _$P_SCOM_Same   ;AN025;(tm10)

                                _$P_not_applicable:                     ;AN000;(tm10)
                                ;ENDIF                                  ;AN000;(tm10)

                                    ;test   word [es:bx+_$P_Control_Blk.Match_Flag],_$P_Ig_Colon
                                                                        ;AN000; ignore colon option specified ?
                                    ;test   byte [es:bx+_$P_Control_Blk.Match_Flag],_$P_Ig_Colon
                                    ; 12/12/2022
00001FD1 26F60710                   test    byte [es:bx],_$P_Ig_Colon
                                    ; 26/10/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
                                    ;test   word [es:bx],_$P_Ig_Colon ; 10h
00001FD5 7418                       jz      short _$P_SCOM_Differ ;AN000; if no, say different.

00001FD7 3C3A                       cmp     al,_$P_Colon            ;AN000; End up with ":" and
00001FD9 7509                       jne     short _$P_SCOM02        ;AN000;    subseqently

                                    ; 26/10/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
                                    ;cmp    byte [es:bp+0],_$P_NULL
                                    ; 11/12/2022
00001FDB 26807E0000                 cmp     byte [es:bp],_$P_NULL ;AN000; NULL ?
00001FE0 750D                       jne     short _$P_SCOM_Differ ;AN000; if no, say different

00001FE2 EB0E                       jmp     short _$P_SCOM_Same   ;AN000; else, say same

                                _$P_SCOM02:                             ;AN000;
00001FE4 3C00                       cmp     al,_$P_NULL             ;AN000; end up NULL and :
00001FE6 7507                       jne     short _$P_SCOM_Differ ;AN000;

                                    ; 26/10/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
                                    ;cmp    byte [es:bp+0],_$P_Colon
                                    ; 11/12/2022
00001FE8 26807E003A                 cmp     byte [es:bp],_$P_Colon;AN000; if no, say different
00001FED 7403                       je      short _$P_SCOM_Same   ;AN000; else, say same

                                _$P_SCOM_Differ:                ;AN000;
00001FEF F9                         stc                             ;AN000; indicate not found
00001FF0 EB05                       jmp     short _$P_SCOM_Exit   ;AN000;

                                _$P_SCOM_Same:                          ;AN000;
                                    ; 12/12/2022
                                    ; cf=0
00001FF2 2E8936[8019]               mov     [cs:_$P_KEYorSW_Ptr],si ;AC034; for later use by keyword or switch
                                    ; 12/12/2022
                                    ;clc                            ;AN000; indicate found
                                _$P_SCOM_Exit:                          ;AN000;
00001FF7 5E                         pop     si                      ;AN000;
00001FF8 5A                         pop     dx                      ;AN000;
00001FF9 5D                         pop     bp                      ;AN000;
00001FFA 58                         pop     ax                      ;AN000;
00001FFB C3                         retn

                                ; 30/03/2019

                                ;IF FileSW+DrvSW                ;AN000;(Check if file spec or drive only is supported)

                                ;******************************************************************************
                                ; _$P_File_Format;
                                ;
                                ; Function:  Check if the input string is valid file spec format.
                                ;        And set the result buffer.
                                ;
                                ; Input:     cs:SI -> _$P_STRING_BUF
                                ;        ES:BX -> CONTROL block
                                ;
                                ; Output:    None
                                ;
                                ; Use:      _$P_Fill_Result, _$P_Chk_DBCS, _$P_FileSp_Chk
                                ;
                                ; Vars: _$P_RC(W), _$P_SI_Save(W), _$P_Terminator(W), _$P_SaveSI_Cmpx(R)
                                ;   _$P_SaveSI_Cmpx(R)
                                ;******************************************************************************

                                _$P_File_Format:
00001FFC 50                         push    ax                      ;AN000;
00001FFD 57                         push    di                      ;AN000;
00001FFE 56                         push    si                      ;AN000;
00001FFF 2E8B3E[7E19]               mov     di,[cs:_$P_SaveSI_Cmpx]       ;AC034; get user buffer address
                                _$P_FileF_Loop0:                ;AN000; / skip special characters
00002004 2E8A04                     mov     al,[cs:si]              ;AN000; load character
00002007 08C0                       or      al,al                   ;AN000; end of line ?
00002009 7413                       jz      short _$P_FileF_Err   ;AN000; if yes, error exit

0000200B E85D00                     call    _$P_FileSp_Chk          ;AN000; else, check if file special character
0000200E 7523                       jne     short _$P_FileF03     ;AN000; if yes,

00002010 2EC606[141A]01             mov     byte [cs:_$P_err_flag],_$P_error_filespec
                                                                    ;AN033;AC034;; set error flag- bad char.
00002016 5E                         pop     si                      ;AN033;
00002017 2EC60400                   mov     byte [cs:si],_$P_NULL ;AN033;
0000201B 5F                         pop     di                      ;AN033;
0000201C EB3E                       jmp     short _$P_FileF02     ;AN033;

                                _$P_FileF_Err:                          ;AN000;
0000201E 5E                         pop     si                      ;AN000;
0000201F 2EC60400                   mov     byte [cs:si],_$P_NULL ;AN000;
00002023 5F                         pop     di                      ;AN000;

                                    ;test   word [es:bx+_$P_Control_Blk.Match_Flag],_$P_Optional ;AN000; is it optional ?
                                    ;test   byte [es:bx+_$P_Control_Blk.Match_Flag],_$P_Optional
                                    ; 12/12/2022
00002024 26F60701                   test    byte [es:bx],_$P_Optional
                                    ; 26/10/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
                                    ;test   word [es:bx],_$P_Optional
00002028 7532                       jnz     short _$P_FileF02     ;AN000;
```

```
30856
30857 0000202A 2EC706[7119]0200          mov     word [cs:_$P_RC],_$P_Op_Missing ;AC034; 3/17/87
30858 00002031 EB29                     jmp     short _$P_FileF02       ;AN000;
30859
30860                                 _$P_FileF03:                            ;AN000;
30861 00002033 58                       pop     ax                      ;AN000; discard save si
30862 00002034 56                       push    si                      ;AN000; save new si
30863                                 _$P_FileF_Loop1:                        ;AN000;
30864 00002035 2E8A04                   mov     al,[cs:si]              ;AN000; load character (not special char)
30865 00002038 08C0                     or      al,al                   ;AN000; end of line ?
30866 0000203A 741E                     jz      short _$P_FileF_RLT     ;AN000;
30867
30868 0000203C E82C00                   call    _$P_FileSp_Chk          ;AN000; File special character ?
30869 0000203F 740B                     je      short _$P_FileF00       ;AN000;
30870
30871 00002041 E83B01                   call    _$P_Chk_DBCS            ;AN000; no, then DBCS ?
30872 00002044 7302                     jnc     short _$P_FileF01       ;AN000;
30873 00002046 47                       inc     di                      ;AN000; if yes, skip next byte
30874 00002047 46                       inc     si                      ;AN000;
30875                                 _$P_FileF01:                            ;AN000;
30876 00002048 47                       inc     di                      ;AN000;
30877 00002049 46                       inc     si                      ;AN000;
30878 0000204A EBE9                     jmp     short _$P_FileF_Loop1   ;AN000;
30879                                 ;
30880                                 _$P_FileF00:                            ;AN000;
30881 0000204C 2EA2[7719]               mov     [cs:_$P_Terminator],al;AC034;
30882 00002050 2EC60400                 mov     byte [cs:si],_$P_NULL   ;AN000; update end of string
30883 00002054 47                       inc     di                      ;AN000;
30884 00002055 2E893E[7319]             mov     [cs:_$P_SI_Save],di     ;AC034; update next pointer in command line
30885                                 _$P_FileF_RLT:                          ;AN000;
30886 0000205A 5E                       pop     si                      ;AN000;
30887 0000205B 5F                       pop     di                      ;AN000;
30888                                 _$P_FileF02:                            ;AN000;
30889 0000205C 58                       pop     ax                      ;AN000; (tm14)
30890                                   ;test   ax,_$P_File_Spc         ; 200h ;AN000; (tm14)
30891                                   ; 08/07/2023
30892 0000205D F6C402                   test    ah,(_$P_File_Spc>>8) ; 2
30893 00002060 7408                     jz      short _$P_Drv_Only_Exit       ;AN000; (tm14)
30894
30895 00002062 50                       push    ax                      ;AN000; (tm14)
30896                                   ;mov    ah,_$P_No_Tag           ;AN000; set
30897                                   ;mov    al,_$P_File_Spec        ;AN000; result
30898                                   ; 08/07/2023
30899 00002063 B805FF                   mov     ax,(_$P_No_Tag<<8)|_$P_File_Spec ; 0FF05h
30900                                                                   ; set result
30901 00002066 E8BEFB                   call    _$P_Fill_Result                 ;AN000; buffer to file spec
30902 00002069 58                       pop     ax                      ;AN000;
30903
30904                                 _$P_Drv_Only_Exit:                      ;AN000; (tm14)
30905 0000206A C3                       retn                            ;AN000;
30906
30907                                 ;******************************************************************************
30908                                 ; _$P_FileSp_Chk
30909                                 ;
30910                                 ; Function:  Check if the input byte is one of file special characters
30911                                 ;
30912                                 ; Input:     cs:SI -> _$P_STRING_BUF
30913                                 ;        AL = character code to be examineed
30914                                 ;
30915                                 ; Output:    ZF = 1 , AL is one of special characters
30916                                 ;******************************************************************************
30917
30918                                 _$P_FileSp_Chk:
30919 0000206B 53                       push    bx                      ;AN000;
30920 0000206C 51                       push    cx                      ;AN000;
30921                                   ;;lea   bx,[cs:_$P_FileSp_Char] ;AC034; special character table
30922                                   ;lea    bx,[_$P_FileSp_Char]    ; "[]|<>+=;\"" at
30923                                                                   ; MSDOS 6.21 IO.SYS - SYSINIT:1838h
30924                                   ; 07/09/2023
30925 0000206D BB[0B1A]                 mov     bx,_$P_FileSp_Char
30926 00002070 B90900                   mov     cx,_$P_FileSp_Len ; 9 ;AN000; load length of it
30927                                 _$P_FileSp_Loop:                        ;AN000;
30928 00002073 2E3A07                   cmp     al,[cs:bx]              ;AN000; is it one of special character ?
30929 00002076 7404                     je      short _$P_FileSp_Exit ;AN000;
30930
30931 00002078 43                       inc     bx                      ;AN000;
30932 00002079 E2F8                     loop    _$P_FileSp_Loop                 ;AN000;
30933
30934 0000207B 41                       inc     cx                      ;AN000; reset ZF
30935                                 _$P_FileSp_Exit:                        ;AN000;
30936 0000207C 59                       pop     cx                      ;AN000;
30937 0000207D 5B                       pop     bx                      ;AN000;
30938 0000207E C3                       retn
30939
30940                                 ;ENDIF                                  ;AN000;(of FileSW+DrvSW)
30941
30942                                 ;IF DrvSW                       ;AN000;(Check if drive only is supported)
30943
30944                                 ;******************************************************************************
30945                                 ; _$P_Drive_Format;
30946                                 ;
30947                                 ; Function:  Check if the input string is valid drive only format.
30948                                 ;        And set the result buffer.
30949                                 ;
30950                                 ; Input:     cs:SI -> _$P_STRING_BUF
30951                                 ;        ES:BX -> CONTROL block
30952                                 ;
30953                                 ; Output:    None
30954                                 ;
30955                                 ; Use:      _$P_Fill_Result, _$P_Chk_DBCS
30956                                 ;
30957                                 ; Vars: _$P_RC(W)
30958                                 ;******************************************************************************
30959
30960                                 _$P_Drive_Format:
30961 0000207F 50                       push    ax                      ;AN000;
30962 00002080 52                       push    dx                      ;AN000;
30963 00002081 2E8A04                   mov     al,[cs:si]              ;AN000;
30964 00002084 08C0                     or      al,al                   ;AN000; if null string
30965 00002086 7436                     je      short _$P_Drv_Exit      ;AN000; do nothing
30966
30967 00002088 E8F400                   call    _$P_Chk_DBCS            ;AN000; is it leading byte ?
30968 0000208B 722A                     jc      short _$P_Drv_Err       ;AN000;
30969
30970 0000208D 2E837C013A               cmp     word [cs:si+1],_$P_Colon ;AN000; "d", ":", 0  ?
30971 00002092 740D                     je      short _$P_DrvF00        ;AN000;
30972
30973                                   ;test   word [es:bx+_$P_Control_Blk.Match_Flag],_$P_Ig_Colon
30974                                   ;test   byte [es:bx+_$P_Control_Blk.Match_Flag],_$P_Ig_Colon ;AN000; colon can be ignored?
30975                                   ; 12/12/2022
30976 00002094 26F60710                 test    byte [es:bx],_$P_Ig_Colon
30977                                   ; 26/10/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
30978                                   ;test   word [es:bx],_$P_Ig_Colon
30979 00002098 741D                     jz      short _$P_Drv_Err       ;AN000;
```

```
30980
30981 0000209A 2E807C0100              cmp     byte [cs:si+1],_$P_NULL ;AN000; "d", 0  ?
30982 0000209F 7516                    jne     short _$P_Drv_Err       ;AN000;
30983
30984                                 _$P_DrvF00:                             ;AN000;
30985 000020A1 0C20                    or      al,_$P_Make_Lower       ;AN000; lower case
30986 000020A3 3C61                    cmp     al,"a"                    ;AN000; drive letter must
30987 000020A5 7210                    jb      short _$P_Drv_Err       ;AN000; in range of
30988
30989 000020A7 3C7A                    cmp     al,"z"                    ;AN000; "a"-"z"
30990 000020A9 770C                    ja      short _$P_Drv_Err       ;AN000; if no, error
30991
30992 000020AB 2C60                    sub     al,"a"-1                  ;AN000; make text drive to binary drive
30993 000020AD 88C2                    mov     dl,al                   ;AN000; set
30994                                  ;mov    ah,_$P_No_Tag           ;AN000; result
30995                                  ;mov    al,_$P_Drive            ;AN000; buffer
30996                                  ; 08/07/2023
30997 000020AF B806FF                  mov     ax,(_$P_No_Tag<<8)|_$P_Drive ; 0FF06h
30998                                                                  ; set result buffer
30999 000020B2 E872FB                  call    _$P_Fill_Result           ;AN000; to drive
31000 000020B5 EB07                    jmp     short _$P_Drv_Exit      ;AN000;
31001
31002                                 _$P_Drv_Err:                            ;AN000;
31003 000020B7 2EC706[7119]0900        mov     word [cs:_$P_RC],_$P_Syntax ;AC034;
31004                                 _$P_Drv_Exit:                           ;AN000;
31005 000020BE 5A                      pop     dx                      ;AN000;
31006 000020BF 58                      pop     ax                      ;AN000;
31007 000020C0 C3                      retn                            ;AN000;
31008
31009                                 ;ENDIF                                  ;AN000;(of DrvSW)
31010
31011                                 ;******************************************************************************
31012                                 ; _$P_Skip_Delim;
31013                                 ;
31014                                 ; Function: Skip delimiters specified in the PARMS list, white space
31015                                 ;       and comma.
31016                                 ;
31017                                 ; Input:    DS:SI -> Command String
31018                                 ;       ES:DI -> Parameter List
31019                                 ;
31020                                 ; Output:   CY = 1 if the end of line encounterd
31021                                 ;       CY = 0 then SI move to 1st non-delimiter character
31022                                 ;       AL = Last examineed character
31023                                 ;
31024                                 ; Use:          _$P_Chk_EOL, _$P_Chk_Delim,
31025                                 ;
31026                                 ; Vars:     _$P_Flags(R)
31027                                 ;******************************************************************************
31028
31029                                 _$P_Skip_Delim:
31030                                 _$P_Skip_Delim_Loop:                    ;AN000;
31031 000020C1 AC                      lodsb                           ;AN000;
31032 000020C2 E81E00                  call    _$P_Chk_EOL             ;AN000; is it EOL character ?
31033 000020C5 7416                    jz      short _$P_Skip_Delim_CY       ;AN000; if yes, exit w/ CY on
31034
31035 000020C7 E84E00                  call    _$P_Chk_Delim           ;AN000; is it one of delimiters ?
31036 000020CA 7514                    jnz     short _$P_Skip_Delim_NCY ;AN000; if no, exit w/ CY off
31037
31038 000020CC 2EF606[7D19]20          test    byte [cs:_$P_Flags2],_$P_Extra ;AC034; extra delim or comma found ?
31039 000020D2 74ED                    jz      short _$P_Skip_Delim_Loop ;AN000; if no, loop
31040
31041 000020D4 2EF606[7D19]41          test    byte [cs:_$P_Flags2],_$P_SW+_$P_equ ;AC034; /x , or xxx=zzz , (tm08)
31042                                  ;jz     short _$P_Exit_At_Extra       ;AN000; no switch, no keyword (tm08)
31043                                  ; 08/07/2023
31044                                  ; cf=0
31045 000020DA 7505                    jnz     short _$P_Skip_Delim_Exit
31046 000020DC C3                      retn
31047
31048                                  ;dec    si                      ;AN000; backup si for next call (tm08)
31049                                  ;jmp    short _$P_Exit_At_Extra       ;AN000; else exit w/ CY off
31050                                  ; 12/12/2022
31051                                  ; cf=0
31052                                  ; 08/07/2023
31053                                  ;jmp    short _$P_Skip_Delim_Exit
31054
31055                                 _$P_Skip_Delim_CY:                      ;AN000;
31056 000020DD F9                      stc                             ;AN000; indicate EOL
31057 000020DE EB01                    jmp     short _$P_Skip_Delim_Exit ;AN000;
31058
31059                                 _$P_Skip_Delim_NCY:                     ;AN000;
31060 000020E0 F8                      clc                             ;AN000; indicate non delim
31061                                 _$P_Skip_Delim_Exit:                    ;AN000; in this case, need
31062 000020E1 4E                      dec     si                      ;AN000;  backup index pointer
31063                                  ; 08/07/2023
31064                                  ; 12/12/2022
31065                                 ;_$P_Exit_At_Extra:        ; cf=0
31066 000020E2 C3                      retn                            ;AN000;
31067
31068                                  ; 12/12/2022
31069                                 ;_$P_Exit_At_Extra:                     ;AN000;
31070                                  ;clc                           ;AN000; indicate extra delim
31071                                  ;retn                          ;AN000;
31072
31073                                 ;******************************************************************************
31074                                 ; _$P_Chk_EOL;
31075                                 ;
31076                                 ; Function: Check if AL is one of End of Line characters.
31077                                 ;
31078                                 ; Input:    AL = character code
31079                                 ;       ES:DI -> Parameter List
31080                                 ;
31081                                 ; Output:   ZF = 1 if one of End of Line characters
31082                                 ;******************************************************************************
31083
31084                                 _$P_Chk_EOL:
31085 000020E3 53                      push    bx                      ;AN000;
31086 000020E4 51                      push    cx                      ;AN000;
31087 000020E5 3C0D                    cmp     al,_$P_CR               ;AN000; Carriage return ?
31088 000020E7 742C                    je      short _$P_Chk_EOL_Exit        ;AN000;
31089 000020E9 3C00                    cmp     al,_$P_NULL             ;AN000; zero ?
31090 000020EB 7428                    je      short _$P_Chk_EOL_Exit        ;AN000;
31091                                 ;IF LFEOLSW                             ;AN028; IF LF TO BE ACCEPTED AS EOL
31092 000020ED 3C0A                    cmp     al,_$P_LF               ;AN000; Line feed ?
31093 000020EF 7424                    je      short _$P_Chk_EOL_Exit        ;AN000;
31094                                 ;ENDIF                                  ;AN028;
31095 000020F1 26807D0202              cmp     byte [es:di+_$P_PARMS_Blk.Num_Extra],_$P_I_Have_EOL
31096                                                                  ;AN000; EOL character specified ?
31097 000020F6 721D                    jb      short _$P_Chk_EOL_Exit        ;AN000;
31098 000020F8 31DB                    xor     bx,bx                   ;AN000;
31099 000020FA 268A5D03                mov     bl,[es:di+_$P_PARMS_Blk.Len_Extra_Delim]
31100                                                                  ;AN000; get length of delimiter list
31101 000020FE 83C304                  add     bx,_$P_Len_PARMS        ;AN000; skip it
31102                                  ; 08/07/2023
31103 00002101 31C9                    xor     cx,cx ; *
```

```
31104 00002103 26803900                     cmp     byte [es:bx+di],_$P_I_Use_Default ;AN000; No extra EOL character ?
31105 00002107 740B                         je      short _$P_Chk_EOL_NZ  ;AN000;
31106                                       ; 08/07/2023
31107                                       ;;xor   cx,cx                 ;AN000; Get number of extra character
31108                                       ;xor    ch,ch ; *
31109 00002109 268A09                       mov     cl,[es:bx+di]         ;AN000;
31110                                     _$P_Chk_EOL_Loop:               ;AN000;
31111 0000210C 43                           inc     bx                    ;AN000;
31112 0000210D 263A01                       cmp     al,[es:bx+di]         ;AN000; Check extra EOL character
31113 00002110 7403                         je      short _$P_Chk_EOL_Exit        ;AN000;
31114 00002112 E2F8                         loop    _$P_Chk_EOL_Loop      ;AN000;
31115                                       ; 08/07/2023
31116                                       ; cx=0
31117                                     _$P_Chk_EOL_NZ:                         ;AN000;
31118                                       ;cmp    al,_$P_CR             ;AN000; reset ZF
31119                                       ; 08/07/2023
31120 00002114 41                           inc     cx  ; zf=0  (cx=1) ; *
31121                                     _$P_Chk_EOL_Exit:               ;AN000;
31122 00002115 59                           pop     cx                    ;AN000;
31123 00002116 5B                           pop     bx                    ;AN000;
31124 00002117 C3                           retn
31125
31126                                     ;****************************************************************************
31127                                     ; _$P_Chk_Delim;
31128                                     ;
31129                                     ; Function: Check if AL is one of delimiter characters.
31130                                     ;       if AL+[si] is DBCS blank, it is replaced with two SBCS
31131                                     ;       blanks.
31132                                     ;
31133                                     ; Input:    AL = character code
31134                                     ;       DS:SI -> Next Character
31135                                     ;       ES:DI -> Parameter List
31136                                     ;
31137                                     ; Output:   ZF = 1 if one of delimiter characters
31138                                     ;       SI points to the next character
31139                                     ; Vars:  _$P_Terminator(W), _$P_Flags(W)
31140                                     ;****************************************************************************
31141
31142                                       ; 08/07/2023 - Retro DOS v4.2 IO.SYS (optimization)
31143                                       ; MSDOS 6.21 IO.SYS - SYSINIT:1FAEh
31144                                       ; (PCDOS 7.1 IBMBIO.COM - SYSINIT:2451h) ; (Retro DOS v5.0)
31145
31146                                     _$P_Chk_Delim:
31147 00002118 53                           push    bx                    ;AN000;
31148 00002119 51                           push    cx                    ;AN000;
31149 0000211A 2EC606[7719]20               mov     byte [cs:_$P_Terminator],_$P_Space
31150                                                                       ;AC034; Assume terminated by space
31151                                       ;and    byte [cs:_$P_Flags20,0DFh
31152 00002120 2E8026[7D19]DF               and     byte [cs:_$P_Flags2],0FFh-_$P_Extra ;AC034;
31153 00002126 3C20                         cmp     al,_$P_Space ; 20h    ;AN000; Space ?
31154 00002128 7423                         je      short _$P_Chk_Delim_Exit ;AN000;
31155
31156 0000212A 3C09                         cmp     al,_$P_TAB            ;AN000; TAB ?
31157 0000212C 741F                         je      short _$P_Chk_Delim_Exit ;AN000;
31158
31159 0000212E 3C2C                         cmp     al,_$P_Comma          ;AN000; Comma ?
31160 00002130 741E                         je      short _$P_Chk_Delim_Exit0 ;AN000;
31161
31162                                     ; Note: _$P_Chk_Delim00 part of code is nonsense here
31163                                     ;       because _$P_Space = _$P_DBSP1 = 20h
31164                                     ;       Erdogan Tan - 08/07/2023 - Retro DOS v4.2 IO.SYS (optimization)
31165                                     ;_$P_Chk_Delim00:
31166                                       ; (PCDOS 7.1 IBMBIO.COM - SYSINIT:246Bh)
31167                                       ; (MSDOS 6.21 IO.SYS - SYSINIT:1FC8h)
31168                                     %if 0
31169                                       ; 26/10/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
31170                                     _$P_Chk_Delim00:                ;AN000;
31171                                       cmp     al,_$P_DBSP1   ; 20h  ;AN000; 1st byte of DBCS Space ?
31172                                       jne     short _$P_Chk_Delim01 ;AN000;
31173
31174                                       cmp     byte [si],_$P_DBSP2 ; 20h ;AN000; 2nd byte of DBCS Space ?
31175                                       jne     short _$P_Chk_Delim01 ;AN000;
31176
31177                                       mov     al,_$P_Space          ;AN000;
31178                                       inc     si                    ;AN000; make si point to next character
31179                                       cmp     al,al                 ;AN000; Set ZF
31180                                       jmp     short _$P_Chk_Delim_Exit ;AN000;
31181                                     %endif
31182
31183                                     _$P_Chk_Delim01:                ;AN000;
31184 00002132 26807DFE01                   cmp     byte [es:di-_$P_PARMS_Blk.Num_Extra],_$P_I_Have_Delim
31185                                                                       ;AN000; delimiter character specified ?
31186 00002137 7214                         jb      short _$P_Chk_Delim_Exit ;AN000;
31187
31188                                       ;xor    cx,cx                 ;AN000;
31189 00002139 30ED                         xor     ch,ch
31190                                       ;mov    cl,[es:di+3]
31191 0000213B 268A4D03                     mov     cl,[es:di+_$P_PARMS_Blk.Len_Extra_Delim]
31192                                                                       ;AN000; get length of delimiter list
31193                                       ;or     cx,cx                 ;AN000; No extra Delim character ?
31194                                       ;jz     short _$P_Chk_Delim_NZ        ;AN000;
31195                                       ; 08/07/2023
31196 0000213F E30B                         jcxz    _$P_Chk_Delim_NZ
31197
31198 00002141 BB0300                       mov     bx,_$P_Len_PARMS-1 ; 3;AN000; set bx to 1st extra delimiter
31199                                     _$P_Chk_Delim_Loop:                     ;AN000;
31200 00002144 43                           inc     bx                    ;AN000;
31201 00002145 263A01                       cmp     al,[es:bx+di]         ;AN000; Check extra Delim character
31202 00002148 7406                         je      short _$P_Chk_Delim_Exit0 ;AN000;
31203
31204 0000214A E2F8                         loop    _$P_Chk_Delim_Loop    ;AN000; examine all extra delimiter
31205
31206                                     _$P_Chk_Delim_NZ:                 ;AN000;
31207                                       ;cmp    al,_$P_Space          ;AN000; reset ZF
31208                                       ; 08/07/2023
31209                                       ; cx=0 here
31210 0000214C 41                           inc     cx ; cx=1, zf=0
31211                                     _$P_Chk_Delim_Exit:                     ;AN000;
31212                                     _$P_ChkDfin:                            ;AN000;
31213 0000214D 59                           pop     cx                    ;AN000;
31214 0000214E 5B                           pop     bx                    ;AN000;
31215 0000214F C3                           retn                          ;AN000;
31216
31217                                     _$P_Chk_Delim_Exit0:                    ;AN000;
31218 00002150 2EA2[7719]                   mov     [cs:_$P_Terminator],al ;AC034; keep terminated delimiter
31219 00002154 2EF606[7D19]01               test    byte [cs:_$P_Flags2],_$P_equ ;AN027;AC034;; if terminating a key=
31220 0000215A 7506                         jnz     short _$P_No_Set_Extra        ;AN027; then do not set the EXTRA bit
31221
31222 0000215C 2E800E[7D19]20               or      byte [cs:_$P_Flags2],_$P_Extra
31223                                                                       ;AC034; flag terminated extra delimiter or comma
31224                                     _$P_No_Set_Extra:                 ;AN027;
31225 00002162 38C0                         cmp     al,al                 ;AN000; set ZF
31226 00002164 EBE7                         jmp     short _$P_Chk_Delim_Exit ;AN000;
31227
```

```
                                ;****************************************************************************
                                ; _$P_Chk_Switch;
                                ;
                                ; Function: Check if AL is the switch character not in first position of
                                ;       _$P_STRING_BUF
                                ;
                                ; Input:    AL = character code
                                ;       BX = current pointer within _$P_String_Buf
                                ;       SI =>next char on command line (following the one in AL)
                                ;
                                ; Output:   CF = 1 (set)if AL is switch character, and not in first
                                ;            position, and has no chance of being part of a date string,
                                ;            i.e. should be treated as a delimiter.
                                ;
                                ;       CF = 0 (reset, cleared) if AL is not a switch char, is in the first
                                ;            position, or is a slash but may be part of a date string, i.e.
                                ;            should not be treated as a delimiter.
                                ;
                                ; Vars:  _$P_Terminator(W)

                                ; Use:       _$P_0099
                                ;****************************************************************************

                                _$P_Chk_Switch:
                                    ;;lea   bp,[cs:_$P_STRING_BUF];AN020;AC034
                                    ;lea    bp,[_$P_STRING_BUF]   ;BP=OFFSET of _$P_String_Buf even in group addressing
                                    ; 08/07/2023
00002166 BD[8619]                   mov     bp,_$P_STRING_BUF

                                ;   .IF <BX NE BP> THEN           ;AN020;IF not first char THEN
00002169 39EB                       cmp     bx,bp                 ;AN000;
0000216B 7406                       je      short _$P_STRUC_L2    ;AN000;

                                ;   .IF <AL EQ _$P_Switch> THEN   ;AN020;otherwise see if a slash
0000216D 3C2F                       cmp     al,_$P_Switch         ;AN000;
0000216F 750C                       jne     short _$P_STRUC_L5    ;AN000;

00002171 F9                         stc                           ;AN020;not in first position and is slash
                                    ;jmp     short _$P_STRUC_L1   ;AN000;
                                    ; 12/12/2022
00002172 C3                         retn

                                ; 12/12/2022
                                ;_$P_STRUC_L5:                          ;AN000;
                                ;   CLC                           ;AN020;not a slash
                                ;;      .ENDIF                    ;AN020;
                                ;;  .ELSE                         ;AN020;is first char in the buffer, ZF=0
                                ;   jmp     short _$P_STRUC_L1    ;AN000;

                                _$P_STRUC_L2:                           ;AN000;
                                ;   .IF <AL EQ _$P_Switch> THEN   ;AN020;
00002173 3C2F                       cmp     al,_$P_Switch         ;AN000;
00002175 7506                       jne     short _$P_STRUC_L12   ;AN000;

00002177 2E800E[7D19]40             or      byte [cs:_$P_Flags2],_$P_SW ;AN020 ;AC034;;could be valid switch, first char and is slash
                                ;   .ENDIF                        ;AN020;

                                    ; 12/12/2022
                                    ; cf=0
                                    ;retn

                                _$P_STRUC_L5:
                                    ; 12/12/2022
                                _$P_STRUC_L12:                          ;AN000;
0000217D F8                         clc                           ;AN020;CF=0 indicating first char
                                ;   .ENDIF                        ;AN020;
                                _$P_STRUC_L1:                           ;AN000;
0000217E C3                         retn                          ;AN000;

                                ;****************************************************************************
                                ; _$P_Chk_DBCS:
                                ;
                                ;  Function: Check if a specified byte is in ranges of the DBCS lead bytes
                                ;
                                ;  Input:
                                ;     AL    = Code to be examineed
                                ;
                                ;  Output:
                                ;     If CF is on then a lead byte of DBCS
                                ;
                                ; Use: INT 21h w/AH=63
                                ;
                                ; Vars:  _$P_DBCSEV_Seg(RW), _$P_DBCSEV_Off(RW)
                                ;****************************************************************************

                                _$P_Chk_DBCS:
0000217F 1E                         push    ds                    ;AN000;
00002180 56                         push    si                    ;AN000;
00002181 53                         push    bx                    ;AN000; (tm11)
                                    ;cmp    word [cs:_$P_DBCSEV_SEG],0 ;AC034; ALREADY SET ?
                                    ;jne    short _$P_DBCS00      ;AN000;
                                    ; 08/07/2023
00002182 2E8B36[7A19]               mov     si,[cs:_$P_DBCSEV_SEG]
00002187 21F6                       and     si,si ; 0 ?
00002189 7525                       jnz     short _$P_DBCS00 ; already set
0000218B 50                         push    ax                    ;AN000;
0000218C 1E                         push    ds                    ;AN000; (tm11)
0000218D 51                         push    cx                    ;AN000;
0000218E 52                         push    dx                    ;AN000;
0000218F 57                         push    di                    ;AN000;
00002190 55                         push    bp                    ;AN000;
00002191 06                         push    es                    ;AN000;
                                    ; si = 0 ; 08/07/2023
                                    ;xor    si,si                 ;AN000;
00002192 8EDE                       mov     ds,si ; 0             ;AN000;
00002194 B80063                     mov     ax,_$P_DOS_GetEV ; 6300h ;AN000; GET DBCS EV CALL
00002197 CD21                       int     21h                   ;AN000;
                                                                  ; DOS - 3.2+ only - GET DOUBLE BYTE CHARACTER SET LEAD TABLE
00002199 8CDB                       mov     bx,ds                 ;AN000; (tm11)
0000219B 09DB                       or      bx,bx                 ;AN000; (tm11)
0000219D 07                         pop     es                    ;AN000;
0000219E 5D                         pop     bp                    ;AN000;
0000219F 5F                         pop     di                    ;AN000;
000021A0 5A                         pop     dx                    ;AN000;
000021A1 59                         pop     cx                    ;AN000;
000021A2 1F                         pop     ds                    ;AN000; (tm11)
000021A3 58                         pop     ax                    ;AN000;
000021A4 7424                       jz      short _$P_NON_DBCS    ;AN000;
                                _$P_DBCS02:                       ;AN000;
000021A6 2E8936[7819]               mov     [cs:_$P_DBCSEV_OFF],si;AC034; save EV offset
000021AB 2E891E[7A19]               mov     [cs:_$P_DBCSEV_SEG],bx;AC034; save EV segment (tm11)
                                _$P_DBCS00:                       ;AN000;
                                    ;mov    si,[cs:_$P_DBCSEV_OFF];AC034; load EV offset
                                    ;mov    ds,[cs:_$P_DBCSEV_SEG];AC034; and segment
```

```
                                    ; 08/07/2023
000021B0 2EC536[7819]               lds     si,[cs:_$P_DBCSEV_OFF]
                                _$P_DBCS_LOOP:                          ;AN000;
000021B5 833C00                     cmp     word [si],0             ;AN000; zero vector ?
000021B8 7410                       je      short _$P_NON_DBCS      ;AN000; then exit
000021BA 3A04                       cmp     al,[si]                 ;AN000;
000021BC 7208                       jb      short _$P_DBCS01        ;AN000; Check if AL is in
000021BE 3A4401                     cmp     al,[si+1]               ;AN000;   range of
000021C1 7703                       ja      short _$P_DBCS01        ;AN000;      the vector
000021C3 F9                         stc                             ;AN000; if yes, indicate DBCS and exit
000021C4 EB04                       jmp     short _$P_DBCS_EXIT     ;AN000;
                                _$P_DBCS01:                             ;AN000;
000021C6 46                         inc     si                      ;AC035; add '2' to
000021C7 46                         inc     si                      ;AC035;  SI reg
                                                                    ;AN000; get next vector
000021C8 EBEB                       jmp     short _$P_DBCS_LOOP     ;AN000; loop until zero vector found
                                _$P_NON_DBCS:                           ;AN000;
                                    ; 12/12/2022
                                    ; cf=0
                                    ;clc                            ;AN000; indicate SBCS
                                _$P_DBCS_EXIT:                          ;AN000;
000021CA 5B                         pop     bx                      ;AN000; (tm11)
000021CB 5E                         pop     si                      ;AN000;
000021CC 1F                         pop     ds                      ;AN000;
000021CD C3                         retn                            ;AN000;

                                ; SYSCONF.ASM - MSDOS 6.0 - 1991
                                ; ============================================================================
                                ; 27/03/2019 - Retro DOS v4.0

                                ;control block definitions for parser.
                                ;----------------------------------------------------------------------------
                                ; buffer = [n | n,m] {/e}

                                ; 30/03/2019

                                struc p_parms
00000000 ????                       resw    1       ; dw ?
00000002 ??                         resb    1       ; db 1  ; an extra delimiter list
00000003 ??                         resb    1       ; db 1  ; length is 1
00000004 ??                         resb    1       ; db ';' ; delimiter
                                .size:
                                endstruc

                                struc p_pos
00000000 ????                       resw    1       ; dw ?  ; numeric value??
00000002 ????                       resw    1       ; dw ?  ; function
00000004 ????                       resw    1       ; dw ?  ; result value buffer

                                ; note: by defining result_val before this structure, we could remove
                                ;  the "result_val" from every structure invocation

00000006 ????                       resw    1       ; dw ?  ; value list
00000008 ??                         resb    1       ; db 0  ; no switches/keywords
                                .size:
                                endstruc

                                struc       p_range
00000000 ??                         resb    1       ; db 1  ; range definition
00000001 ??                         resb    1       ; db 1  ; 1 definition of range
00000002 ??                         resb    1       ; db 1  ; item tag for this range
00000003 ????????                   resd    1       ; dd ?  ; numeric min
00000007 ????????                   resd    1       ; dd ?  ; numeric max
                                .size:
                                endstruc

                                ;----------------------------------------------------------------------------

                                    ; 26/10/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 IO.SYS)
                                    ; (SYSINIT:1F48h)

                                    ; 08/07/2023 - Retro DOS v4.2 (Modified MSDOS 6.21 IO.SYS)
                                    ; MSDOS 6.21 IO.SYS - SYSINIT:2083h
                                    ; (PCDOS 7.1 IBMBIO.COM - SYSINIT:251Dh) ; (Retro DOS v5.0)

                                ; buffer = [n | n,m] {/e}

                                ;buf_parms p_parms <buf_parmsx>
                                buf_parms:
000021CE [D321]                     dw      buf_parmsx
000021D0 01                         db      1       ; an extra delimiter list
000021D1 01                         db      1       ; length is 1
000021D2 3B                         db      ';'     ; delimiter

                                buf_parmsx:
000021D3 0102[DD21][F121]           dw      201h,buf_pos1,buf_pos2; min 1, max 2 positionals
000021D9 01                         db      1                     ; one switch
000021DA [0522]                     dw      sw_x_ctrl
000021DC 00                         db      0                     ; no keywords

                                ;buf_pos1 p_pos <8000h,0,result_val,buf_range_1>  ; numeric
                                buf_pos1:
000021DD 0080                       dw      8000h   ; numeric value??
000021DF 0000                       dw      0       ; function
000021E1 [1722]                     dw      result_val ; result value buffer
000021E3 [E621]                     dw      buf_range_1 ; value list
000021E5 00                         db      0       ; no switches/keywords

                                ;buf_range_1 p_range <,,,1,99>           ; M050
                                buf_range_1:
000021E6 01                         db      1       ; range definition
000021E7 01                         db      1       ; 1 definition of range
000021E8 01                         db      1       ; item tag for this range
000021E9 01000000                   dd      1       ; numeric min
000021ED 63000000                   dd      99      ; numeric max

                                ;buf_pos2 p_pos <8001h,0,result_val,buf_range_2> ; optional num.
                                buf_pos2:
000021F1 0180                       dw      8001h
000021F3 0000                       dw      0
000021F5 [1722]                     dw      result_val
000021F7 [FA21]                     dw      buf_range_2
000021F9 00                         db      0

                                ;buf_range_2 p_range <,,,0,8>
                                buf_range_2:
000021FA 01                         db      1
000021FB 01                         db      1
000021FC 01                         db      1
000021FD 00000000                   dd      0
00002201 08000000                   dd      8

                                ;sw_x_ctrl p_pos <0,0,result_val,noval,1> ; followed by one switch
                                sw_x_ctrl:
```

```
                                    dw      0
                                    dw      0
                                    dw      result_val
                                    dw      noval
                                    db      1       ; 1 switch

                                switch_x:
                                    db      '/X',0          ; M016

                                p_buffers:
                                    dw      0       ; local variables
                                p_h_buffers:
                                    dw      0
                                    ; 26/10/2022  (MSDOS 5.0 IO.SYS SYSINIT compatibility)
                                p_buffer_slash_x:
                                    db      0 ; 31/03/2019

                                ;-- common definitions ---------------------------------------------

                                noval:      db      0

                                result_val:         ;label byte
                                    db      0               ; type returned
                                result_val_itag:
                                    db      0               ; item tag returned
                                result_val_swoff:
                                    dw      0               ; es:offset of the switch defined
                                rv_byte:    ;label byte
                                rv_dword: dd        0               ; value if number,or seg:offset to string.

                                ;-----------------------------------------------------------------------

                                    ; 27/10/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 IO.SYS)
                                    ; (SYSINIT:1F99h)

                                ; break = [ on | off ]

                                ;brk_parms p_parms  <brk_parmsx>
                                brk_parms:
                                    dw      brk_parmsx
                                    db      1       ; an extra delimiter list
                                    db      1       ; length is 1
                                    db      ';'     ; delimiter

                                brk_parmsx:
                                    dw      101h,brk_pos    ; min,max = 1 positional
                                    db      0               ; no switches
                                    db      0               ; no keywords

                                ;brk_pos p_pos <2000h,0,result_val,on_off_string> ; simple string
                                brk_pos:
                                    dw      2000h
                                    dw      0
                                    dw      result_val
                                    dw      on_off_string
                                    db      0

                                on_off_string:      ;label byte
                                    db      3               ; signals that there is a string choice
                                    db      0               ; no range definition
                                    db      0               ; no numeric values choice
                                    db      2               ; 2 strings for choice
                                    db      1               ; the 1st string tag
                                    dw      on_string
                                    db      2               ; the 2nd string tag
                                    dw      off_string

                                on_string:
                                    db      "ON",0
                                off_string:
                                    db      "OFF",0

                                p_ctrl_break:
                                    db      0       ; local variable

                                ;-----------------------------------------------------------------------

                                    ; 27/10/2022

                                ; country = n {m {path}}
                                ; or
                                ; country = n,,path

                                ;cntry_parms p_parms <cntry_parmsx>
                                cntry_parms:
                                    dw      cntry_parmsx
                                    db      1
                                    db      ';'

                                cntry_parmsx:
                                    dw      301h,cntry_pos1,cntry_pos2,cntry_pos3 ; min 1, max 3 pos.
                                    db      0               ; no switches
                                    db      0               ; no keywords

                                ;cntry_pos1 p_pos <8000h,0,result_val,cc_range> ; numeric value
                                cntry_pos1:
                                    dw      8000h
                                    dw      0
                                    dw      result_val
                                    dw      cc_range
                                    db      0

                                ;cc_range p_range <,,,1,999>
                                cc_range:
                                    db      1
                                    db      1
                                    db      1
                                    dd      1
                                    dd      999

                                ;cntry_pos2 p_pos <8001h,0,result_val,cc_range> ; optional num.
                                cntry_pos2:
                                    dw      8001h
                                    dw      0
                                    dw      result_val
                                    dw      cc_range
                                    db      0

                                ;cntry_pos3 p_pos <201h,0,result_val,noval>     ; optional filespec
                                cntry_pos3:
                                    dw      201h
                                    dw      0
```

```
                                    dw      result_val
                                    dw      noval
                                    db      0

                                p_cntry_code:
                                    dw      0       ; local variable
                                p_code_page:
                                    dw      0       ; local variable

                                ;--------------------------------------------------------------------------

                                    ; 27/10/2022

                                ; files = n

                                ;files_parms p_parms <files_parmsx>
                                files_parms:
                                    dw      files_parmsx
                                    db      1
                                    db      1
                                    db      ';'

                                files_parmsx:
                                    dw      101h,files_pos ; min,max 1 positional
                                    db      0              ; no switches
                                    db      0              ; no keywords

                                ;files_pos p_pos <8000h,0,result_val,files_range,0> ; numeric value
                                files_pos:
                                    dw      8000h
                                    dw      0
                                    dw      result_val
                                    dw      files_range
                                    db      0

                                ;files_range p_range <,,,8,255>
                                files_range:
                                    db      1
                                    db      1
                                    db      1
                                    dd      8
                                    dd      255

                                p_files:
                                    db      0               ; local variable

                                ;--------------------------------------------------------------------------

                                    ; 27/10/2022

                                ; fcbs = n,m

                                ;fcbs_parms p_parms <fcbs_parmsx>
                                fcbs_parms:
                                    dw      fcbs_parmsx
                                    db      1
                                    db      1
                                    db      ';'

                                fcbs_parmsx:
                                    dw      201h,fcbs_pos_1,fcbs_pos_2 ; min,max = 2 positional
                                    db      0               ; no switches
                                    db      0               ; no keywords

                                ;fcbs_pos_1 p_pos <8000h,0,result_val,fcbs_range> ; numeric value
                                fcbs_pos_1:
                                    dw      8000h
                                    dw      0
                                    dw      result_val
                                    dw      fcbs_range
                                    db      0

                                ;fcbs_range p_range <,,,1,255>
                                fcbs_range:
                                    db      1
                                    db      1
                                    db      1
                                    dd      1
                                    dd      255

                                ;fcbs_pos_2 p_pos <8000h,0,result_val,fcbs_keep_range> ; numeric value
                                fcbs_pos_2:
                                    dw      8000h
                                    dw      0
                                    dw      result_val
                                    dw      fcbs_keep_range
                                    db      0

                                ;fcbs_keep_range p_range <,,,0,255>
                                fcbs_keep_range:
                                    db      1
                                    db      1
                                    db      1
                                    dd      0
                                    dd      255

                                p_fcbs:     db      0               ; local variable
                                p_keep:     db      0               ; local variable

                                ;--------------------------------------------------------------------------

                                    ; 27/10/2022

                                ; lastdrive = x

                                ;ldrv_parms p_parms <ldrv_parmsx>
                                ldrv_parms:
                                    dw      ldrv_parmsx
                                    db      1
                                    db      1
                                    db      ';'

                                ldrv_parmsx:
                                    dw      101h,ldrv_pos  ; min,max = 1 positional
                                    db      0              ; no switches
                                    db      0              ; no keywords

                                ;ldrv_pos p_pos    <110h,10h,result_val,noval> ; drive only, ignore colon
                                ldrv_pos:                               ; remove colon at end
                                    dw      110h
                                    dw      10h
                                    dw      result_val
                                    dw      noval
                                    db      0
```

```
                                
000022E9 00                     p_ldrv:     db      0               ; local variable
                                
                                ;--------------------------------------------------------------------------
                                
                                    ; 27/10/2022
                                
                                ; stacks = n,m
                                
                                ;stks_parms p_parms <stks_parmsx>
                                stks_parms:
000022EA [EF22]                     dw      stks_parmsx
000022EC 01                         db      1
000022ED 01                         db      1
000022EE 3B                         db      ';'
                                
                                stks_parmsx:
000022EF 0202[F722][0B23]           dw      202h,stks_pos_1,stks_pos_2 ; min,max = 2 positionals
000022F5 00                         db      0               ; no switches
000022F6 00                         db      0               ; no keywords
                                
                                ;stks_pos_1 p_pos <8000h,0,result_val,stks_range> ; numeric value
                                stks_pos_1:
000022F7 0080                       dw      8000h
000022F9 0000                       dw      0
000022FB [1722]                     dw      result_val
000022FD [0023]                     dw      stks_range
000022FF 00                         db      0
                                
                                ;stks_range p_range <,,,0,64>
                                stks_range:
00002300 01                         db      1
00002301 01                         db      1
00002302 01                         db      1
00002303 00000000                   dd      0
00002307 40000000                   dd      64
                                
                                ;stks_pos_2 p_pos <8000h,0,result_val,stk_size_range> ; numeric value
                                stks_pos_2:
0000230B 0080                       dw      8000h
0000230D 0000                       dw      0
0000230F [1722]                     dw      result_val
00002311 [1423]                     dw      stk_size_range
00002313 00                         db      0
                                
                                ;stk_size_range p_range <,,,0,512>
                                stk_size_range:
00002314 01                         db      1
00002315 01                         db      1
00002316 01                         db      1
00002317 00000000                   dd      0
0000231B 00020000                   dd      512
                                
                                p_stack_count:
0000231F 0000                       dw      0       ; local variable
                                p_stack_size:
00002321 0000                       dw      0       ; local variable
                                
                                ;--------------------------------------------------------------------------
                                
                                    ; 27/10/2022
                                
                                ; multitrack = [ on | off ]
                                
                                ;mtrk_parms p_parms <mtrk_parmsx>
                                mtrk_parms:
00002323 [2823]                     dw      mtrk_parmsx
00002325 01                         db      1
00002326 01                         db      1
00002327 3B                         db      ';'
                                
                                mtrk_parmsx:
00002328 0101[2E23]                 dw      101h,mtrk_pos  ; min,max = 1 positional
0000232C 00                         db      0               ; no switches
0000232D 00                         db      0               ; no keywords
                                
                                ;mtrk_pos p_pos <2000h,0,result_val,on_off_string> ; simple string
                                mtrk_pos:
0000232E 0020                       dw      2000h
00002330 0000                       dw      0
00002332 [1722]                     dw      result_val
00002334 [3322]                     dw      on_off_string
00002336 00                         db      0
                                
00002337 00                     p_mtrk:     db      0               ; local variable
                                
                                ;--------------------------------------------------------------------------
                                
                                    ; 27/10/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 IO.SYS)
                                    ; (SYSINIT:20B2h)
                                
                                ; switches=/k
                                
                                ;swit_parms p_parms <swit_parmsx>
                                swit_parms:
00002338 [3D23]                     dw      swit_parmsx
0000233A 01                         db      1
0000233B 01                         db      1
0000233C 3B                         db      ';'
                                
                                swit_parmsx:
0000233D 0000                       dw      0               ; no positionals
                                    ; 08/07/2023 - Retro DOS v4.2 (MSDOS 6.21 IO.SYS, SYSINIT)
                                    ;db     5                 ; # of switches
                                    ; 14/04/2024 - Retro DOS v5.0 (PCDOS 7.1 IBMBIO.COM, SYSINIT)
0000233F 06                         db      6
                                    ; 27/10/2022 - Retro DOS v4.0 (MSDOS 5.0 IO.SYS, SYSINIT)
                                    ;db     3
00002340 [4D23]                     dw      swit_k_ctrl    ; /k control
                                    ; 01/01/2023 - Retro DOS v4.2 ; *
00002342 [5923]                     dw      swit_n_ctrl ; * ; /n control (for MULTI_CONFIG only)
00002344 [6523]                     dw      swit_f_ctrl ; * ; /f control (for MULTI_CONFIG only)
00002346 [7123]                     dw      swit_t_ctrl     ; /t control
00002348 [7D23]                     dw      swit_w_ctrl     ; /w control
                                    ; 14/04/2024 - Retro DOS v5.0 ; **
0000234A [8923]                     dw      swit_i_ctrl    ; /i control
0000234C 00                         db      0               ; no keywords
                                
                                ;swit_k_ctrl p_pos <0,0,result_val,noval,1> ; switch string follows
                                swit_k_ctrl:
0000234D 00000000[1722]-            dw      0,0,result_val,noval
00002353 [1622]
00002355 01                         db      1
00002356 2F4B00                 swit_k:     db      '/K',0
```

```
                                ; 01/01/2023 - Retro DOS v4.2 (MSDOS 6.21 IO.SYS)
                                ; (SYSINIT:220Ch) ; *

                                ; 27/10/2022 - Retro DOS v4.0 (MSDOS 5.0 IO.SYS, SYSINIT)
                                ;
                                ;swit_n_ctrl p_pos <0,0,result_val,noval,1> ; switch string follows
                                swit_n_ctrl: ; *
00002359 00000000[1722]-            dw      0,0,result_val,noval
0000235F [1622]
00002361 01                         db      1
00002362 2F4E00                 swit_n: db  '/N',0

                                ;swit_f_ctrl p_pos <0,0,result_val,noval,1> ; switch string follows
                                swit_f_ctrl: ; *
00002365 00000000[1722]-            dw      0,0,result_val,noval
0000236B [1622]
0000236D 01                         db      1
0000236E 2F4600                 swit_f: db  '/F',0

                                    ; 27/10/2022

                                ;swit_t_ctrl p_pos <0,0,result_val,noval,1> ; switch string follows   M059
                                swit_t_ctrl:
00002371 00000000[1722]-            dw      0,0,result_val,noval
00002377 [1622]
00002379 01                         db      1
0000237A 2F5400                 swit_t:     db      '/T',0                  ;                       M059
                                ;swit_w_ctrl p_pos <0,0,result_val,noval,1> ; switch string follows   M063
                                swit_w_ctrl:
0000237D 00000000[1722]-            dw      0,0,result_val,noval
00002383 [1622]
00002385 01                         db      1
00002386 2F5700                 swit_w:     db      '/W',0                  ;                       M063

                                ; 14/04/2024 - Retro DOS v5.0 (PCDOS 7.1 IBMBIO.COM, SYSINIT)
                                ;;;
                                ;swit_i_ctrl p_pos <0,0,result_val,noval,1> ; switch string follows
                                swit_i_ctrl:
00002389 0000                       dw      0
0000238B 0000                       dw      0
0000238D [1722]                     dw      result_val
0000238F [1622]                     dw      noval
00002391 01                         db      1
00002392 2F4900                 swit_i:     db      '/I',0
                                ;;;

                                ;   There doesn't need to be p_swit_n or p_swit_f because /N and /F are
                                ;   acted upon during MULTI_CONFIG processing; we only needed entries
                                ;   in the above table to prevent the parsing code from complaining about them

00002395 00                     p_swit_k:   db      0       ; local variable
00002396 00                     p_swit_t:   db      0       ; local variable                    M059
00002397 00                     p_swit_w:   db      0       ; local variable                    M063
                                ; 14/04/2024 - Retro DOS v5.0 (PCDOS 7.1 IBMBIO.COM)
00002398 00                     p_swit_i:   db      0

                                ;--------------------------------------------------------------------------

                                    ; 27/10/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 IO.SYS)
                                    ; (SYSINIT:20E8h)

                                ; DOS = [ high | low ]

                                ;dos_parms p_parms  <dos_parmsx>
                                dos_parms:
00002399 [9E23]                     dw      dos_parmsx
0000239B 01                         db      1
0000239C 01                         db      1
0000239D 3B                         db      ';'
                                dos_parmsx:
0000239E 01                         db      1               ; min parameters
0000239F 02                         db      2               ; max parameters
000023A0 [A623]                     dw      dos_pos         ;
000023A2 [A623]                     dw      dos_pos         ;
000023A4 00                         db      0               ; no switches
000023A5 00                         db      0               ; no keywords

                                ;dos_pos p_pos     <2000h,0,result_val,dos_strings> ; simple string
                                ;       p_pos     <2000h,0,result_val,dos_strings> ; simple string
                                dos_pos:
000023A6 00200000[1722]-            dw      2000h,0,result_val,dos_strings
000023AC [B823]
000023AE 00                         db      0
000023AF 00200000[1722]-            dw      2000h,0,result_val,dos_strings
000023B5 [B823]
000023B7 00                         db      0

                                dos_strings:        ;label  byte
000023B8 03                         db      3               ; signals that there is a string choice
000023B9 00                         db      0               ; no range definition
000023BA 00                         db      0               ; no numeric values choice
000023BB 04                         db      4               ; 4 strings for choice
000023BC 01                         db      1               ; the 1st string tag
000023BD [E623]                     dw      hi_string
000023BF 02                         db      2               ; the 2nd string tag
000023C0 [EB23]                     dw      lo_string
000023C2 03                         db      3
000023C3 [EF23]                     dw      umb_string
000023C5 04                         db      4
000023C6 [F323]                     dw      noumb_string

                                ; 14/04/2024 - Retro DOS v5.0
                                ; (PCDOS 7.1 IBMDOS.COM - SYSINIT:273Eh)
                                ;;;
                                dosdata_parms:
000023C8 [CD23]                     dw      dosdata_parmsx ; DOSDATA = UMB|NOUMB
000023CA 01                         db      1
000023CB 01                         db      1
000023CC 3B                         db      ';'
                                dosdata_parmsx:
000023CD 01                         db      1
000023CE 01                         db      1               ; min,max = 1 positional
000023CF [D323]                     dw      dosdata_pos
000023D1 00                         db      0               ; no switches
000023D2 00                         db      0               ; no keywords

                                    ; dosdata_pos p_pos <2000h,0,result_val,dosdata_strings>
                                dosdata_pos:
000023D3 0020                       dw      2000h           ; simple string
000023D5 0000                       dw      0
000023D7 [1722]                     dw      result_val
000023D9 [DC23]                     dw      dosdata_strings
000023DB 00                         db      0
```

```
                                dosdata_strings:
                                    db      3               ; signals that there is a string choice
                                    db      0               ; no range definition
                                    db      0               ; no numeric values choice
                                    db      2               ; 2 strings for choice
                                    db      1               ; the 1st string tag
000023E1 [EF23]                     dw      umb_string      ; "UMB"
                                    db      2               ; the 2nd string tag
000023E4 [F323]                     dw      noumb_string    ; "NOUMB"
                                ;;;

000023E6 4849474800             hi_string:  db      "HIGH",0
000023EB 4C4F5700               lo_string:  db      "LOW",0
000023EF 554D4200               umb_string: db      "UMB",0
000023F3 4E4F554D4200           noumb_string:       db      "NOUMB",0

                                p_dos_hi:
000023F9 00                         db      0       ; local variable
                                                    ; BUGBUG : I dont know whether PARSER uses
                                                    ;          this variable or not
                                    ; 14/04/2024 (PCDOS 7.1 IBMBIO.COM)
000023FA 00                         db      0

                                ; 01/01/2023 - Retro DOS v4.2 (Modified MSDOS 6.21 IO.SYS)
                                ; 27/10/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 IO.SYS)
                                ;%if 0

                                ;******************************************************************** RICHID ****

                                ;include    highvar.inc    ; devicehigh variables (used by loadhigh also)

                                ; 30/03/2019 - Retro DOS v4.0
                                ;--------------------------------------------------------------------------------

                                ;   Module:   HIGHVAR.INC - Data common to LOADHIGH and DEVICEHIGH, res seg
                                ;
                                ;   Date:     May 14, 1992
                                ;
                                ;*******************************************************************************
                                ;
                                ;   Modification log:
                                ;
                                ;     DATE    WHO      DESCRIPTION
                                ;   --------  -------  -------------------------------------------------------
                                ;   05/14/92  t-richj  Original
                                ;   06/21/92  t-richj  Final revisions before check-in
                                ;
                                ;*******************************************************************************
                                ;
                                ; There are two primary definitions which need to be made, selectively, before
                                ; this include file should be used.  These are:
                                ;    HV_Extern - If this has been defined, variables for this module will be
                                ;                declared as external.  Otherwise, variables will be declared
                                ;                public, as well as defined, here.  LoadHigh declares HV_Extern
                                ;                in stub.asm and loadhi.asm, and does not declare it in
                                ;                rdata.asm... DeviceHigh does not declare HV_Extern anywhere
                                ;                (as only one module, sysconf.asm, includes this file).
                                ;    HV_LoadHigh - This should be defined when this module is going into
                                ;                  command.com, for LoadHigh.  All of loadhi.asm, stub.asm and
                                ;                  rdata.asm define this, while io.sys' sysconf.asm does not.
                                ;
                                ;*******************************************************************************

                                ; To keep track of which UMBs were specified on the DH/LH command lines, and
                                ; to keep track of the minimum sizes given for each, there're two arrays kept
                                ; in { IO.SYS: sysinitseg / COMMAND.COM: DATARES }... each is MAXUMB elements
                                ; big.  16 should be around 14 too many for most users, so there's no expected
                                ; space problem (it's just such a nice round number, eh?).

                                MAXUMB      equ     16

                                ; Memory elements owned by the system are marked as PSP address 8 in both the
                                ; USA and Japan; Japanese systems also use 9 under more bizzarre conditions.

                                FreePSPOwner        equ     0       ; Free MCBs all have an owner PSP address of 0
                                SystemPSPOwner      equ     8
                                ;JapanPSPOwner      equ     9

                                ; for LoadHigh and DeviceHigh:
                                ;
                                ;   fInHigh - Is set to 1 during HideUMBs(), and back to zero in
                                ;             UnHideUMBs().
                                ;   fUmbTiny - Is set to 1 iff the user has specified /S on the command-
                                ;              line.
                                ;   SegLoad - Segment address for first UMB specified; set automatically.
                                ;   UmbLoad - The load UMB number; for example, this is 3 if the user has
                                ;             given a command-line like "/L:3,500;4"
                                ;   UmbUsed - An array of characters, each of which is 1 iff the UMB
                                ;             matching its index number was specified on the command-line;
                                ;             for example, after "/L:3,500;4;7", UmbUsed[3], [4] and [7]
                                ;             will be set to 1.  All others will be set to 0.
                                ;   UmbSize - An array of words, each of which is interpereted as a size
                                ;             specified by the user for a UMB (in the above example, all
                                ;             elements would be zero save UmbSize[3], which would be 500.
                                ;   fm_umb - Set to the old UMB link-state (0x80 or 0x00)
                                ;   fm_strat - Set to the old memory-allocation strategy (0$00000???)
                                ;   fm_argc  - Number of arguments received by ParseVar() (see ParseVar()
                                ;              for details).

000023FB 00                     fInHigh:  db        0
000023FC 00                     fUmbTiny: db        0
000023FD 0000                   SegLoad:  dw        0
000023FF 00                     UmbLoad:  db        0
00002400 00<rep 10h>            UmbUsed:  times MAXUMB db 0 ; times 16 db 0  ; db 16 dup(?)
00002410 0000<rep 10h>          UmbSize:  times MAXUMB dw 0 ; times 16 dw 0  ; dw 16 dup(?)
00002430 00                     fm_umb:   db        0
00002431 00                     fm_strat: db        0
00002432 00                     fm_argc:  db        0

                                ; UmbLoad is set to UNSPECIFED, below, until /L:umb is read; at which point
                                ; UmbLoad is set to the UMB number given.

                                UNSPECIFIED equ     -1

                                ;%endif ; 27/10/2022

                                ;******************************************************************** RICHID ****

                                ; 30/03/2019 - Retro DOS v4.0 (MSDOS 6.0, SYSCONF.ASM)
                                ; ((MSDOS 6.21 IO.SYS -> SYNINIT:22BAh))

                                ; 27/10/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 IO.SYS)
                                ; (SYSINIT:212Bh)
```

```
32088                                           ;public DevEntry
32089
32090 00002433 0000                     DevSize:    dw      0       ; size of the device driver being loaded(paras)
32091 00002435 0000                     DevLoadAddr:        dw      0       ; Mem addr where the device driver is 2 b loaded
32092 00002437 0000                     DevLoadEnd: dw      0       ; MaxAddr to which device can be loaded
32093 00002439 00000000                 DevEntry:   dd      0       ; Entry point to the device driver
32094 0000243D 00000000                 DevBrkAddr: dd      0       ; Break address of the device driver
32095                                   ; 30/12/2022
32096                                   ; 27/10/2022
32097 00002441 00                       ConvLoad:   db      0       ; Use conventional (dos 5 -style) InitDevLoad?
32098                                   ;
32099 00002442 00                       DevUMB:             db      0       ; byte indicating whether to load DDs in UMBs
32100 00002443 0000                     DevUMBAddr: dw      0       ; cuurent UMB used fro loading devices (paras)
32101 00002445 0000                     DevUMBSize: dw      0       ; Size of the current UMB being used   (paras)
32102 00002447 0000                     DevUMBFree: dw      0       ; Start of free are in the current UMB (paras)
32103                                   ;
32104 00002449 00000000                 DevXMSAddr: dd      0
32105                                   ;
32106 0000244D 0000                     DevExecAddr:        dw      0       ; Device load address parameter to Exec call
32107 0000244F 0000                     DevExecReloc:       dw      0       ; Device load relocation factor
32108                                   ;
32109 00002451 00                       DeviceHi:   db      0       ; Flag indicating whther the current device
32110                                                                   ;  is being loaded into UMB
32111 00002452 0000                     DevSizeOption:      dw      0       ; SIZE= option
32112                                   ;
32113 00002454 00                       Int12Lied:  db      0       ; did we trap int 12 ?
32114 00002455 0000                     OldInt12Mem:        dw      0       ; value in 40:13h (int 12 ram)
32115 00002457 50524F544D414E24         ThreeComName:       db      'PROTMAN$'      ; 3Com Device name
32116                                   ;
32117 0000245F 00                       FirstUMBLinked:     db      0
32118 00002460 0000                     DevDOSData: dw      0       ; segment of DOS Data
32119 00002462 00000000                 DevCmdLine: dd      0       ; Current Command line
32120 00002466 00                       DevSavedDelim:      db      0       ; The delimiter which was replaced with null
32121                                                                   ; to use the file name in the command line
32122                                   ; 13/04/2024 - Retro DOS v5.0 (PCDOS 7.1 IBMBIO.COM)
32123                                   ;   ifdef  dblspace_hooks
32124 00002467 00                       MagicHomeFlag:      db      0       ; set non-zero when MagicDrv is final placed
32125                                   ;   endif
32126
32127                                   ; ================================================================================
32128
32129                                   ; 31/03/2019 - Retro DOS v4.0
32130
32131                                   ; 27/10/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 IO.SYS)
32132                                   ; (SYSINIT:215Eh)
32133
32134                                   ;--------------------------------------------------------------------------
32135                                   ;
32136                                   ; procedure : doconf
32137                                   ;
32138                                   ;             Config file is parsed initially with this routine. For the
32139                                   ;             Subsequent passes 'multi_pass' entry is used .
32140                                   ;
32141                                   ;--------------------------------------------------------------------------
32142
32143                                       ; 27/10/2022
32144                                   doconf:
32145 00002468 0E                           push    cs
32146 00002469 1F                           pop     ds
32147
32148 0000246A B80037                       mov     ax,3700h
32149                                           ;mov       ax,(CHAR_OPER<<8)     ; get switch character
32150 0000246D CD21                         int     21h
32151 0000246F 8816[BC4B]                   mov     [command_line+1],dl   ; set in default command line
32152
32153                                   ; 31/12/2022 - Retro DOS v4.2 (Modified MSDOS 6.21  IO.SYS)
32154                                   ; 27/10/2022
32155                                   ;;ifdef     MULTI_CONFIG
32156                                   ;   ;mov    [command_line-1],dl     ; save default switchchar
32157 00002473 8816[BA4B]                   mov     [def_swchr],dl ; 31/03/2019
32158                                   ;;endif     ;MULTI_CONFIG
32159
32160 00002477 BA[D14A]                     mov     dx,config ;'\CONFIG.SYS' ;now pointing to file description
32161 0000247A B8003D                       mov     ax,3D00h
32162                                       ;mov    ax,OPEN<<8              ;open file "config.sys"
32163 0000247D F9                           stc                             ;in case of int 24
32164 0000247E CD21                         int     21h                     ;function request
32165 00002480 7309                         jnc     short noprob            ; brif opened okay
32166
32167                                   ; 31/12/2022
32168                                   ; 27/10/2022
32169                                   ;;ifdef     MULTI_CONFIG
32170 00002482 E8A019                       call    kbd_read                ; we still want to give the guy
32171                                   ;                                   ; a chance to select clean boot!
32172                                   ;;endif                                     ; (ie, no autoexec.bat processing)
32173 00002485 C606[CD02]0B                 mov     byte [multi_pass_id],11     ; set it to unreasonable number
32174 0000248A C3                           retn
32175                                   noprob:                             ;get file size (note < 64k!!)
32176 0000248B 89C3                         mov     bx,ax  ; File handle
32177 0000248D 31C9                         xor     cx,cx                   ; 0
32178 0000248F 31D2                         xor     dx,dx                   ; 0
32179                                       ;mov    ax,4202h
32180 00002491 B80242                       mov     ax,(LSEEK<<8)|2
32181 00002494 CD21                         int     21h
32182 00002496 A3[5603]                     mov     [count],ax              ; dx:ax = file size ; 08/09/2023
32183                                                                       ; 08/09/2023 - Erdogan Tan - Note:
32184 00002499 31D2                         xor     dx,dx                   ; dx already must be 0 here ; 08/09/2023
32185                                                                       ; I am not removing 'xor dx,dx' here
32186                                                                       ; for MSDOS compatibility.
32187                                                                       ; ((Also PCDOS 7.1 has 'xor dx,dx' here))
32188                                                                       ; (Error will be same if CONGIG.SYS file
32189                                                                       ;  size > 64KB)
32190                                       ;mov    ax,4200h
32191 0000249B B80042                       mov     ax,LSEEK<<8             ;reset pointer to beginning of file
32192 0000249E CD21                         int     21h
32193
32194                                       ; 31/12/2022 - Retro DOS v4.2
32195 000024A0 8B16[A502]                   mov     dx,[ALLOCLIM]           ;use current alloclim value
32196                                       ; 27/10/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
32197                                       ;mov    dx,[top_of_cdss]
32198
32199 000024A4 A1[5603]                     mov     ax,[count]
32200 000024A7 A3[D002]                     mov     [config_size],ax        ;save the size of config.sys file.
32201 000024AA E868EE                       call    ParaRound
32202 000024AD 29C2                         sub     dx,ax
32203
32204                                   ; 31/12/2022
32205                                   ; 27/10/2022
32206                                   ;ifdef      MULTI_CONFIG
32207                                   ;
32208                                   ;  The size of the CONFIG.SYS workspace (for recreating the in-memory
32209                                   ;  CONFIG.SYS image, and later for building the initial environment) need
32210                                   ;  not be any larger than CONFIG.SYS itself, EXCEPT for the fact that
32211                                   ;  we (may) add a variable to the environment that does not explicity appear
```

```
                                ;  in CONFIG.SYS, and that variable is CONFIG (as in CONFIG=COMMON).
                                ;  The default setting for CONFIG cannot result in more than 1 paragraph
                                ;  of extra space, so here we account for it (the worst case of course is
                                ;  when CONFIG.SYS is some very small size, like 0 -JTP)
                                ;
000024AF 4A                         dec     dx                      ;reserve 1 additional paragraph
000024B0 8916[6219]                 mov     [config_wrkseg],dx      ;this is the segment to be used for
000024B4 29C2                       sub     dx,ax                   ;rebuilding the config.sys memory image
                                ;;endif    ;MULTI_CONFIG

000024B6 83EA11                     sub     dx,11h                  ;room for header

                                    ; 31/12/2022
000024B9 8916[A502]                 mov     [ALLOCLIM],dx           ;config starts here. new alloclim value.
000024BD 8916[A302]                 mov     [CONFBOT],dx

                                    ; 27/10/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
                                    ;mov    [top_of_cdss],dx
                                    ;call    TempCDS
                                    ; 31/12/2022
                                    ; 11/12/2022
                                    ; ds <> cs
                                    ;mov    dx,[cs:top_of_cdss]

                                    ; 08/09/2023
                                    ; ds = cs
000024C1 8B0E[5603]                 mov     cx,[count]

000024C5 8EDA                       mov     ds,dx
000024C7 8EC2                       mov     es,dx

000024C9 31D2                       xor     dx,dx
                                    ; 08/09/2023
                                    ;mov    cx,[cs:count]
000024CB B43F                       mov     ah,3Fh
                                    ;mov    ah,READ  ; 3Fh
000024CD F9                         stc                             ;in case of int 24
000024CE CD21                       int     21h                     ;function request
000024D0 9C                         pushf

                                ; find the eof mark in the file. if present,then trim length.

000024D1 50                         push    ax
000024D2 57                         push    di
000024D3 51                         push    cx
000024D4 B01A                       mov     al,1Ah                  ; eof mark
000024D6 89D7                       mov     di,dx                   ; point to buffer
000024D8 E305                       jcxz    puteol                  ; no chars
000024DA F2AE                       repnz   scasb                   ; find end
000024DC 7501                       jnz     short puteol            ; none found and count exhausted

                                ; we found a 1a. back up

000024DE 4F                         dec     di                      ; backup past 1Ah

                                ;  just for the halibut, stick in an extra eol

                                puteol:
000024DF B00D                       mov     al,cr ; 0Dh
000024E1 AA                         stosb
000024E2 B00A                       mov     al,lf  ;0Ah
000024E4 AA                         stosb
000024E5 29D7                       sub     di,dx                   ; difference moved
                                    ; 27/10/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
                                    ;mov    [cs:count],di           ; new count

                                    ; 11/12/2022
                                    ; 31/03/2019 - Retro DOS v4.0
000024E7 0E                         push    cs
000024E8 1F                         pop     ds

000024E9 893E[5603]                 mov     [count],di              ; new count

000024ED 59                         pop     cx
000024EE 5F                         pop     di
000024EF 58                         pop     ax

                                    ; 11/12/2022
                                    ; 27/10/2022
                                    ;push   cs
                                    ;pop    ds

000024F0 50                         push    ax
                                    ;mov    ah,CLOSE
000024F1 B43E                       mov     ah,3Eh
000024F3 CD21                       int     21h
000024F5 58                         pop     ax
000024F6 9D                         popf
000024F7 7204                       jc      short conferr           ;if not we've got a problem
000024F9 39C1                       cmp     cx,ax
000024FB 742C                       jz      short getcom            ;couldn't read the file
                                conferr:
000024FD BA[D14A]                   mov     dx,config               ;want to print config error
00002500 E82525                     call    badfil
                                    ; 14/04/2024
                                endconv:    ; 01/01/2023
00002503 C3                         retn

                                ;--------------------------------------------------------------------------
                                ;
                                ; entry : multi_pass
                                ;
                                ;             called to execute device=,install= commands
                                ;
                                ;--------------------------------------------------------------------------

                                    ; 27/10/2022
                                multi_pass:
00002504 0E                         push    cs
00002505 1F                         pop     ds

00002506 803E[CD02]0A               cmp     byte [multi_pass_id],10
                                ;jae_endconv:
0000250B 73F6                       jae     short endconv           ; do nothing. just return.

                                    ; 31/12/2022 - Retro DOS v4.2 (Modified MSDOS 6.21 IO.SYS)
0000250D FF36[A302]                 push    word [CONFBOT]
                                    ; 27/10/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
                                    ;push   word [top_of_cdss]
00002511 07                         pop     es                      ; es -> confbot

00002512 8B36[5803]                 mov     si,[org_count]
00002516 8936[5603]                 mov     [count],si              ; set count
0000251A 31F6                       xor     si,si ; 0
```

```
32336 0000251C 8936[5A03]                       mov     [chrptr],si             ; reset chrptr
32337 00002520 8936[AF02]                       mov     [linecount],si          ; reset linecount
32338
32339 00002524 E88822                       call    getchr
32340 00002527 EB06                         jmp     short conflp
32341
32342                                       ; 14/04/2024
32343                                       ; 01/01/2023
32344                                     ;endconv:
32345                                       ;retn
32346
32347                                     getcom:
32348                                       ; 03/01/2023
32349                                       ; ds = cs
32350 00002529 E8C016                           call    organize                ; organize the file
32351 0000252C E88022                       call    getchr
32352                                     conflp:
32353 0000252F 72D2                         jc      short endconv
32354
32355 00002531 FF06[AF02]                       inc     word [linecount]  ; increase linecount
32356
32357                                       ; 08/09/2023
32358 00002535 30E4                         xor     ah,ah ; 0
32359                                       ;mov    byte [multdeviceflag],0       ; reset multdeviceflag.
32360                                       ;mov    byte [setdevmarkflag],0       ; reset setdevmarkflag.
32361 00002537 8826[6619]                   mov     [multdeviceflag],ah ; 0
32362 0000253B 8826[6919]                   mov     [setdevmarkflag],ah ; 0
32363
32364 0000253F 3C0A                         cmp     al,lf                   ; linefeed?
32365 00002541 7448                         je      short blank_line        ;  then ignore this line.
32366
32367                                     ; 31/12/2022 - Retro DOS v4.2 (Modified MSDOS 6.21 IO.SYS)
32368                                     ; (SYSINIT:23CCh)
32369                                     ; 27/10/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
32370                                     ;%if 0
32371
32372                                     ;ifdef      MULTI_CONFIG
32373
32374                                     ;   If this is a genuine CONFIG.SYS command, then there should be a line
32375                                     ;   number immediately following it....
32376
32377 00002543 A2[6419]                       mov     [config_cmd],al         ; save original command code
32378                                       ;and    al,NOT CONFIG_OPTION_QUERY
32379 00002546 247F                         and     al,~CONFIG_OPTION_QUERY ; and al,7Fh
32380
32381                                       ; 08/09/2023
32382 00002548 3826[6519]                   cmp     [config_multi],ah ; 0
32383                                       ;cmp    byte [config_multi],0 ; is this a multi-config config.sys?
32384 0000254C 7427                             je      short not_final         ; no, line number is not embedded
32385
32386 0000254E 50                               push    ax                      ;
32387 0000254F E85D22                           call    getchr                  ; ignore end-of-image errors,
32388 00002552 88C4                             mov     ah,al                   ; because if there's an error
32389 00002554 E85822                           call    getchr                  ; fetching the line number that's
32390 00002557 86C4                             xchg    al,ah                   ; supposed to be there, the next
32391 00002559 A3[AF02]                         mov     [linecount],ax          ; getchr call will get the same error
32392 0000255C 58                               pop     ax
32393                                     ;
32394                                     ;   HACK: when 4DOS.COM is the shell and it doesn't have an environment from
32395                                     ;   which to obtain its original program name, it grovels through all of
32396                                     ;   memory to find the filename that was used to exec it; it wants to find
32397                                     ;   the SHELL= line in the in-memory copy of CONFIG.SYS, and it knows that
32398                                     ;   sysinit converts the SHELL= keyword to an 'S', so it expects to find an 'S'
32399                                     ;   immediately before the filename, but since we are now storing line # info
32400                                     ;   in the config.sys memory image, 4DOS fails to find the 'S' in the right
32401                                     ;   spot.
32402                                     ;
32403                                     ;   So, on the final pass of CONFIG.SYS, copy the command code (eg, 'S')
32404                                     ;   over the line number info, since we no longer need that info anyway. This
32405                                     ;   relies on the fact that getchr leaves ES:SI pointing to the last byte
32406                                     ;   retrieved.
32407                                     ;
32408 0000255D 803E[CD02]02                     cmp byte [multi_pass_id],2; final pass?
32409 00002562 7211                             jb  short not_final             ; no
32410                                           ;test      word [install_flag],have_install_cmd
32411 00002564 F606[CE02]01                 test    byte [install_flag],have_install_cmd ; 1
32412 00002569 7407                             jz  short final             ; no install cmds, so yes it is
32413 0000256B 803E[CD02]03                     cmp byte [multi_pass_id],3; final pass?
32414 00002570 7203                             jb  short not_final             ; no
32415                                     final:                                  ;
32416 00002572 268804                       mov     [es:si],al              ; save backward-compatible command code
32417                                     not_final:                              ;
32418                                     ;endif
32419
32420                                     ; 31/12/2022
32421                                     ;%endif ; 27/10/2022
32422
32423 00002575 88C4                         mov     ah,al
32424 00002577 E83522                       call    getchr
32425 0000257A 7314                         jnc     short tryi
32426
32427 0000257C 803E[CD02]02                 cmp     byte [multi_pass_id],2
32428                                       ;jae    short jae_endconv       ; do not show badop again for multi_pass.
32429                                       ; 27/10/2022
32430 00002581 7380                         jnb     short endconv
32431 00002583 E90009                       jmp     badop
32432
32433                                     coff:
32434                                       ; 11/12/2022
32435                                       ; ds = cs
32436                                       ;push   cs
32437                                       ;pop    ds
32438 00002586 E81D22                       call    newline
32439 00002589 EBA4                         jmp     short conflp   ; 13/05/2019
32440
32441                                     blank_line:
32442 0000258B E82122                       call    getchr
32443 0000258E EB9F                         jmp     short conflp
32444
32445                                       ; 27/10/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
32446                                     ; 11/12/2022
32447                                     ; (there is not a jump or call to here from anywhere!)
32448                                     ;coff_p:
32449                                       ;push   cs
32450                                       ;pop    ds
32451
32452                                     ;to handle install= commands,we are going to use multi-pass.
32453                                     ;the first pass handles the other commands and only set install_flag when
32454                                     ;it finds any install command. the second pass will only handle the
32455                                     ;install= command.
32456
32457                                     ;--------------------------------------------------------------------------
32458                                     ;install command
32459                                     ;--------------------------------------------------------------------------
```

```
32460
32461                                   ; 27/10/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 IO.SYS)
32462                                   ; (SYSINIT:2250h)
32463                                tryi:
32464 00002590 803E[CD02]00             cmp     byte [multi_pass_id],0; the initial pass for DOS=HI
32465 00002595 7503                     jne     short not_init_pass
32466 00002597 E97F01                   jmp     multi_try_doshi
32467                                not_init_pass:
32468 0000259A 803E[CD02]02             cmp     byte [multi_pass_id],2; the second pass was for ifs=
32469                                       ; 11/12/2022
32470                                   ;je     short multi_pass_coff2; now it is NOPs
32471 0000259F 74E5                     je      short coff
32472                                   ; 27/10/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
32473                                   ;je     short multi_pass_coff
32474                                                   ; This pass can be made use of if
32475                                                   ; we want do some config.sys process
32476                                                   ; after device drivers are loaded
32477                                                   ; and before install= commands
32478                                                   ; are processed
32479
32480 000025A1 803E[CD02]03               cmp     byte [multi_pass_id],3; the third pass for install= ?
32481 000025A6 741D                     je      short multi_try_i
32482 000025A8 80FC48                       cmp     ah, CONFIG_DOS  ; 'H'
32483                                   ; 11/12/2022
32484                                   ;je     short multi_pass_coff2
32485 000025AB 74D9                     je      short coff
32486                                   ; 27/10/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
32487                                   ;je     short multi_pass_coff
32488
32489                                ;       make note of any INSTALL= or INSTALLHIGH= commands we find,
32490                                ;       but don't process them now.
32491
32492 000025AD 80FC49                       cmp     ah,CONFIG_INSTALL ; 'I'  ; install= command?
32493                                   ; 31/12/2022 - Retro DOS v4.2 (Modified MSDOS 6.21 IO.SYS)
32494 000025B0 7507                     jne     short precheck_installhigh ; the first pass is for normal operation.
32495                                   ; 27/10/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
32496                                   ;jne    short tryb
32497
32498                                       ;or     word [install_flag],have_install_cmd ; set the flag
32499 000025B2 800E[CE02]01               or      byte [install_flag],have_install_cmd ; 1
32500                                multi_pass_coff2:
32501 000025B7 EBCD                     jmp     short coff ; 13/05/2019       ; and handles the next command
32502
32503                                ; 31/12/2022 - Retro DOS v4.2 (Modified MSDOS 6.21 IO.SYS)
32504                                ; (SYSINIT:2448h)
32505                                ; 27/10/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
32506                                ;%if 0
32507                                precheck_installhigh:
32508 000025B9 80FC57                       cmp     ah,CONFIG_INSTALLHIGH ; 'W' ; signifier for INSTALLHIGH
32509 000025BC 756B                         jne     short tryb            ; carry on with normal processing
32510                                       ;or     word [install_flag],have_install_cmd
32511 000025BE 800E[CE02]01               or      byte [install_flag],have_install_cmd ; 1
32512 000025C3 EBC1                       jmp     short coff
32513                                ;%endif ; 27/10/2022
32514
32515                                multi_try_i:
32516 000025C5 80FC49                           cmp     ah,CONFIG_INSTALL ; 'I' ; install= command?
32517                                   ; 31/12/2022 - Retro DOS v4.2
32518 000025C8 750A                     jne     short multi_try_n     ; no, check for installhigh
32519                                   ; 27/10/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
32520                                   ;jne    short multi_pass_filter
32521
32522                                ; 31/12/2022
32523                                ;%if 1
32524                                ; 27/10/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
32525                                ;%if 0
32526                                ;ifdef      MULTI_CONFIG
32527 000025CA E84F20                     call    query_user              ; query the user if config_cmd
32528 000025CD 7241                       jc      short multi_pass_filter       ; has the CONFIG_OPTION_QUERY bit set
32529                                ;endif
32530                                ;%endif ; 27/10/2022
32531
32532 000025CF E8C7EC                     call    do_install_exec       ;install it.
32533 000025D2 EBB2                       jmp     short coff            ;to handle next install= command.
32534
32535                                ; 31/12/2022 - Retro DOS v4.2 (Modified MSDOS 6.21 IO.SYS)
32536                                ; (SYSINIT:2463h)
32537                                ;%if 1
32538                                ; 27/10/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
32539                                ;%if 0
32540
32541                                multi_try_n:
32542 000025D4 80FC57                           cmp     ah,CONFIG_INSTALLHIGH   ; installhigh= command?
32543 000025D7 7537                             jne short multi_pass_filter      ; no. ignore this.
32544                                ;ifdef      MULTI_CONFIG
32545 000025D9 E84020                           call    query_user              ; query the user if config_cmd
32546 000025DC 7232                             jc      short multi_pass_filter  ; has the CONFIG_OPTION_QUERY bit set
32547                                ;endif
32548
32549                                ;       The memory environment is in its normal DOS state, so do
32550                                ;       the standard calls to set the alloc strategy for loading high
32551
32552 000025DE B80058                    mov     ax,(ALLOCOPER<<8)|0 ; 5800h
32553 000025E1 CD21                     int     21h                   ;get alloc strategy
32554 000025E3 89C3                     mov     bx,ax
32555 000025E5 53                           push    bx                      ; save for the return
32556
32557 000025E6 81CB8000                     or  bx,HIGH_FIRST  ; 80h  ;set alloc to HighFirst
32558 000025EA B80158                   mov     ax,(ALLOCOPER<<8)|1 ; 5801h
32559 000025ED CD21                     int     21h                   ;set alloc strategy
32560
32561 000025EF B80258                   mov      ax,(ALLOCOPER<<8)|2 ; 5802h
32562 000025F2 CD21                         int     21h                     ; get link state
32563 000025F4 30E4                         xor     ah,ah                   ; clear top byte
32564 000025F6 50                           push    ax                      ; save for return
32565
32566 000025F7 B80358                       mov ax,(ALLOCOPER<<8)|3 ; 5803h
32567 000025FA BB0100                   mov     bx,1
32568 000025FD CD21                     int     21h                   ;link in UMBs
32569
32570 000025FF E897EC                   call    do_install_exec       ;install it.
32571
32572 00002602 B80358                       mov     ax,(ALLOCOPER<<8)|3
32573 00002605 5B                           pop     bx                      ; recover original link state
32574 00002606 CD21                         int     21h
32575 00002608 5B                           pop     bx                      ; recover original alloc strategy
32576 00002609 B80158                       mov     ax,(ALLOCOPER<<8)|1
32577 0000260C CD21                         int     21h
32578
32579                                   ;jmp    short coff            ;to handle next install= command.
32580                                   ; 01/01/2023
32581 0000260E EBA7                     jmp     short multi_pass_coff2
32582
32583                                ;%endif ; 27/10/2022
```

```
                                multi_pass_filter:
00002610 80FC59                         cmp     ah,CONFIG_COMMENT ; 'Y' ; comment?
00002613 740A                       je      short multi_pass_adjust
00002615 80FC5A                         cmp     ah,CONFIG_UNKNOWN ; 'Z' ; bad command?
00002618 7405                       je      short multi_pass_adjust
0000261A 80FC30                         cmp     ah,CONFIG_REM  ; '0'     ; rem?
0000261D 7508                       jne     short multi_pass_coff ; ignore the rest of the commands.

                                multi_pass_adjust:                ; these commands need to
0000261F FF0E[5A03]                 dec     word [chrptr]         ;  adjust chrptr,count
00002623 FF06[5603]                 inc     word [count]          ;  for newline proc.

                                multi_pass_coff:
                                    ; 11/12/2022
                                    ;jmp    short coff            ; to handle next install= commands.
                                    ; 01/01/2023
00002627 EB8E                       jmp     short multi_pass_coff2

                                ;--------------------------------------------------------------------------------
                                ; buffer command
                                ;--------------------------------------------------------------------------------

                                ;******************************************************************************
                                ;                                                                             *
                                ; function: parse the parameters of buffers= command.                               *
                                ;                                                                             *
                                ; input :                                                                     *
                                ;   es:si -> parameters in command line.                                      *
                                ; output:                                                                    *
                                ;   buffers set                                                               *
                                ;   buffer_slash_x flag set if /x option chosen.                              *
                                ;   h_buffers set if secondary buffer cache specified.                        *
                                ;                                                                             *
                                ; subroutines to be called:                                                        *
                                ;   sysinit_parse                                                             *
                                ; logic:                                                                     *
                                ; {                                                                           *
                                ;   set di points to buf_parms;  /*parse control definition*/                  *
                                ;   set dx,cx to 0;                                                           *
                                ;   reset buffer_slash_x;                                                     *
                                ;   while (end of command line)                                               *
                                ;   { sysinit_parse;                                                          *
                                ;     if (no error) then                                                      *
                                ;          if (result_val._$P_synonym_ptr == slash_e) then /*not a switch *
                                ;              buffer_slash_x = 1                                           *
                                ;          else if  (cx == 1) then           /* first positional */    *
                                ;                    buffers = result_val._$P_picked_val;                          *
                                ;               else  h_buffers = result_val._$P_picked_val;                *
                                ;     else {show error message;error exit}                                    *
                                ;   };                                                                        *
                                ;   if (buffer_slash_x is off & buffers > 99) then show_error;                *
                                ; };                                                                          *
                                ;                                                                             *
                                ;******************************************************************************

                                    ; 27/10/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 IO.SYS)
                                    ; (SYSINIT:229Ch)
                                tryb:
00002629 80FC42                         cmp     ah,CONFIG_BUFFERS ; 'B'
0000262C 755C                       jne     short tryc

                                ; 31/12/2022 - Retro DOS v4.2 (Modified MSDOS 6.21 IO.SYS)
                                ; (SYSINIT:24BFh)
                                ; 27/10/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
                                ;%if 0
                                ;ifdef      MULTI_CONFIG
0000262E E8EB1F                     call    query_user            ; query the user if config_cmd
00002631 7257                       jc      short tryc            ; has the CONFIG_OPTION_QUERY bit set
                                ;endif
                                ;%endif ; 27/10/2022

                                    ; 27/10/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
                                    ; 18/12/2022
00002633 31C9                       xor     cx,cx
                                    ;mov    byte [p_buffer_slash_x],0 ; 31/03/2019
00002635 880E[1522]                 mov     [p_buffer_slash_x],cl ; 0

00002639 BF[CE21]                   mov     di,buf_parms
                                    ;xor    cx,cx   ; 18/12/2022
                                    ; 03/01/2023
                                    ;mov    dx,cx
                                do7:
0000263C E82808                     call    sysinit_parse
0000263F 7303                       jnc     short if7             ; parse error,
                                    ;call   badparm_p             ;  and show messages and end the search loop.
                                    ;;jmp   short sr7
                                    ; 31/12/2022
                                ;sr7:
                                    ;jmp    coff
                                    ; 03/01/2023
00002641 E91207                     jmp     badparm_p_coff
                                if7:
00002644 83F8FF                     cmp     ax,_$P_RC_EOL ; 0FFFFh; end of line?
00002647 741A                       je      short en7             ;  then jmp to $endloop for semantic check
                                    ;cmp    word [result_val_swoff],switch_x
00002649 813E[1922][0E22]           cmp     word [result_val+_$P_Result_Blk.SYNONYM_Ptr],switch_x
                                    ;jne    short if11
                                    ; 31/12/2022
0000264F 74EB                       je      short do7 ;je short en11

                                ;   mov     byte [p_buffer_slash_x],1 ; set the flag M016
                                    ;jmp    short en11 ; 31/12/2022
                                if11:
                                    ;mov    ax,[rv_dword]
00002651 A1[1B22]                   mov     ax,[result_val+_$P_Result_Blk.Picked_Val]
00002654 83F901                     cmp     cx,1
00002657 7505                       jne     short if13

00002659 A3[1122]                   mov     [p_buffers],ax
                                    ;jmp    short en11
                                    ; 31/12/2022
0000265C EBDE                       jmp     short do7
                                if13:
0000265E A3[1322]                   mov     [p_h_buffers],ax
                                en11:
00002661 EBD9                       jmp     short do7
                                en7:
00002663 833E[1122]63               cmp     word [p_buffers],99
00002668 760B                       jbe     short if18

                                ;   cmp     byte [p_buffer_slash_x],0 ; M016
                                ;   jne     short if18
```

```
32708 0000266A E82508                   call    badparm_p
32709 0000266D C706[1322]0000           mov     word [p_h_buffers],0
32710 00002673 EB12                     jmp     short sr7
32711                                   if18:
32712 00002675 A1[1122]                     mov     ax,[p_buffers] ; we don't have any problem.
32713 00002678 A3[9902]                     mov     [buffers],ax   ; now,let's set it really.
32714
32715 0000267B A1[1322]                     mov     ax,[p_h_buffers]
32716 0000267E A3[9B02]                     mov     [h_buffers],ax
32717
32718                                   ;   mov     al,[p_buffer_slash_x] ; M016
32719                                   ;   mov     [buffer_slash_x],al
32720
32721 00002681 A1[AF02]                     mov     ax,[linecount]
32722 00002684 A3[B902]                     mov     [buffer_linenum],ax ; save the line number for the future use.
32723                                       ; 31/12/2022
32724                                       ;jmp    short sr7
32725                                       ; 03/01/2023
32726                                   sr7:
32727 00002687 E9FCFE                       jmp     coff
32728
32729                                   ;---------------------------------------------------------------------------
32730                                   ; break command
32731                                   ;---------------------------------------------------------------------------
32732
32733                                   ;****************************************************************************
32734                                   ;                                                                           *
32735                                   ; function: parse the parameters of break = command.                        *
32736                                   ;                                                                           *
32737                                   ; input :                                                                   *
32738                                   ;   es:si -> parameters in command line.                                   *
32739                                   ; output:                                                                   *
32740                                   ;   turn the control-c check on or off.                                    *
32741                                   ;                                                                           *
32742                                   ; subroutines to be called:                                                 *
32743                                   ;   sysinit_parse                                                          *
32744                                   ; logic:                                                                    *
32745                                   ; {                                                                         *
32746                                   ;   set di to brk_parms;                                                   *
32747                                   ;   set dx,cx to 0;                                                       *
32748                                   ;   while (end of command line)                                           *
32749                                   ;   { sysinit_parse;                                                      *
32750                                   ;     if (no error) then                                                    *
32751                                   ;          if (result_val._$P_item_tag == 1) then        /*on          */ *
32752                                   ;             set p_ctrl_break,on;                                          *
32753                                   ;          else                                    /*off         */ *
32754                                   ;             set p_ctrl_break,off;                                         *
32755                                   ;     else {show message;error_exit};                                       *
32756                                   ;   };                                                                      *
32757                                   ;   if (no error) then                                                     *
32758                                   ;      dos function call to set ctrl_break check according to             *
32759                                   ; };                                                                        *
32760                                   ;                                                                           *
32761                                   ;****************************************************************************
32762
32763                                       ; 27/10/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 IO.SYS)
32764                                       ; (SYSINIT:22FFh)
32765                                   tryc:
32766 0000268A 80FC43                         cmp     ah,CONFIG_BREAK ; 'C'
32767 0000268D 7539                     jne     short trym
32768
32769                                   ; 31/12/2022 - Retro DOS v4.2 (Modified MSDOS 6.21 IO.SYS)
32770                                   ; (SYSINIT:2527h)
32771                                   ; 27/10/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
32772                                   ;%if 0
32773                                   ;ifdef      MULTI_CONFIG
32774 0000268F E88A1F                       call    query_user              ; query the user if config_cmd
32775 00002692 7234                     jc      short trym              ; has the CONFIG_OPTION_QUERY bit set
32776                                   ;endif
32777                                   ;%endif ; 27/10/2022
32778
32779 00002694 BF[1F22]                     mov     di,brk_parms
32780 00002697 31C9                     xor     cx,cx
32781                                       ; 03/01/2023
32782                                       ;mov    dx,cx
32783                                   do22:
32784 00002699 E8CB07                       call    sysinit_parse
32785 0000269C 7303                     jnc     short if22              ; parse error
32786                                       ;call   badparm_p               ;  show message and end the search loop.
32787                                       ;;jmp   short sr22
32788                                       ; 31/12/2022
32789                                   ;sr22:
32790                                       ;jmp    coff
32791                                       ; 03/01/2023
32792 0000269E E9B506                   jmp     badparm_p_coff
32793                                   if22:
32794 000026A1 83F8FF                       cmp     ax,_$P_RC_EOL           ; end of line?
32795 000026A4 7415                     je      short en22              ; then end the $endloop
32796
32797                                       ;cmp    byte [result_val_itag],1
32798 000026A6 803E[1822]01                 cmp     byte [result_val+_$P_Result_Blk.Item_Tag],1
32799 000026AB 7507                         jne     short if26
32800
32801 000026AD C606[4422]01                 mov     byte [p_ctrl_break],1 ; turn it on
32802                                       ;jmp    short en26
32803                                       ; 31/12/2022
32804 000026B2 EBE5                     jmp     short do22
32805                                   if26:
32806 000026B4 C606[4422]00                 mov     byte [p_ctrl_break],0 ; turn it off
32807                                   en26:
32808 000026B9 EBDE                     jmp     short do22              ; we actually set the ctrl break
32809                                   en22:
32810 000026BB B433                         mov     ah,SET_CTRL_C_TRAPPING ; if we don't have any parse error.
32811 000026BD B001                         mov     al,1
32812 000026BF 8A16[4422]                   mov     dl,[p_ctrl_break]
32813 000026C3 CD21                         int     21h
32814                                       ; 31/12/2022
32815                                       ;jmp    short sr22
32816                                       ; 03/01/2023
32817                                   sr22:
32818 000026C5 E9BEFE                       jmp     coff
32819
32820                                   ;---------------------------------------------------------------------------
32821                                   ; multitrack command
32822                                   ;---------------------------------------------------------------------------
32823
32824                                   ;****************************************************************************
32825                                   ;                                                                           *
32826                                   ; function: parse the parameters of multitrack= command.                    *
32827                                   ;                                                                           *
32828                                   ; input :                                                                   *
32829                                   ;   es:si -> parameters in command line.                                   *
32830                                   ; output:                                                                   *
32831                                   ;   turn multrk_flag on or off.                                            *
```

```
                                ;                                                                       *
                                ; subroutines to be called:                                                    *
                                ;   sysinit_parse                                                       *
                                ; logic:                                                                *
                                ; {                                                                     *
                                ;   set di to brk_parms;                                                *
                                ;   set dx,cx to 0;                                                     *
                                ;   while (end of command line)                                         *
                                ;   { sysinit_parse;                                                    *
                                ;     if (no error) then                                                *
                                ;          if (result_val._$P_item_tag == 1) then        /*on          */   *
                                ;             set p_mtrk,on;                                           *
                                ;          else                                    /*off         */   *
                                ;             set p_mtrk,off;                                          *
                                ;     else {show message;error_exit};                                   *
                                ;   };                                                                  *
                                ;   if (no error) then                                                  *
                                ;      dos function call to set multrk_flag according to p_mtrk.                *
                                ;                                                                       *
                                ; };                                                                    *
                                ;                                                                       *
                                ;***************************************************************************
                                
                                    ; 27/10/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 IO.SYS)
                                trym:
000026C8 80FC4D                     cmp     ah,CONFIG_MULTITRACK  ; 'M'
000026CB 7573                       jne     short tryu
                                
                                ; 31/12/2022 - Retro DOS v4.2 (Modified MSDOS 6.21 IO.SYS)
                                ; (SYSINIT:2569h)
                                ; 27/10/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
                                ;%if 0
                                ;ifdef      MULTI_CONFIG
000026CD E84C1F                     call    query_user      ; query the user if config_cmd
000026D0 726E                       jc      short tryu      ; has the CONFIG_OPTION_QUERY bit set
                                ;endif
                                ;%endif     ; 27/10/2022
                                
000026D2 BF[2323]                   mov     di,mtrk_parms
000026D5 31C9                       xor     cx,cx
                                    ; 03/01/2023
                                    ;mov    dx,cx
                                do31:
000026D7 E88D07                     call    sysinit_parse
000026DA 7303                       jnc     short if31      ; parse error
                                    ;call   badparm_p       ;  show message and end the search loop.
                                    ;;jmp   short sr31
                                    ; 31/12/2022
                                ;sr31:
                                    ;jmp    coff
                                    ; 03/01/2023
000026DC E97706                     jmp     badparm_p_coff
                                if31:
000026DF 83F8FF                     cmp     ax,_$P_RC_EOL  ; end of line?
000026E2 7415                       je      short en31      ; then end the $endloop
                                
                                    ;cmp    byte [result_val_itag],1
000026E4 803E[1822]01               cmp     byte [result_val+_$P_Result_Blk.Item_Tag],1
000026E9 7507                       jne     short if35
                                
000026EB C606[3723]01               mov     byte [p_mtrk],1       ; turn it on temporarily.
                                    ;jmp    short en35
                                    ; 31/12/2022
000026F0 EBE5                       jmp     short do31
                                if35:
000026F2 C606[3723]00               mov     byte [p_mtrk],0       ; turn it off temporarily.
                                en35:
000026F7 EBDE                       jmp     short do31      ; we actually set the multrk_flag here
                                en31:
000026F9 1E                         push    ds
                                    ;;mov   ax,Bios_Data ; 70h
                                    ;mov    ax,KERNEL_SEGMENT ; 70h
                                    ; 21/10/2022
000026FA B87000                     mov     ax,DOSBIODATASEG ; 0070h
000026FD 8ED8                       mov     ds,ax
                                
000026FF 2E803E[3723]00             cmp     byte [cs:p_mtrk],0
00002705 7508                       jne     short if39
                                
00002707 C706[A004]0100             mov     word [multrk_flag],multrk_off2       ; 0001h
0000270D EB06                       jmp     short en39
                                if39:
0000270F C706[A004]8000             mov     word [multrk_flag],multrk_on  ; 0080h
                                en39:
00002715 1F                         pop     ds
                                    ; 31/12/2022
                                    ;jmp    short sr31
                                    ; 03/01/2023
                                sr31:
00002716 E96DFE                     jmp     coff
                                
                                ;--------------------------------------------------------------------------
                                ; DOS=HIGH/LOW command
                                ;--------------------------------------------------------------------------
                                
                                    ; 27/10/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 IO.SYS)
                                multi_try_doshi:
00002719 80FC48                     cmp     ah,CONFIG_DOS ; 'H'
0000271C 7403                       je      short it_is_h
                                skip_it:
0000271E E9EFFE                     jmp     multi_pass_filter
                                it_is_h:                            ; M003 - removed initing DevUMB
                                                                    ;        & runhigh
                                ; 31/12/2022 - Retro DOS v4.2 (Modified MSDOS 6.21 IO.SYS)
                                ; (SYSINIT:25C1h)
                                ; 27/10/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
                                ;%if 0
                                ;ifdef      MULTI_CONFIG
00002721 E8F81E                     call    query_user              ; query the user if config_cmd
00002724 72F8                       jc      short skip_it         ; has the CONFIG_OPTION_QUERY bit set
                                ;endif
                                ;%endif ; 27/10/2022
                                
00002726 BF[9923]                   mov     di,dos_parms
00002729 31C9                       xor     cx,cx
                                    ; 03/01/2023
                                    ;mov    dx,cx
                                h_do_parse:
0000272B E83907                     call    sysinit_parse
0000272E 7303                       jnc     short h_parse_ok      ; parse error
                                h_badparm:
                                    ; 03/01/2023
                                    ;call   badparm_p             ; show message and end the search loop.
                                    ;;jmp   short h_end
```

```
                                    ; 11/12/2022
                                ;h_end:
                                    ;jmp    coff
                                    ; 03/01/2023
00002730 E92306                     jmp     badparm_p_coff
                                h_parse_ok:
00002733 83F8FF                     cmp     ax,_$P_RC_EOL           ; end of line?
00002736 7405                       je      short h_end             ; then end the $endloop
00002738 E8AE07                     call    ProcDOS
0000273B EBEE                       jmp     short h_do_parse
                                    ; 11/12/2022
                                    ; 03/01/2023
                                h_end:
0000273D E946FE                     jmp     coff

                                ;--------------------------------------------------------------------------------
                                ; devicehigh command
                                ;--------------------------------------------------------------------------------

                                    ; 28/10/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 IO.SYS)
                                tryu:
00002740 80FC55                         cmp     ah,CONFIG_DEVICEHIGH ; 'U'
00002743 7554                       jne     short tryd

                                ; 31/12/2022 - Retro DOS v4.2 (Modified MSDOS 6.21 IO.SYS)
                                ; (SYSINIT:25E9h)
                                ; 28/10/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
                                ;%if 0
                                ;ifdef      MULTI_CONFIG
00002745 E8D41E                     call    query_user              ; query the user if config_cmd
00002748 724F                       jc      short tryd              ; has the CONFIG_OPTION_QUERY bit set
                                ;endif
                                ;%endif ; 28/10/2022

                                ; 28/10/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
                                ;%if 0
                                    ; 01/01/2023
                                    ; ds = cs
0000274A E83108                     call    InitVar
0000274D E80510                     call    ParseSize               ; process the size= option
                                    ;jnc    short tryu_0
                                    ; 31/12/2022
00002750 720C                       jc      short tryu_1 ; 31/03/2019 - Retro DOS v4.0

                                ;%endif ; 28/10/2022

                                ; 31/12/2022
                                %if 0
                                    ; 28/10/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
                                    ;mov    [cs:badparm_off], si  ; stash it there in case of an error
                                    ;mov    [cs:badparm_seg], es
                                    ; 11/12/2022
                                    ; ds = cs
                                    mov     [badparm_off], si
                                    mov     [badparm_seg], es

                                    ; 31/12/2022
                                    ;call   ParseSize
                                    ;jnc    short tryu_2   ; 28/10/2022

                                    ;call   badparm_p
                                    ;jmp    coff
                                    ; 03/01/2023
                                    jmp     badparm_p_coff
                                %endif

                                ; 31/12/2022 - Retro DOS v4.2 (Modified MSDOS 6.21 IO.SYS)
                                ; (SYSINIT:2606h)
                                ; 28/10/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
                                ;%if 0
                                tryu_0:
                                    ;mov    ax,[cs:DevSizeOption]
                                    ; 31/12/2022
00002752 A1[5224]                   mov     ax,[DevSizeOption] ; ds = cs
00002755 09C0                       or      ax,ax
00002757 7510                       jnz     short tryu_2

00002759 E8B408                     call    ParseVar
0000275C 730B                       jnc     short tryu_2
                                tryu_1:
                                    ; 31/12/2022
                                    ; ds = cs
0000275E 8936[6B19]                 mov     [badparm_off], si
00002762 8C06[6D19]                 mov     [badparm_seg], es
                                    ;mov    [cs:badparm_off], si  ; If ParseVar up there failed, then
                                    ;mov    [cs:badparm_seg], es  ; ES:SI points to its problem area...
                                    ;call   badparm_p             ; so all we have to do is choke and
                                    ;jmp    coff                  ; die, rather verbosely.
                                    ; 03/01/2023
00002766 E9ED05                     jmp     badparm_p_coff

                                ;%endif ; 28/10/2022

                                tryu_2:
00002769 56                         push    si
0000276A 06                         push    es

                                    ; 08/09/2023 - Retro DOS 4.2 IO.SYS (Optimization)
                                    ; MSDOS 6.21 IO.SYS - SYSINIT:2623h
                                    ; PCDOS 7.1 IBMBIO.COM - SYSINIT:2B17h
                                tryu_3:
0000276B 268A04                     mov     al,[es:si]
0000276E 3C0D                       cmp     al,cr
                                    ; 14/04/2024
                                    ;je     short tryu_4
                                    ; 08/09/2023 - Retro DOS 5.0 (PCDOS 7.1 IBMBIO.COM)
00002770 740C                       je      short tryu_5
00002772 3C0A                       cmp     al,lf
00002774 740A                       je      short tryu_4
00002776 E81120                     call    delim
00002779 7405                       jz      short tryu_4
0000277B 46                         inc     si
0000277C EBED                       jmp     short tryu_3

                                    ; 14/04/2024
                                    ; 08/09/2023 - Retro DOS 5.0 (PCDOS 7.1 IBMBIO.COM)
                                tryu_5:
0000277E B020                       mov     al,20h ; ' '   ; blank instead of cr

                                tryu_4:
                                    ; 11/12/2022
                                    ; ds = cs
```

```
33080 00002780 A2[6624]                 mov     [DevSavedDelim],al
33081                                   ;mov    [cs:DevSavedDelim],al ; Save the delimiter before replacing
33082                                                                 ;  it with null
33083                                   ; 18/12/2022
33084 00002783 29DB                     sub     bx,bx
33085 00002785 26881C                   mov     [es:si],bl ; 0
33086                                   ;mov    byte [es:si],0
33087
33088 00002788 07                        pop     es
33089 00002789 5E                       pop     si      ; 14/04/2024
33090
33091                                 ;--------------------------------------------------------------------------
33092                                 ; BEGIN PATCH TO CHECK FOR NON-EXISTANT UMBs   -- t-richj 7-21-92
33093                                 ;--------------------------------------------------------------------------
33094
33095                                 ; 31/12/2022 - Retro DOS v4.2 (Modified MSDOS 6.21 IO.SYS)
33096                                 ; (SYSINIT:2642h)
33097                                 ; 28/10/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
33098                                 ;%if 0
33099                                 ; 10/07/2023 - Retro DOS v4.2 IO.SYS (optimized)
33100                                 ; MSDOS 6.21 IO.SYS - SYSINIT:2642h
33101                                 %if 1
33102                                     ; 01/01/2023
33103                                     ; ds = cs
33104 0000278A E8F00C                   call    UmbTest                 ; See if UMBs are around...
33105                                   ; 01/01/2023
33106                                   ;jnc    short NrmTst            ; ...yep. So do that normal thang.
33107
33108                                   ;mov    byte [cs:DeviceHi],0    ; ...nope... so load low.
33109                                   ; 31/12/2022
33110                                   ; ds = cs, bx = 0
33111                                   ;mov    byte [DeviceHi],bl ; 0
33112                                   ;jmp    short LoadDevice
33113                                   ; 01/01/2023
33114 0000278D 7222                     jc      short LoadDevice ; bl = 0
33115                                 %endif
33116                                 ;%endif
33117                                 ;--------------------------------------------------------------------------
33118                                 ; END PATCH TO CHECK FOR NON-EXISTANT UMBs   -- t-richj 7-21-92
33119                                 ;--------------------------------------------------------------------------
33120
33121                                 NrmTst:
33122                                     ; 11/12/2022
33123                                     ; ds = cs
33124                                     ;;mov   byte [cs:DeviceHi],0
33125                                     ;mov    byte [DeviceHi],0
33126                                     ; 18/12/2022
33127                                     ; bx = 0
33128 0000278F 381E[4224]               cmp     [DevUMB],bl ; 0
33129                                   ;cmp    byte [DevUMB],0
33130                                   ;;cmp   byte [cs:DevUMB],0      ; do we support UMBs
33131 00002793 741C                     je      short LoadDevice        ; no, we don't
33132                                   ;mov    byte [cs:DeviceHi],1
33133                                   ; 11/12/2022
33134                                   ;mov    byte [DeviceHi],1
33135                                   ; 18/12/2022
33136 00002795 FEC3                     inc     bl ; mov bl,1 ; (*)
33137                                   ; 11/12/2022
33138                                   ;jmp    short LoadDevice2       ; 11/12/2022
33139 00002797 EB18                     jmp     short LoadDevice
33140
33141                                 ;--------------------------------------------------------------------------
33142                                 ; device command
33143                                 ;--------------------------------------------------------------------------
33144
33145                                     ; 31/12/2022 - Retro DOS v4.2 (Modified MSDOS 6.21 IO.SYS)
33146                                     ; (SYSINIT:2665h)
33147
33148                                     ; 28/10/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 IO.SYS)
33149                                     ; (SYSINIT:2401h)
33150                                 tryd:
33151                                     ; 11/12/2022
33152                                     ;xor    bx,bx ; 31/12/2022
33153                                     ;
33154 00002799 80FC44                       cmp     ah,CONFIG_DEVICE ; 'D'
33155 0000279C 7403                     je      short gotd
33156                                 skip_it2:
33157 0000279E E9FC02                   jmp     tryq
33158                                 gotd:
33159
33160                                 ; 31/12/2022 - Retro DOS v4.2
33161                                 ; 28/10/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
33162                                 ;%if 0
33163                                 ;ifdef      MULTI_CONFIG
33164 000027A1 E8781E                   call    query_user              ; query the user if config_cmd
33165 000027A4 72F8                     jc      short skip_it2          ; has the CONFIG_OPTION_QUERY bit set
33166                                 ;endif
33167                                 ;%endif ; 28/10/2022
33168
33169                                     ; 31/12/2022
33170 000027A6 29DB                     sub     bx,bx
33171                                   ; bx = 0
33172                                   ; 11/12/2022
33173                                   ; ds = cs
33174                                   ;mov    byte [DeviceHi],0
33175                                   ;mov    word [DevSizeOption],0
33176 000027A8 891E[5224]               mov     [DevSizeOption],bx ; 0
33177 000027AC C606[6624]20             mov     byte [DevSavedDelim],' '
33178                                   ;mov    byte [cs:DeviceHi],0    ; not to be loaded in UMB ;M007
33179                                   ;mov    word [cs:DevSizeOption],0
33180                                   ;mov    byte [cs:DevSavedDelim],' ' ; In case of DEVICE= the null has to
33181                                                                       ;  be replaced with a ' '
33182                                 LoadDevice:                             ; device= or devicehigh= command.
33183                                     ; 11/12/2022
33184                                     ;mov    byte [DeviceHi],0
33185 000027B1 881E[5124]               mov     byte [DeviceHi],bl      ; 0 or 1 (*)
33186                                 LoadDevice2:
33187                                     ; 28/10/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
33188                                     ;
33189                                     ;push   cs
33190                                        ;pop     ds
33191                                     ;
33192                                     ;mov    [bpb_addr],si           ; pass the command line to the dvice
33193                                     ;mov    [bpb_addr+2],es
33194                                     ;
33195                                     ;mov    [DevCmdLine],si                 ; save it for ourself
33196                                     ;mov    [DevCmdLine+2],es
33197                                     ;
33198                                     ;mov    byte [driver_units],0 ; clear total block units for driver
33199
33200                                     ; 11/12/2022
33201                                     ; ds = cs
33202                                     ;mov    bx,cs
33203                                     ;mov    ds,bx
```

```
33204
33205                                   ;mov    [cs:bpb_addr],si      ; pass the command line to the dvice
33206 000027B5 8936[8103]               mov     [bpb_addr],si
33207                                   ;mov    [cs:bpb_addr+2],es
33208 000027B9 8C06[8303]               mov     [bpb_addr+2],es
33209
33210                                   ;mov    [cs:DevCmdLine],si    ; save it for ourself
33211 000027BD 8936[6224]               mov     [DevCmdLine],si
33212                                   ;mov    [cs:DevCmdLine+2],es
33213 000027C1 8C06[6424]               mov     [DevCmdLine+2],es
33214
33215                                   ; 31/12/2022 - Retro DOS v4.2
33216 000027C5 C606[6A19]00             mov     byte [driver_units],0 ; clear total block units for driver
33217
33218 000027CA E82520                   call    round
33219
33220 000027CD E8910E                   call    SizeDevice
33221 000027D0 723F                     jc      short BadFile
33222
33223                                   ; 11/12/2022
33224                                   ; ds = cs
33225
33226                                 ; - Begin DeviceHigh primary logic changes ---------------------------------
33227
33228                                 ; 31/12/2022 - Retro DOS v4.2 (Modified MSDOS 6.21 IO.SYS)
33229                                 ; (SYSINIT:26A4h)
33230
33231                                 ; 28/10/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
33232                                 ;%if 0
33233 000027D2 C606[4124]01             mov     byte [ConvLoad],1     ; Doesn't matter if DeviceHi==0
33234
33235                                   ; 22/07/2023
33236                                   ;mov    al,[DeviceHi]         ; If not using upper memory,
33237 000027D7 800E[5124]00             or      byte [DeviceHi],0     ; Skip all this and go on to
33238                                   ; 10/07/2023
33239                                   ;or     al,al
33240 000027DC 741E                     jz      short DevConvLoad     ; the actual load.
33241
33242                                   ;call   GetLoadUMB            ; Returns first UMB spec'ed in AX
33243 000027DE A0[FF23]                 mov     al,[UmbLoad]   ; 19/04/2019 - Retro DOS v4.0
33244
33245 000027E1 3CFF                     cmp     al,-1                 ; If umb0 not specified, it's old style
33246 000027E3 7417                     jz      short DevConvLoad     ; so load high even if SIZE= is smaller
33247
33248 000027E5 FE0E[4124]               dec     byte [ConvLoad] ; 0   ; They specified /L, so use new loader
33249
33250 000027E9 E8590A                   call    GetLoadSize           ; Returns size of first UMB specified
33251 000027EC 09C0                     or      ax,ax
33252 000027EE 7406                     jz      short tryd_1          ; If size1 not specified, nada to do:
33253
33254 000027F0 3B06[3324]               cmp     ax,[DevSize]          ; /L:...,Size < DevSize?
33255 000027F4 7D06                     jge     short DevConvLoad
33256                                 tryd_1:
33257 000027F6 A1[3324]                 mov     ax,[DevSize]          ; Size < DevSize, so write DevSize as
33258 000027F9 E8550A                   call    StoLoadSize           ; minsize for load UMB.
33259
33260                                 ;%endif ; 28/10/2022
33261
33262                                 ; - End DeviceHigh primary logic changes -----------------------------------
33263
33264                                 DevConvLoad:
33265                                     ; 28/10/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
33266 000027FC E8660D                   call    InitDevLoad
33267
33268                                     ; 11/12/2022
33269                                     ; ds = cs
33270 000027FF A1[3524]                 mov     ax,[DevLoadAddr]
33271 00002802 0306[3324]               add     ax,[DevSize]
33272 00002806 7206                     jc      short NoMem
33273 00002808 3906[3724]               cmp     [DevLoadEnd],ax
33274 0000280C 7315                     jae     short LoadDev
33275
33276                                   ; 11/12/2022
33277                                   ;mov    ax,[cs:DevLoadAddr]
33278                                   ;add    ax,[cs:DevSize]
33279                                   ;jc     short NoMem
33280                                   ;cmp    [cs:DevLoadEnd],ax
33281                                   ;jae    short LoadDev
33282                                 NoMem:
33283                                   ; 11/12/2022
33284                                   ; ds = cs
33285                                   ;jmp    mem_err
33286 0000280E E92020                   jmp     mem_err2
33287
33288                                 BadFile:
33289                                     ; 28/10/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
33290                                     ;;call  RetFromUM             ; Does nothing if didn't call HideUMBs
33291                                     ;;cmp   byte [es:si],' '
33292                                         ;;jae       short tryd_2
33293                                     ; 31/12/2022
33294                                     ;cmp    byte [es:si],0Dh      ; cr
33295                                         ;jne        short tryd_2
33296                                     ;jmp    badop
33297                                     ; 31/12/2022
33298                                     ; ds = cs
33299                                     ; 31/12/2022 - Retro DOS v4.2 (Modified MSDOS 6.21 IO.SYS)
33300                                     ; (SYSINIT:26E6h)
33301 00002811 E8AE0E                   call    RetFromUM             ; Does nothing if didn't call HideUMBs
33302 00002814 26803C20                 cmp     byte [es:si],' '
33303                                   ;cmp    byte [es:si],20h ; space
33304 00002818 7303                     jnb     short tryd_2
33305 0000281A E96906                   jmp     badop
33306                                 tryd_2:
33307 0000281D E80C22                   call    badload
33308 00002820 E963FD                   jmp     coff
33309
33310                                 LoadDev:
33311 00002823 06                       push    es
33312 00002824 1F                       pop     ds
33313
33314 00002825 89F2                     mov     dx,si                 ;ds:dx points to file name
33315 00002827 E87C0E                   call    ExecDev               ; load device driver using exec call
33316                                 badldreset:
33317 0000282A 1E                       push    ds
33318 0000282B 07                       pop     es                    ;es:si back to config.sys
33319 0000282C 0E                       push    cs
33320 0000282D 1F                       pop     ds                    ;ds back to sysinit
33321 0000282E 72E1                     jc      short BadFile
33322                                 goodld:
33323                                     ; 11/12/2022
33324                                     ; ds = cs
33325
33326 00002830 06                       push    es ; + ; 31/12/2022
33327 00002831 56                       push    si ; ++
```

```
33328 00002832 E89E0E                   call    RemoveNull
33329 00002835 06                       push    es
33330 00002836 56                       push    si
33331
33332 00002837 0E                       push    cs
33333 00002838 07                       pop     es
33334
33335 00002839 1E                       push    ds ; **  ; ds = cs
33336 0000283A 56                       push    si
33337
33338                                   ;lds    si,[cs:DevEntry]      ; peeks the header attribute
33339                                   ; 31/12/2022
33340                                   ; ds = cs
33341 0000283B C536[3924]               lds     si,[DevEntry]
33342
33343                                   ;test   word [si+4],8000h
33344                                   ; 11/12/2022
33345 0000283F F6440580                 test    byte [si+SYSDEV.ATT+1],DEVTYP>>8
33346                                   ;test   word [si+SYSDEV.ATT],DEVTYP ; block device driver?
33347 00002843 7514                     jnz     short got_device_com_cont   ; no.
33348
33349 00002845 2EC536[6D02]             lds     si,[cs:DOSINFO]               ; ds:si -> sys_var
33350                                   ;cmp    byte [si+32],26
33351 0000284A 807C201A                 cmp     byte [si+SYSI_NUMIO],26       ; no more than 26 drive number
33352 0000284E 7209                     jb      short got_device_com_cont
33353
33354 00002850 5E                       pop     si
33355 00002851 1F                       pop     ds ; **
33356
33357 00002852 5E                       pop     si                    ; clear the stack
33358 00002853 07                       pop     es
33359
33360                                   ; 28/10/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
33361                                   ;call   RetFromUM
33362                                   ; 31/12/2022
33363                                   ; ds = cs ; **
33364 00002854 E86B0E                   call    RetFromUM             ; Do this before we leave
33365
33366                                   ;jmp    short badnumblock
33367                                   ; 31/12/2022
33368 00002857 EB73                     jmp     short badnumblock2  ; ds = cs
33369
33370                                 got_device_com_cont:
33371 00002859 5E                       pop     si
33372 0000285A 1F                       pop     ds
33373
33374                                   ; 11/12/2022
33375                                   ; ds = cs
33376
33377 0000285B E8AE06                   call    LieInt12Mem
33378 0000285E E80B07                   call    UpdatePDB             ; update the PSP:2 value M020
33379
33380                                   ; 11/12/2022
33381                                   ; ds = cs
33382                                   ; 08/09/2023
33383 00002861 31C0                     xor     ax, ax ; 0
33384 00002863 3806[6619]               cmp     byte [multdeviceflag],al ; 0
33385                                   ;cmp    byte [multdeviceflag],0
33386                                   ;cmp    byte [cs:multdeviceflag],0 ; Pass limit only for the 1st device
33387                                                                      ;  driver in the file ; M027
33388 00002867 750B                     jne     short skip_pass_limit ;                     ; M027
33389
33390                                   ; 11/12/2022
33391                                   ; ds = cs
33392                                   ;mov    word [cs:break_addr],0; pass the limit to the DD
33393                                   ;mov    bx,[cs:DevLoadEnd]
33394                                   ;mov    [cs:break_addr+2],bx
33395
33396                                   ;mov    word [break_addr],0
33397                                   ; 08/09/2023
33398 00002869 A3[7D03]                 mov     [break_addr],ax ; 0
33399 0000286C 8B1E[3724]               mov     bx,[DevLoadEnd]
33400 00002870 891E[7F03]               mov     [break_addr+2],bx
33401
33402                                 skip_pass_limit:
33403                                 ;   Note: sysi_numio (in DOS DATA) currently reflects the REAL
33404                                 ;   number of installed devices (including DblSpace drives) where
33405                                 ;   "drivenumber" is the number that the next block device will
33406                                 ;   be assigned to. Because some naughty device drivers (like
33407                                 ;   interlnk) look at the internal DOS variable instead of the
33408                                 ;   value we pass it, we'll temporarily stick our value into
33409                                 ;   DOS DATA while we're initializing the device drivers.
33410                                 ;
33411                                 ;   Note that this will make it impossible for this device
33412                                 ;   driver to access the DblSpace drive letters, whether
33413                                 ;   they are swapped-hosts or unswapped compressed drives,
33414                                 ;   during its initialization phase.
33415
33416                                 ; 31/12/2022 - Retro DOS v4.2 (Modified MSDOS 6.21 IO.SYS)
33417                                 ; (SYSINIT:2752h)
33418                                 ; 29/10/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
33419                                 ;%if 0
33420                                     ; 31/12/2022
33421                                     ;push  ds
33422
33423                                     ;lds   bx,[cs:DOSINFO]              ; ds:bx -> sys_var
33424                                     ; 31/12/2022
33425                                     ; ds = cs
33426                                     ; 08/09/2023
33427                                     ;lds   bx,[DOSINFO]          ; ds:bx -> sys_var
33428
33429                                     ;mov   al,[cs:drivenumber]   ; temporarily use this next drv value
33430                                     ;mov   [cs:devdrivenum],al   ; pass drive number in packet to driver
33431                                     ;mov   ah,al
33432
33433                                     ; 08/09/2023
33434                                     ; ds = cs
33435 00002874 A0[8503]                 mov     al,[drivenumber]      ; temporarily use this next drv value
33436 00002877 A2[8503]                 mov     [devdrivenum],al      ; pass drive number in packet to driver
33437 0000287A 88C4                     mov     ah,al
33438 0000287C C51E[6D02]               lds     bx,[DOSINFO]          ; ds:bx -> sys_var
33439
33440 00002880 874720                   xchg    ax,[bx+SYSI_NUMIO]    ; swap with existing values
33441                                   ; 31/12/2022
33442                                   ;pop    ds
33443
33444 00002883 50                       push    ax                    ; save real sysi_numio/ncds in ax
33445
33446                                 ;%endif ; 29/10/2022
33447
33448                                     ; 29/10/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 IO.SYS)
33449                                     ; (SYSINIT:24B9h)
33450
33451 00002884 BB0600                     mov     bx,SYSDEV.STRAT ; 6
```

```
33452 00002887 E8B01F                   call    calldev                 ; calldev (sdevstrat);
33453 0000288A BB0800                   mov     bx,SYSDEV.INT ; 8
33454 0000288D E8AA1F                   call    calldev                 ; calldev (sdevint);
33455
33456                                   ; 11/12/2022
33457                                   ; ds <> cs (from calldev) ; 31/12/2022
33458
33459                                ; 31/12/2022 - Retro DOS v4.2 (Modified MSDOS 6.21 IO.SYS)
33460                                ; (SYSINIT:2773h)
33461                                ; 29/10/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
33462                                ;%if 0
33463 00002890 58                       pop     ax                      ; get real sysi_numio value
33464                                   ; 31/12/2022
33465                                   ;push  ds
33466 00002891 2EC51E[6D02]             lds     bx,[cs:DOSINFO]                 ; ds:bx -> sys_var
33467 00002896 894720                   mov     [bx+SYSI_NUMIO],ax      ; swap with existing values
33468                                   ; 31/12/2022
33469                                   ;pop   ds
33470
33471                                ;%endif ; 29/10/2022
33472
33473                                   ; 11/12/2022
33474 00002899 0E                       push    cs
33475 0000289A 1F                       pop     ds
33476
33477 0000289B E89C06                   call    TrueInt12Mem
33478
33479                                   ; 11/12/2022
33480                                   ; ds = cs
33481                                   ;mov    ax,[cs:break_addr]      ; move break addr from the req packet
33482                                   ;mov    [cs:DevBrkAddr],ax
33483                                   ;mov    ax,[cs:break_addr+2]
33484                                   ;mov    [cs:DevBrkAddr+2],ax
33485 0000289E A1[7D03]                 mov     ax,[break_addr]
33486 000028A1 A3[3D24]                 mov     [DevBrkAddr],ax
33487 000028A4 A1[7F03]                 mov     ax,[break_addr+2]
33488 000028A7 A3[3F24]                 mov     [DevBrkAddr+2],ax
33489
33490                                   ; 29/10/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
33491                                   ;call  RetFromUM               ; There we go... all done.
33492                                   ; 31/12/2022 - Retro DOS v4.2 (Modified MSDOS 6.21 IO.SYS)
33493                                   ; (SYSINIT:2791h)
33494 000028AA E8150E                   call    RetFromUM               ; There we go... all done.
33495
33496                                   ; 31/12/2022
33497                                   ; ds = cs
33498
33499                                   ; 11/12/2022
33500 000028AD 803E[4224]00             cmp     byte [DevUMB],0
33501                                   ;cmp    byte [cs:DevUMB],0
33502 000028B2 7403                     je      short tryd_3
33503 000028B4 E80010                   call    AllocUMB
33504                                   ; 31/12/2022
33505                                   ; ds = cs
33506                                tryd_3:
33507
33508                                ;ifndef ROMDOS
33509                                ;------ If we are waiting to be moved into hma lets try it now !!!
33510
33511                                   ; 11/12/2022
33512                                   ; ds = cs
33513
33514                                   ;cmp    byte [cs:runhigh],0FFh
33515 000028B7 803E[6C02]FF             cmp     byte [runhigh],0FFh ; 11/12/2022
33516 000028BC 7503                     jne     short tryd_4
33517
33518                                   ; 11/12/2022
33519                                   ; ds = cs
33520 000028BE E8D7E1                   call    TryToMovDOSHi           ; move DOS into HMA if reqd
33521                                tryd_4:
33522                                ;endif ; ROMDOS
33523
33524 000028C1 5E                        pop     si
33525 000028C2 1F                       pop     ds
33526 000028C3 C60400                   mov     byte [si],0             ; *p = 0;
33527
33528 000028C6 0E                       push    cs
33529 000028C7 1F                       pop     ds
33530
33531 000028C8 EB1F                     jmp     short was_device_com
33532
33533                                ;--------------------------------------------------------------------------
33534
33535                                ; 02/04/2019 - Retro DOS v4.0
33536
33537                                ; 31/12/2022 - Retro DOS v4.2 (Modified MSDOS 6.21 IO.SYS)
33538                                ; (SYSINIT:27B3h)
33539
33540                                badnumblock:
33541 000028CA 0E                       push    cs
33542 000028CB 1F                       pop     ds
33543                                badnumblock2:       ; 31/12/2022 (ds=cs)
33544 000028CC BA[7051]                 mov     dx,badblock
33545 000028CF E88221                   call    print
33546
33547                                ;------ fall thru ---------------------------------------------------------
33548
33549                                   ; 31/12/2022 - Retro DOS v4.2
33550
33551                                erase_dev_do:                           ; modified to show message "error in config.sys..."
33552
33553                                   ;call  CheckDoubleSpace ; MSDOS 6.21 IO.SYS SYSINIT:27BBh
33554                                                           ; (Note: 'call CheckDoubleSpace'
33555                                                           ; has been removed at 'erase_dev_do:' pos
33556                                                           ; in PCDOS 7.1 IBMBIO.COM - SYSINIT:2CBAh)
33557                                                           ; Erdogan Tan - 10/07/2023
33558 000028D2 5E                       pop     si ; ++
33559 000028D3 07                       pop     es ; + ; 31/12/2022
33560
33561 000028D4 0E                       push    cs
33562 000028D5 1F                       pop     ds
33563
33564                                skip1_resetmemhi:
33565                                   ; 11/12/2022
33566                                   ; ds = cs
33567 000028D6 833E[8603]00             cmp     word [configmsgflag],0
33568                                   ;cmp    word [cs:configmsgflag],0
33569 000028DB 7409                     je      short no_error_line_msg
33570
33571 000028DD E8DA05                   call    error_line              ; no "error in config.sys" msg for device driver. dcr d493
33572                                   ; 11/12/2022
33573                                   ; ds = cs
33574                                   ;mov    word [cs:configmsgflag],0
33575 000028E0 C706[8603]0000           mov     word [configmsgflag],0; set the default value again.
```

```
33576
33577                                   no_error_line_msg:
33578 000028E6 E99DFC                      jmp     coff
33579
33580                                   ;--------------------------------------------------------------------------
33581
33582                                   was_device_com:
33583                                       ; 14/12/2022
33584                                       ; ds = cs
33585 000028E9 A1[3F24]                    mov     ax,[DevBrkAddr+2]
33586                                       ;mov    ax,[cs:DevBrkAddr+2] ; 13/05/2019
33587 000028EC 3B06[3724]                  cmp     ax,[DevLoadEnd]
33588                                       ;cmp    ax,[cs:DevLoadEnd]
33589 000028F0 7605                        jbe     short breakok
33590
33591 000028F2 5E                          pop     si
33592 000028F3 07                          pop     es
33593 000028F4 E91AFF                      jmp     BadFile
33594
33595                                   breakok:
33596                                       ; 14/12/2022
33597                                       ; ds = cs
33598 000028F7 C43E[6D02]                  les     di,[DOSINFO]
33599 000028FB C516[3924]                  lds     dx,[DevEntry]
33600                                       ;lds    dx,[cs:DevEntry]      ;set ds:dx to header
33601 000028FF 89D6                        mov     si,dx
33602
33603                                       ; 14/11/2022
33604                                       ;les    di,[cs:DOSINFO]       ;es:di point to dos info
33605
33606                                       ; 14/12/2022
33607                                       ; ds <> cs
33608
33609                                       ;mov    ax,[si+4]
33610 00002901 8B4404                      mov     ax,[si+SYSDEV.ATT]    ;get attributes
33611                                       ; 12/12/2022
33612 00002904 F6C480                      test    ah,DEVTYP>>8 ; 80h
33613                                       ;test   ax,DEVTYP ; 8000h     ;test if block dev
33614 00002907 7426                        jz      short isblock
33615
33616                                   ;------ lets deal with character devices
33617
33618 00002909 2E800E[6919]02               or      byte [cs:setdevmarkflag],for_devmark ; 2
33619 0000290F E8F40D                      call    DevSetBreak           ;go ahead and alloc mem for device
33620                                   jc_edd:
33621 00002912 72BE                        jc      short erase_dev_do    ;device driver's init routine failed.
33622
33623                                       ; 12/12/2022
33624 00002914 A801                        test    al,ISCIN
33625                                       ;test   ax,ISCIN ; 1          ;is it a console in?
33626 00002916 7408                        jz      short tryclk
33627
33628 00002918 2689550C                     mov     [es:di+SYSI_CON],dx   ; es:di+12
33629 0000291C 268C5D0E                    mov     [es:di+SYSI_CON+2],ds ; es:di+14
33630                                   tryclk:
33631                                       ; 12/12/2022
33632 00002920 A808                        test    al,ISCLOCK
33633                                       ;test   ax,ISCLOCK ; 8        ;is it a clock device?
33634 00002922 7408                        jz      short golink
33635
33636 00002924 26895508                    mov     [es:di+SYSI_CLOCK],dx ; es:di+8
33637 00002928 268C5D0A                    mov     [es:di+SYSI_CLOCK+2],ds ; es:di+10
33638                                   golink:
33639 0000292C E9DF00                      jmp     linkit
33640
33641                                   ;------ deal with block device drivers
33642
33643                                   isblock:
33644 0000292F 2EA0[7C03]                  mov     al,[cs:unitcount]     ;if no units found,erase the device
33645 00002933 08C0                        or      al,al
33646 00002935 749B                        jz      short erase_dev_do
33647                                       ;mov    [si+10],al
33648 00002937 88440A                      mov     [si+SYSDEV.NAME],al   ; number of units in name field
33649                                       ; 29/10/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
33650                                       ;add    [cs:driver_units],al
33651                                       ; 31/12/2022 - Retro DOS v4.2 (Modified MSDOS 6.21 IO.SYS)
33652 0000293A 2E0006[6A19]                add     [cs:driver_units],al  ; keep total for all drivers in file
33653                                   perdrv:
33654 0000293F 98                          cbw                           ; warning no device > 127 units
33655 00002940 89C1                        mov     cx,ax
33656 00002942 88E6                        mov     dh,ah
33657                                       ;mov    dl,[es:di+32]
33658 00002944 268A5520                    mov     dl,[es:di+SYSI_NUMIO] ;get number of devices
33659 00002948 88D4                        mov     ah,dl
33660 0000294A 00C4                        add     ah,al                 ; check for too many devices
33661 0000294C 80FC1A                      cmp     ah,26                 ; 'A' - 'Z' is 26 devices
33662 0000294F 7603                        jbe     short ok_block
33663 00002951 E976FF                      jmp     badnumblock
33664
33665                                   ok_block:
33666 00002954 2E800E[6919]02               or      byte [cs:setdevmarkflag],for_devmark ; 2
33667 0000295A E8A90D                      call    DevSetBreak           ; alloc the device
33668 0000295D 72B3                        jc      short jc_edd
33669 0000295F 26004520                    add     [es:di+SYSI_NUMIO],al ; update the amount
33670
33671 00002963 2E0006[8503]                add     [cs:drivenumber],al   ; remember amount for next device
33672 00002968 2EC51E[8103]                lds     bx,[cs:bpb_addr]      ; point to bpb array
33673                                   perunit:
33674 0000296D 2EC42E[6D02]                les     bp,[cs:DOSINFO]
33675                                       ;les    bp,[es:bp+SYSI_DPB]   ; get first dpb
33676                                       ; 11/12/2022
33677 00002972 26C46E00                    les     bp,[es:bp]
33678                                       ; 29/10/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
33679                                       ;les    bp,[es:bp+0]          ; [es:bp+SYSI_DPB]
33680                                   scandpb:
33681                                       ;cmp    word [es:bp+25],-1
33682 00002976 26837E19FF                  cmp     word [es:bp+DPB.NEXT_DPB],-1
33683 0000297B 7406                        je      short foundpb
33684                                       ;les    bp,[es:bp+25]
33685 0000297D 26C46E19                    les     bp,[es:bp+DPB.NEXT_DPB]
33686 00002981 EBF3                        jmp     short scandpb
33687                                   foundpb:
33688 00002983 2EA1[3D24]                  mov     ax,[cs:DevBrkAddr]
33689 00002987 26894619                    mov     [es:bp+DPB.NEXT_DPB],ax
33690 0000298B 2EA1[3F24]                  mov     ax,[cs:DevBrkAddr+2]
33691 0000298F 2689461B                    mov     [es:bp+DPB.NEXT_DPB+2],ax
33692
33693 00002993 2EC42E[3D24]                les     bp,[cs:DevBrkAddr]
33694 00002998 2E8306[3D24]3D              add     word [cs:DevBrkAddr],DPBSIZ ; 33
33695                                                                    ; 08/09/2023
33696                                                                    ; (61 in PCDOS 7.1 IBMBIO.COM)
33697 0000299E E8440D                      call    RoundBreakAddr
33698
33699 000029A1 26C74619FFFF                mov     word [es:bp+DPB.NEXT_DPB],-1
```

```
33700 000029A7 26C64618FF               mov     byte [es:bp+DPB.FIRST_ACCESS],-1
33701
33702 000029AC 8B37                     mov     si,[bx]                 ;ds:si points to bpb
33703 000029AE 43                       inc     bx
33704 000029AF 43                       inc     bx                      ;point to next guy
33705                                   ;mov    [es:bp+DPB.DRIVE],dx
33706                                   ; 11/12/2022
33707 000029B0 26895600                 mov     [es:bp],dx ; 13/05/2019
33708                                   ; 29/10/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
33709                                   ;mov    [es:bp+0],dx            ; [es:bp+DPB.DRIVE]
33710
33711                                   ; 13/04/2024 - Retro DOS v5.0
33712                                   ; PCDOS 7.1 IBMBIO.COM
33713                                   ;;;
33714 000029B4 52                       push    dx
33715 000029B5 51                       push    cx              ; initialize FAT32 extended DPB parameters/fields
33716 000029B6 BA5241                   mov     dx,4152h        ; 'AR' signature for FAT32 extended DPB
33717 000029B9 31C9                     xor     cx,cx   ; 0
33718                                   ;mov    [es:bp+1Dh],cx
33719 000029BB 26894E1D                 mov     [es:bp+DPB.NEXT_FREE],cx ; last allocated cluster #
33720                                   ;cmp    [si+0Bh],cx     ; BPB.fatsecs16
33721 000029BF 394C0B                   cmp     [si+A_BPB.SECTORSPERFAT],cx ; 0
33722 000029C2 7514                     jnz     short set_dpb   ; FAT DPB (33 bytes)   -jnz-
33723                                                           ; FAT32 DPB (61 bytes) -jz-
33724                                   ;mov    [es:bp+39h],cx
33725 000029C4 26894E39                 mov     [es:bp+DPB.FAT32_NXTFREE],cx ; 0
33726                                   ;mov    [es:bp+3Bh],cx
33727 000029C8 26894E3B                 mov     [es:bp+DPB.FAT32_NXTFREE+2],cx ; 0
33728 000029CC 49                       dec     cx      ; 0FFFFh ; -1
33729                                   ;mov    [es:bp+1Fh],cx
33730 000029CD 26894E1F                 mov     [es:bp+DPB.FREE_CNT],cx ; -1 = unknown
33731                                   ;mov    [es:bp+21h],cx
33732 000029D1 26894E21                 mov     [es:bp+DPB.FREE_CNT+2],cx ; -1 = unknown
33733 000029D5 B95845                   mov     cx,4558h        ; 'EX' signature for FAT32 extended DPB
33734                                 set_dpb:
33735                                   ;;;
33736
33737 000029D8 B453                     mov     ah,SETDPB ; 53h                 ;hidden system call
33738 000029DA CD21                     int     21h
33739                                                   ; DOS - 2+ internal - TRANSLATE BIOS PARAMETER BLOCK
33740                                                   ; DS:SI -> BPB (BIOS Parameter Block)
33741                                                   ; ES:BP -> buffer for DOS Drive Parameter Block
33742                                   ; 13/04/2024
33743                                   ;;;
33744 000029DC 59                       pop     cx
33745 000029DD 5A                       pop     dx
33746                                   ;;;
33747
33748                                   ;mov    ax,[es:bp+2]
33749 000029DE 268B4602                 mov     ax,[es:bp+DPB.SECTOR_SIZE]
33750 000029E2 06                       push    es
33751 000029E3 2EC43E[6D02]             les     di,[cs:DOSINFO]         ;es:di point to dos info
33752                                   ;cmp    ax,[es:di+10h]
33753 000029E8 263B4510                 cmp     ax,[es:di+SYSI_MAXSEC]
33754 000029EC 07                       pop     es
33755                                   ; 13/04/2024
33756                                   ;jna    short iblk_1
33757                                   ;jmp    bad_bpb_size_sector
33758                                   ; 29/10/2022
33759 000029ED 777F                     ja      short bad_bpb_size_sector
33760                                 iblk_1:
33761 000029EF 1E                       push    ds
33762 000029F0 52                       push    dx
33763
33764 000029F1 2EC516[3924]             lds     dx,[cs:DevEntry]
33765                                   ;mov    [es:bp+13h],dx
33766 000029F6 26895613                 mov     [es:bp+DPB.DRIVER_ADDR],dx
33767                                   ;mov    [es:bp+15h],ds
33768 000029FA 268C5E15                 mov     [es:bp+DPB.DRIVER_ADDR+2],ds
33769
33770 000029FE 5A                       pop     dx
33771 000029FF 1F                       pop     ds
33772
33773 00002A00 42                       inc     dx
33774 00002A01 FEC6                     inc     dh
33775                                   ;loop   perunit
33776                                   ; 13/04/2024
33777                                   ;;;
33778 00002A03 49                       dec     cx              ; cx = cx - 1
33779                                                           ; cx = remain count from [cs:unitcount]
33780 00002A04 7403                     jz      short iblk_2    ; cx = 0 -> done
33781 00002A06 E964FF                   jmp     perunit         ; loop until cx is 0
33782                                 iblk_2:
33783                                   ;;;
33784
33785 00002A09 0E                        push    cs
33786 00002A0A 1F                       pop     ds
33787
33788 00002A0B E895E3                   call    TempCDS                 ; set cds for new drives
33789                                   ; 31/12/2022
33790                                   ; ds <> cs
33791                                 linkit:
33792 00002A0E 2EC43E[6D02]             les     di,[cs:DOSINFO]         ;es:di = dos table
33793 00002A13 268B4D22                 mov     cx,[es:di+SYSI_DEV]     ;dx:cx = head of list
33794 00002A17 268B5524                 mov     dx,[es:di+SYSI_DEV+2]
33795
33796 00002A1B 2EC536[3924]             lds     si,[cs:DevEntry]        ;ds:si = device location
33797 00002A20 26897522                 mov     [es:di+SYSI_DEV],si     ;set head of list in dos
33798 00002A24 268C5D24                 mov     [es:di+SYSI_DEV+2],ds
33799 00002A28 8B04                     mov     ax,[si]                 ;get pointer to next device
33800 00002A2A 2EA3[3924]               mov     [cs:DevEntry],ax        ;and save it
33801
33802 00002A2E 890C                     mov     [si],cx                 ;link in the driver
33803 00002A30 895402                   mov     [si+2],dx
33804                                 enddev:
33805 00002A33 5E                       pop     si
33806 00002A34 07                       pop     es
33807 00002A35 40                       inc     ax                      ;ax = ffff (no more devs if yes)?
33808 00002A36 740B                     jz      short coffj3
33809
33810 00002A38 2EFE06[6619]             inc     byte [cs:multdeviceflag] ; possibly multiple device driver.
33811 00002A3D E8E80C                   call    DevBreak                ; M009
33812                                   ; 11/12/2022
33813                                   ; ds = cs (DevBreak)
33814
33815                                   ; 03/04/2019 - Retro DOS v4.0
33816                                   ; MSDOS 6.21 IO.SYS - SYSINIT:290Dh
33817 00002A40 E9EDFD                   jmp     goodld                  ; otherwise pretend we loaded it in
33818                                 coffj3:
33819                                   ; 18/12/2022
33820                                   ; ax = 0
33821 00002A43 2EA2[6619]               mov     [cs:multdeviceflag],al ; 0
33822                                   ;mov    byte [cs:multdeviceflag],0 ; reset the flag
33823 00002A47 E8DE0C                   call    DevBreak
```

```
                                    ; 11/12/2022
                                    ; ds = cs (DevBreak)

                                    ; 31/12/2022 - Retro DOS v4.2 (Modified MSDOS 6.21 IO.SYS)
                                    ; (SYSINIT:2919h)
                                    ; 11/07/2023
00002A4A E80204                     call    CheckProtmanArena

                                    ; 02/11/2022 (MSDOS 5.0 IO.SYS compatibility)
                                    ;;call  CheckProtmanArena     ; adjust alloclim if Protman$ just
                                ;                                 ;  created a bogus arena to try
                                ;                                 ;  to protect some of its resident-
                                ;                                 ;  init code.
                                    ; 13/04/2024 - Retro DOS v5.0
                                    ; PCDOS 7.1 IBMBIO.COM
                                    ;;call  CheckDoubleSpace
                                    ;jmp    coff

                                ;--------------------------------------------------------------------------

                                    ; 13/04/2024 - Retro DOS v5.0
                                    ; PCDOS 7.1 IBMBIO.COM
                                    ;;;

                                CheckDoubleSpace:

                                ;;  ifdef   dblspace_hooks

                                ;   Now check for two special MagicDrv cases:
                                ;
                                ;       a) the last driver load was MagicDrv final placement:
                                ;      -> add number of MagicDrv reserved drives to drivenumber
                                ;
                                ;       b) MagicDrv is currently in temporary home:
                                ;          -> call it to give it a chance to mount and shuffle drives

                                    ;cmp    byte [cs:MagicHomeFlag],0 ; already home?
                                    ; 13/04/2024 (PCDOS 7.1 IBMBIO.COM)
00002A4D 2EF606[6724]01             test    byte [cs:MagicHomeFlag],1 ; already home?
00002A53 7545                       jnz     short no_more_magic_calls ;  nothing more to do if so

                                ;   Now inquire of driver whether it is present, and final located

                                    ;mov    ax,multMagicdrv ; 4A11h
                                    ;mov    bx,MD_VERSION ; 0
                                    ;int    2fh                     ; ch = number of MagicDrv drive letters
                                    ;or     ax,ax                   ; is it there?
                                    ; 13/04/2024 (PCDOS 7.1 IBMBIO.COM)
                                    ;;;
00002A55 E8FEEE                     call    get_dblspace_version ; is it there?
                                    ;jnz    short no_more_magic_calls ; done if not
00002A58 750B                       jnz     short set_magichomeflag
                                    ;;;

00002A5A F7C20080                   test    dx,8000h                ; is it final placed?
00002A5E 751C                       jnz     short magic_not_yet_home ;  skip if not

                                ;   Okay, now the driver is final placed!  Set the flag so we
                                ;   don't keep checking it, and add its number of drive letters
                                ;   to drivenumber.

                                    ; 13/04/2024 (PCDOS 7.1 IBMBIO.COM)
                                    ;mov    byte [cs:MagicHomeFlag],0ffh ; set the flag!
00002A60 2E002E[8503]               add     [cs:drivenumber],ch     ; add number of MagicDrv volumes to
                                                                    ;  the drive number we'll pass to the
                                                                    ;  next loadable block device.
                                    ;jmp    short no_more_magic_calls ; and finished.

                                    ;;;
                                set_magichomeflag:
00002A65 2EC606[6724]01             mov     byte [cs:MagicHomeFlag],1 ; set the flag!
00002A6B E918FB                     jmp     coff
                                    ;;;

                                ; 03/04/2019 - Retro DOS v4.0

                                bad_bpb_size_sector:
00002A6E 5E                         pop     si
00002A6F 07                         pop     es
00002A70 BA[9250]                   mov     dx,badsiz_pre
00002A73 BB[7050]                   mov     bx,crlfm
00002A76 E8B91F                     call    prnerr

00002A79 E90AFB                     jmp     coff

                                magic_not_yet_home:
00002A7C 06                         push    es
00002A7D 56                         push    si

00002A7E 2E8B0E[6403]               mov     cx,[cs:memhi]           ; pass it a work buffer
00002A83 2E8B16[A502]               mov     dx,[cs:ALLOCLIM]        ;   address in cx (segment)
00002A88 29CA                       sub     dx,cx                   ;   for len dx (paragraphs)

00002A8A BB0200                     mov     bx,2
00002A8D 2EA0[6A19]                 mov     al,[cs:driver_units]    ; shuffle magicdrives and new drives
                                                                    ;   by this many units

                                ;BUGBUG 29-Oct-1992 bens Take this 55h out after Beta 4
00002A91 B455                       mov     ah,55h                  ; backdoor won't shuffle unless it
                                                                    ;  sees this, to prevent bad things
                                                                    ;  from happening if people run the
                                                                    ;  new driver with an old BIOS
00002A93 2EFF1E[9003]               call    far [cs:MagicBackdoor]

00002A98 5E                         pop     si
00002A99 07                         pop     es

                                ;no_more_magic_calls:
                                ;
                                ;;  endif
                                ;   retn

                                    ; 13/04/2024
                                    ;;;
                                no_more_magic_calls:
00002A9A E9E9FA                     jmp     coff
                                    ;;;

                                ;--------------------------------------------------------------------------
                                ; country command
                                ;      the syntax is:
                                ;   country=country id {,codepage {,path}}
                                ;   country=country id {,,path}   :default codepage id in dos
                                ;--------------------------------------------------------------------------
```

```
                                    ; 30/10/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 IO.SYS)
                                    ; (SYSINIT:2663h)
                                tryq:
00002A9D 80FC51                         cmp     ah,CONFIG_COUNTRY ; 'Q'
00002AA0 7403                       je      short tryq_cont
                                skip_it3:
00002AA2 E90D01                     jmp     tryf
                                tryq_cont:

                                ; 31/12/2022 - Retro DOS v4.2 (Modified MSDOS 6.21 IO.SYS)
                                ; (SYSINIT:297Eh)
                                ; 02/11/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
                                ;%if 0
                                ;ifdef      MULTI_CONFIG
00002AA5 E8741B                     call    query_user              ; query the user if config_cmd
00002AA8 72F8                       jc      short skip_it3          ; has the CONFIG_OPTION_QUERY bit set
                                ;endif
                                ;%endif ; 02/11/2022

                                    ; 31/12/2022
                                    ;xor    bx,bx
00002AAA 31C9                       xor     cx,cx
                                    ; 14/12/2022
                                    ; ds = cs
                                    ; bx = 0
                                    ;mov    byte [cs:cntry_drv],0 ; reset the drive,path to default value.
                                    ;mov    word [cs:p_code_page],0
                                    ; 31/12/2022
                                    ; cx = 0
                                    ;mov    [cntry_drv],bl ; 0
                                    ;mov    [p_code_page],bx ; 0
00002AAC 880E[DD4A]                 mov     [cntry_drv],cl ; 0
00002AB0 890E[7C22]                 mov     [p_code_page],cx ; 0

00002AB4 BF[4522]                   mov     di,cntry_parms
                                    ;xor    cx,cx   ; 31/12/2022
                                    ; 03/01/2023
                                    ;mov    dx,cx
                                do52:
00002AB7 E8AD03                     call    sysinit_parse
00002ABA 730B                       jnc     short if52              ; parse error,check error code and

00002ABC E8E000                     call    cntry_error             ; show message and end the search loop.
                                    ; 14/12/2022
                                    ; ds = cs
00002ABF C706[7A22]FFFF             mov     word [p_cntry_code],-1
                                    ;mov    word [cs:p_cntry_code],-1 ; signals that parse error.
00002AC5 EB34                       jmp     short sr52
                                if52:
00002AC7 83F8FF                     cmp     ax,_$P_RC_EOL ; 0FFFFh; end of line?
00002ACA 742F                       jz      short sr52              ; then end the search loop

                                    ;cmp    byte [cs:result_val+_$P_Result_Blk.Type],_$P_number ; numeric?
                                    ; 14/12/2022
                                    ; ds = cs
00002ACC 803E[1722]01               cmp     byte [result_val],_$P_Number
                                    ;cmp    byte [cs:result_val],_$P_Number
00002AD1 7512                       jnz     short if56

                                    ;;mov   ax,[cs:rw_dword]
                                    ;mov    ax,[cs:result_val+_$P_Result_Blk.Picked_Val]
                                    ; 14/12/2022
00002AD3 A1[1B22]                   mov     ax,[result_val+_$P_Result_Blk.Picked_Val]
00002AD6 83F901                     cmp     cx,1
00002AD9 7505                       jne     short if57

                                    ;mov    [cs:p_cntry_code],ax
                                    ; 14/12/2022
00002ADB A3[7A22]                   mov     [p_cntry_code],ax

                                    ; 30/10/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
                                    ;jmp    short en57
                                    ; 12/12/2022
                                    ;jmp    short en56
00002ADE EBD7                       jmp     short do52
                                if57:
                                    ;mov    [cs:p_code_page],ax
                                    ; 14/12/2022
                                    ; ds = cs
00002AE0 A3[7C22]                   mov     [p_code_page],ax
                                en57:
                                    ;jmp    short en56              ; path entered
                                    ; 12/12/2022
00002AE3 EBD2                       jmp     short do52
                                if56:
00002AE5 1E                         push    ds
00002AE6 06                         push    es
00002AE7 56                         push    si
00002AE8 57                         push    di

00002AE9 0E                         push    cs
00002AEA 07                         pop     es

                                    ;lds    si,[cs:rv_dword]        ; move the path to known place.
                                    ; 14/12/2022
00002AEB C536[1B22]                 lds     si,[rv_dword]
00002AEF BF[DD4A]                   mov     di,cntry_drv
00002AF2 E82C1F                     call    move_asciiz

00002AF5 5F                         pop     di
00002AF6 5E                         pop     si
00002AF7 07                         pop     es
00002AF8 1F                         pop     ds
                                en56:
00002AF9 EBBC                       jmp     short do52
                                sr52:
                                    ; 14/12/2022
                                    ; ds = cs
00002AFB 833E[7A22]FF               cmp     word [p_cntry_code],-1
                                    ;cmp    word [cs:p_cntry_code],-1     ; had a parse error?
00002B00 7509                       jne     short tryq_open
00002B02 E981FA                     jmp     coff

                                tryqbad:                            ;"invalid country code or code page"
00002B05 F9                         stc
00002B06 BA[D950]                   mov     dx,badcountry
00002B09 EB79                       jmp     tryqchkerr

                                tryq_open:
                                    ; 14/12/2022
                                    ; ds = cs
00002B0B 803E[DD4A]00               cmp     byte [cntry_drv],0
                                    ;cmp    byte [cs:cntry_drv],0
```

```
34072 00002B10 7405                     je      short tryq_def
34073 00002B12 BA[DD4A]                 mov     dx,cntry_drv
34074 00002B15 EB03                     jmp     short tryq_openit
34075
34076                                   tryq_def:
34077 00002B17 BA[DF4A]                 mov     dx,cntry_root
34078                                   tryq_openit:
34079 00002B1A B8003D                   mov     ax,3D00h                ;open a file
34080 00002B1D F9                       stc
34081 00002B1E CD21                     int     21h
34082 00002B20 7242                     jc      short tryqfilebad       ;open failure
34083
34084                                   ; 14/12/2022
34085                                   ; ds = cs
34086 00002B22 A3[5C03]                 mov     [cntryfilehandle],ax
34087                                   ;mov    [cs:cntryfilehandle],ax       ;save file handle
34088 00002B25 89C3                     mov     bx,ax
34089 00002B27 A1[7A22]                 mov     ax,[p_cntry_code]
34090 00002B2A 8B16[7C22]               mov     dx,[p_code_page]
34091                                   ;mov    ax,[cs:p_cntry_code]
34092                                   ;mov    dx,[cs:p_code_page]   ;now,ax=country id,bx=filehandle
34093                                   ;mov    cx,[cs:memhi]
34094 00002B2E 8B0E[6403]               mov     cx,[memhi]
34095 00002B32 81C18001                 add     cx,384                  ;need 6k buffer to handle country.sys
34096                                                                   ;MO23
34097                                   ; 14/12/2022
34098                                   ; ds = cs
34099 00002B36 3B0E[A502]               cmp     cx,[ALLOCLIM]
34100                                   ;cmp    cx,[cs:ALLOCLIM]
34101 00002B3A 7745                     ja      short tryqmemory        ;cannot allocate the buffer for country.sys
34102
34103 00002B3C BE[DD4A]                 mov     si,cntry_drv            ;ds:si -> cntry_drv
34104 00002B3F 803C00                   cmp     byte [si],0             ;default path?
34105 00002B42 7502                     jne     short tryq_set_for_dos
34106
34107 00002B44 46                       inc     si
34108 00002B45 46                       inc     si                      ;ds:si -> cntry_root
34109
34110                                   tryq_set_for_dos:
34111                                   ; 14/12/2022
34112                                   ; ds = cs
34113 00002B46 C43E[7902]               les     di,[sysi_country]
34114                                   ;les    di,[cs:sysi_country]  ;es:di -> country info tab in dos
34115 00002B4A 57                       push    di                      ;save di
34116                                   ;add    di,8
34117 00002B4B 83C708                   add     di,country_cdpg_info.ccPath_CountrySys ; 8
34118 00002B4E E8D01E                   call    move_asciiz             ;set the path to country.sys in dos.
34119 00002B51 5F                       pop     di                      ;es:di -> country info tab again.
34120
34121                                   ; 14/12/2022
34122 00002B52 8B0E[6403]               mov     cx,[memhi]
34123                                   ;mov    cx,[cs:memhi]
34124 00002B56 8ED9                     mov     ds,cx
34125 00002B58 31F6                     xor     si,si                   ;ds:si -> 2k buffer to be used.
34126 00002B5A E8601D                   call    setdoscountryinfo       ;now do the job!!!
34127                                   ; ds <> cs ; 14/12/2022
34128 00002B5D 7325                     jnc     short tryqchkerr        ;read error or could not find country,code page combination
34129
34130 00002B5F 83F9FF                   cmp     cx,-1                   ;could not find matching country_id,code page?
34131 00002B62 74A1                     je      short tryqbad           ;then "invalid country code or code page"
34132
34133                                   tryqfilebad:
34134 00002B64 0E                       push    cs
34135 00002B65 07                       pop     es
34136 00002B66 2E803E[DD4A]00           cmp     byte [cs:cntry_drv],0 ;is the default file used?
34137 00002B6C 7405                     je      short tryqdefbad
34138
34139 00002B6E BE[DD4A]                 mov     si,cntry_drv
34140 00002B71 EB03                     jmp     short tryqbadload
34141
34142                                   tryqdefbad:                     ;default file has been used.
34143 00002B73 BE[DF4A]                 mov     si,cntry_root           ;es:si -> \country.sys in sysinit_seg
34144                                   tryqbadload:
34145 00002B76 E8B31E                   call    badload                 ;ds will be restored to sysinit_seg
34146                                   ; 31/12/2022 - Retro DOS v4.2 (Modified MSDOS 6.21 IO.SYS)
34147                                   ; (SYSINIT:2A69h)
34148 00002B79 8B0E[A302]               mov     cx,[CONFBOT] ; ds = cs (from badload)
34149                                   ;mov    cx,[cs:CONFBOT]
34150                                   ; 30/10/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
34151                                   ;mov    cx,[cs:top_of_cdss]
34152                                   ; 11/12/2022
34153                                   ; ds = cs
34154                                   ;mov    cx,[top_of_cdss]  ; mov cx,[CONFBOT]
34155 00002B7D 8EC1                     mov     es,cx                   ;restore es -> confbot.
34156 00002B7F EB13                     jmp     short coffj4
34157
34158                                   tryqmemory:
34159 00002B81 BA[1C51]                 mov     dx,insufmemory
34160                                   tryqchkerr:
34161                                   ;mov    cx,[cs:CONFBOT]
34162                                   ; 30/10/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
34163                                   ;mov    cx,[cs:top_of_cdss]
34164                                   ; 12/12/2022
34165 00002B84 0E                       push    cs
34166 00002B85 1F                       pop     ds
34167                                   ; 31/12/2022 - Retro DOS v4.2
34168 00002B86 8B0E[A302]               mov     cx,[CONFBOT] ; (MSDOS 6.21 IO.SYS, SYSINIT)
34169                                   ;mov    cx,[top_of_cdss]  ; mov cx,[CONFBOT]
34170 00002B8A 8EC1                     mov     es,cx                   ;restore es -> confbot seg
34171                                   ;push   cs
34172                                   ;pop    ds                      ;restore ds to sysinit_seg
34173 00002B8C 7306                     jnc     short coffj4            ;if no error,then exit
34174
34175 00002B8E E8C31E                   call    print                   ;else show error message
34176 00002B91 E82603                   call    error_line
34177                                   coffj4:
34178                                   ;mov    bx,[cs:cntryfilehandle]
34179                                   ; 11/12/2022
34180                                   ; ds = cs
34181 00002B94 8B1E[5C03]               mov     bx,[cntryfilehandle]
34182 00002B98 B43E                     mov     ah,3Eh
34183 00002B9A CD21                     int     21h                     ;close a file. don't care even if it fails.
34184 00002B9C E9E7F9                   jmp     coff
34185
34186                                   ;--------------------------------------------
34187
34188                                   cntry_error:
34189
34190                                   ;function: show "invalid country code or code page" messages,or
34191                                   ;          "error in country command" depending on the error code
34192                                   ;          in ax returned by sysparse;
34193                                   ;in:ax - error code
34194                                   ;   ds - sysinitseg
34195                                   ;   es - confbot
```

```
                        ;out:       show message.  dx destroyed.
                        
00002B9F 83F806             cmp     ax,_$P_Out_Of_Range ; 6
00002BA2 7505               jne     short if64
00002BA4 BA[D950]           mov     dx,badcountry         ;"invalid country code or code page"
00002BA7 EB03               jmp     short en64
                        if64:
00002BA9 BA[FF50]           mov     dx,badcountrycom      ;"error in contry command"
                        en64:
00002BAC E8A51E             call    print
                            ;call   error_line
                            ;retn
                            ; 11/12/2022
00002BAF E90803             jmp     error_line
                        
                        ;--------------------------------------------------------------------------
                        ; files command
                        ;--------------------------------------------------------------------------
                        
                        ;**************************************************************************
                        ; function: parse the parameters of files= command.                       *
                        ;                                                                         *
                        ; input :                                                                 *
                        ;   es:si -> parameters in command line.                                  *
                        ; output:                                                                *
                        ;   variable files set.                                                   *
                        ;                                                                         *
                        ; subroutines to be called:                                              *
                        ;   sysinit_parse                                                       *
                        ; logic:                                                                 *
                        ; {                                                                       *
                        ;   set di points to files_parms;                                       *
                        ;   set dx,cx to 0;                                                     *
                        ;   while (end of command line)                                         *
                        ;   { sysinit_parse;                                                    *
                        ;     if (no error) then                                                *
                        ;        files = result_val._$P_picked_val                              *
                        ;     else                                                              *
                        ;        error exit;                                                    *
                        ;   };                                                                  *
                        ; };                                                                     *
                        ;                                                                         *
                        ;**************************************************************************
                        
                        tryf:
00002BB2 80FC46                 cmp     ah,CONFIG_FILES ;  'F'
00002BB5 7528               jne     short tryl
                        
                        ; 31/12/2022 - Retro DOS v4.2 (Modified MSDOS 6.21 IO.SYS)
                        ; (SYSINIT:2AABh)
                        ; 30/10/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
                        ;%if 0
                        ;ifdef      MULTI_CONFIG
00002BB7 E8621A             call    query_user              ; query the user if config_cmd
00002BBA 7223               jc      short tryl              ; has the CONFIG_OPTION_QUERY bit set
                        ;endif
                        ;%endif ; 30/10/2022
                        
                            ; 14/12/2022
                            ; ds = cs
                        
00002BBC BF[7E22]           mov     di,files_parms
00002BBF 31C9               xor     cx,cx
                            ; 03/01/2023
                            ;mov    dx,cx
                        do67:
00002BC1 E8A302             call    sysinit_parse
00002BC4 7303               jnc     short if67              ; parse error
                            ;call   badparm_p               ;  and show messages and end the search loop.
                            ;jmp    short sr67
                            ; 03/01/2023
00002BC6 E98D01             jmp     badparm_p_coff
                        if67:
00002BC9 83F8FF             cmp     ax,_$P_RC_EOL           ; end of line?
00002BCC 7408               je      short en67              ; then end the $endloop
                        
                            ; 14/12/2022
                            ; ds = cs
                            ;;mov   al,[cs:rv_dword]
                            ;mov    al,[cs:result_val+_$P_Result_Blk.Picked_Val]
                            ;mov    [cs:p_files],al                 ; save it temporarily
                            ;mov    al,[rv_dword]
00002BCE A0[1B22]           mov     al,[result_val+_$P_Result_Blk.Picked_Val]
00002BD1 A2[9D22]           mov     [p_files],al
                        
00002BD4 EBEB               jmp     short do67
                        en67:
                            ; 14/12/2022
                            ; ds = cs
00002BD6 A0[9D22]           mov     al,[p_files]
00002BD9 A2[9F02]           mov     [FILES],al
                            ;mov    al,[cs:p_files]
                            ;mov    [cs:FILES],al           ; no error. really set the value now.
                        sr67:
00002BDC E9A7F9             jmp     coff
                        
                        ; 04/04/2019 - Retro DOS v4.0
                        
                        ;--------------------------------------------------------------------------
                        ; lastdrive command
                        ;--------------------------------------------------------------------------
                        
                        ;**************************************************************************
                        ; function: parse the parameters of lastdrive= command.                   *
                        ;                                                                         *
                        ; input :                                                                 *
                        ;   es:si -> parameters in command line.                                  *
                        ; output:                                                                *
                        ;   set the variable num_cds.                                           *
                        ;                                                                         *
                        ; subroutines to be called:                                              *
                        ;   sysinit_parse                                                       *
                        ; logic:                                                                 *
                        ; {                                                                       *
                        ;   set di points to ldrv_parms;                                        *
                        ;   set dx,cx to 0;                                                     *
                        ;   while (end of command line)                                         *
                        ;   { sysinit_parse;                                                    *
                        ;     if (no error) then                                                *
                        ;        set num_cds to the returned value;                             *
                        ;     else /*error exit*/                                               *
                        ;        error exit;                                                    *
                        ;   };                                                                  *
                        ; };                                                                     *
```

```
                                ;                                                                               *
                                ;*******************************************************************************
                                
                                tryl:
00002BDF 80FC4C                         cmp     ah,CONFIG_LASTDRIVE ; 'L'
00002BE2 7528                       jne     short tryp
                                
                                ; 31/12/2022 - Retro DOS v4.2 (Modified MSDOS 6.21 IO.SYS)
                                ; (SYSINIT:2AE0h)
                                ; 30/10/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
                                ;%if 0
00002BE4 E8351A                     call    query_user      ; query the user if config_cmd
00002BE7 7223                       jc      short tryp      ; has the CONFIG_OPTION_QUERY bit set
                                ;endif
                                ;%endif ; 30/10/2022
                                
                                    ; 14/12/2022
                                    ; ds = cs
                                
00002BE9 BF[D522]                   mov     di,ldrv_parms
00002BEC 31C9                       xor     cx,cx
                                    ; 03/01/2023
                                    ;mov    dx,cx
                                do73:
00002BEE E87602                     call    sysinit_parse
00002BF1 7303                       jnc     short if73      ; parse error
                                    ;call   badparm_p       ;  and show messages and end the search loop.
                                    ;jmp    short sr73
                                    ; 03/01/2023
00002BF3 E96001                     jmp     badparm_p_coff
                                if73:
00002BF6 83F8FF                     cmp     ax,_$P_RC_EOL   ; end of line?
00002BF9 7408                       je      short en73      ; then end the $endloop
                                
                                    ; 14/12/2022
                                    ; ds = cs
                                    ;;mov   al,[cs:rv_dword]
                                    ;mov    al,[cs:rv_byte]         ; pick up the drive number
                                    ;mov    [cs:p_ldrv],al ; save it temporarily
                                
                                    ;mov    al,[rv_dword]
00002BFB A0[1B22]                   mov     al,[rv_byte]
00002BFE A2[E922]                   mov     [p_ldrv],al
                                
00002C01 EBEB                       jmp     short do73
                                en73:
                                    ; 14/12/2022
                                    ; ds = cs
00002C03 A0[E922]                   mov     al,[p_ldrv]
00002C06 A2[A202]                   mov     [NUM_CDS],al
                                    ;mov    al,[cs:p_ldrv]
                                    ;mov    [cs:NUM_CDS],al         ; no error. really set the value now.
                                sr73:
00002C09 E97AF9                     jmp     coff
                                
                                ;--------------------------------------------------------------------------
                                ; setting drive parameters
                                ;--------------------------------------------------------------------------
                                
                                tryp:
00002C0C 80FC50                         cmp     ah,CONFIG_DRIVPARM ; 'P'
00002C0F 7516                       jne     short tryk
                                
                                ; 31/12/2022 - Retro DOS v4.2
                                ; 30/10/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
                                ;%if 0
                                ;ifdef      MULTI_CONFIG
00002C11 E8081A                     call    query_user      ; query the user if config_cmd
00002C14 7211                       jc      short tryk      ; has the CONFIG_OPTION_QUERY bit set
                                ;endif
                                ;%endif ; 30/10/2022
                                
00002C16 E8EC0E                     call    parseline
00002C19 7209                       jc      short trypbad
00002C1B E8050E                     call    setparms
00002C1E E8470E                     call    diddleback
                                
                                ; No error check here, because setparms and diddleback have no error
                                ; returns, and setparms as coded now can return with carry set.
                                ;       jc  short trypbad
                                
                                    ; 12/12/2022
                                    ; cf = 0
                                    ; 02/11/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
                                    ;jc     short trypbad
                                
00002C21 E962F9                     jmp     coff
                                trypbad:
00002C24 E95F02                     jmp     badop
                                
                                ;--------------------------------------------------------------------------
                                ; setting internal stack parameters
                                ; stacks=m,n where
                                ;   m is the number of stacks (range 8 to 64,default 9)
                                ;   n is the stack size (range 32 to 512 bytes,default 128)
                                ; j.k. 5/5/86: stacks=0,0 implies no stack installation.
                                ;   any combinations that are not within the specified limits will
                                ;   result in "unrecognized command" error.
                                ;--------------------------------------------------------------------------
                                
                                ;*******************************************************************************
                                ;                                                                               *
                                ; function: parse the parameters of stacks= command.                                *
                                ;       the minimum value for "number of stacks" and "stack size" is  *
                                ;       8 and 32 each.  in the definition of sysparse value list,they *
                                ;       are set to 0.  this is for accepting the exceptional case of  *
                                ;       stacks=0,0 case (,which means do not install the stack.)        *
                                ;       so,after sysparse is done,we have to check if the entered       *
                                ;       values (stack_count,stack_size) are within the actual range,  *
                                ;       (or if "0,0" pair has been entered.)                            *
                                ; input :                                                                      *
                                ;   es:si -> parameters in command line.                                        *
                                ; output:                                                                      *
                                ;   set the variables stack_count,stack_size.                                   *
                                ;                                                                               *
                                ; subroutines to be called:                                                        *
                                ;   sysinit_parse                                                               *
                                ; logic:                                                                       *
                                ; {                                                                             *
                                ;   set di points to stks_parms;                                                *
                                ;   set dx,cx to 0;                                                             *
                                ;   while (end of command line)                                                 *
                                ;   { sysinit_parse;                                                            *
                                ;     if (no error) then                                                        *
```

```
34444                                   ;        { if (cx == 1) then /* first positional = stack count */       *
34445                                   ;             p_stack_count = result_val._$P_picked_val;               *
34446                                   ;          if (cx == 2) then /* second positional = stack size */       *
34447                                   ;             p_stack_size = result_val._$P_picked_val;                *
34448                                   ;        }                                                             *
34449                                   ;      else /*error exit*/                                             *
34450                                   ;         error exit;                                                  *
34451                                   ;   };                                                                *
34452                                   ;   here check p_stack_count,p_stack_size if it meets the condition;  *
34453                                   ;   if o.k.,then set stack_count,stack_size;                         *
34454                                   ;    else error_exit;                                                 *
34455                                   ; };                                                                  *
34456                                   ;*************************************************************************
34457
34458                                   tryk:
34459                                           ;if      stacksw
34460
34461 00002C27 80FC4B                          cmp     ah,CONFIG_STACKS ; 'K'
34462 00002C2A 7402                      je      short do_tryk
34463                                   skip_it4:
34464 00002C2C EB79                      jmp     short trys      ; 15/12/2022
34465                                   do_tryk:
34466
34467                                   ; 31/12/2022 - Retro DOS v4.2 (Modified MSDOS 6.21 IO.SYS)
34468                                   ; (SYSINIT:2B33h)
34469                                   ; 30/10/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
34470                                   ;%if 0
34471                                   ;ifdef      MULTI_CONFIG
34472 00002C2E E8EB19                          call query_user              ; query the user if config_cmd
34473 00002C31 72F9                            jc  short skip_it4          ; has the CONFIG_OPTION_QUERY bit set
34474                                   ;endif
34475                                   ;%endif     ; 30/10/2022
34476
34477                                       ; 14/12/2022
34478                                       ; ds = cs
34479
34480 00002C33 BF[EA22]                    mov     di,stks_parms
34481 00002C36 31C9                        xor     cx,cx
34482                                       ; 03/01/2023
34483                                       ;mov    dx,cx
34484                                   do79:
34485 00002C38 E82C02                      call    sysinit_parse
34486 00002C3B 730B                        jnc     short if79              ; parse error
34487
34488 00002C3D BA[8B51]                    mov     dx,badstack             ; "invalid stack parameter"
34489 00002C40 E8111E                      call    print                   ;  and show messages and end the search loop.
34490 00002C43 E87402                      call    error_line
34491                                       ;jmp    sr79
34492                                       ; 11/12/2022
34493 00002C46 EB39                        jmp     short sr79
34494                                   if79:
34495 00002C48 83F8FF                      cmp     ax,_$P_RC_EOL           ; end of line?
34496 00002C4B 7412                        je      short en79              ; then end the $endloop
34497
34498                                       ; 14/12/2022
34499                                       ; ds = cs
34500
34501                                       ;;mov   ax,[cs:rv_dword]
34502                                       ;mov    ax,[cs:result_val+_$P_Result_Blk.Picked_Val]
34503                                       ;mov    ax,[rv_dword]
34504 00002C4D A1[1B22]                    mov     ax,[result_val+_$P_Result_Blk.Picked_Val]
34505
34506 00002C50 83F901                      cmp     cx,1
34507 00002C53 7505                        jne     short if83
34508
34509                                       ; 14/12/2022
34510                                       ;mov    [cs:p_stack_count],ax
34511                                       ;jmp    short en83
34512 00002C55 A3[1F23]                    mov     [p_stack_count],ax
34513 00002C58 EBDE                        jmp     short do79
34514                                   if83:
34515                                       ; 14/12/2022
34516                                       ;mov    [cs:p_stack_size],ax
34517 00002C5A A3[2123]                    mov     [p_stack_size],ax
34518                                   en83:
34519 00002C5D EBD9                        jmp     short do79
34520                                   en79:
34521                                       ; 14/12/2022
34522                                       ; ds = cs
34523 00002C5F A1[1F23]                    mov     ax,[p_stack_count]
34524 00002C62 09C0                        or      ax,ax
34525 00002C64 741E                        jz      short if87
34526
34527                                       ; 14/12/2022
34528                                       ;cmp    word [p_stack_count],0
34529                                       ;;cmp   word [cs:p_stack_count],0
34530                                       ;je     short if87
34531
34532                                       ; 14/12/2022
34533 00002C66 83F808                      cmp     ax, mincount ; 8
34534                                       ;cmp    word [cs:p_stack_count],mincount ; 8
34535                                       ; 15/12/2022
34536 00002C69 721F                        jb      short en87
34537 00002C6B 833E[2123]20                cmp     word [p_stack_size],minsize ; 32
34538                                       ;cmp    word [cs:p_stack_size],minsize ; 32
34539                                       ; 15/12/2022
34540 00002C70 7218                        jb      short en87
34541                                   if94:
34542                                       ; 14/12/2022
34543                                       ; ds = cs
34544                                       ; ax = [p_stack_count]
34545                                       ;mov    ax,[p_stack_count]
34546                                       ;;mov   ax,[cs:p_stack_count]
34547 00002C72 A3[8C02]                    mov     [stack_count],ax
34548                                       ;mov    [cs:stack_count],ax
34549                                       ;mov    ax,[cs:p_stack_size]
34550 00002C75 A1[2123]                    mov     ax,[p_stack_size]
34551                                       ;mov    [cs:stack_size],ax
34552 00002C78 A3[8E02]                    mov     [stack_size],ax
34553                                       ;mov    word [cs:stack_addr],-1       ; stacks= been accepted.
34554 00002C7B C706[9002]FFFF              mov     word [stack_addr],-1
34555                                   sr79:
34556 00002C81 E902F9                      jmp     coff
34557
34558                                   if87:
34559                                       ; 14/12/2022
34560 00002C84 3906[2123]                  cmp     [p_stack_size],ax ; 0
34561 00002C88 74E8                        je      short if94 ; ax = [p_stack_count] = 0
34562                                       ;cmp    word [cs:p_stack_size],0
34563                                       ;je     short if94
34564                                   en87:
34565                                       ; 15/12/2022
34566                                       ; ([p_stack_count] is invalid, use default values)
34567                                       ; 14/12/2022
```

```
                                    ; ds = cs
00002C8A C706[8C02]0900             mov     word [stack_count],defaultcount ; 9
00002C90 C706[8E02]8000             mov     word [stack_size],defaultsize ; 128
00002C96 C706[9002]0000             mov     word [stack_addr],0
                                    ;mov    word [cs:stack_count],defaultcount ; 9
                                    ;                               ; reset to default value.
                                    ;mov    word [cs:stack_size],defaultsize ; 128
                                    ;mov    word [cs:stack_addr],0

00002C9C BA[8B51]                   mov     dx,badstack
00002C9F E8B21D                     call    print
00002CA2 E81502                     call    error_line
00002CA5 EBDA                       jmp     short sr79

                                  ; 15/12/2022
                                  %if 0
                                    mov     di,stks_parms
                                    xor     cx,cx
                                    ; 03/01/2023
                                    ;mov    dx,cx
                                  do79:
                                    call    sysinit_parse
                                    jnc     short if79              ; parse error

                                    mov     dx,badstack             ; "invalid stack parameter"
                                    call    print                   ;  and show messages and end the search loop.
                                    call    error_line
                                    ;jmp    sr79
                                    ; 11/12/2022
                                    jmp     short sr79
                                  if79:
                                    cmp     ax,_$P_RC_EOL           ; end of line?
                                    je      short en79              ; then end the $endloop

                                    ;mov    ax,[cs:rv_dword]
                                    mov     ax,[cs:result_val+_$P_Result_Blk.Picked_Val]
                                    cmp     cx,1
                                    jne     short if83

                                    mov     [cs:p_stack_count],ax
                                    jmp     short en83
                                  if83:
                                    mov     [cs:p_stack_size],ax
                                  en83:
                                    jmp     short do79
                                  en79:
                                    cmp     word [cs:p_stack_count],0
                                    je      short if87

                                    cmp     word [cs:p_stack_count],mincount ; 8
                                    jb      short ll88
                                    cmp     word [cs:p_stack_size],minsize ; 32
                                    jnb     short if88
                                  ll88:
                                    mov     word [cs:p_stack_count],-1 ; invalid
                                  if88:
                                    jmp     short en87

                                    ; 11/12/2022
                                  if94:
                                    mov     ax,[cs:p_stack_count]
                                    mov     [cs:stack_count],ax
                                    mov     ax,[cs:p_stack_size]
                                    mov     [cs:stack_size],ax
                                    mov     word [cs:stack_addr],-1       ; stacks= been accepted.
                                  sr79:
                                    jmp     coff

                                  if87:
                                    cmp     word [cs:p_stack_size],0
                                    je      short en87
                                    mov     word [cs:p_stack_count],-1 ; invalid
                                  en87:
                                    cmp     word [cs:p_stack_count],-1 ; invalid?
                                    jne     short if94

                                    mov     word [cs:stack_count],defaultcount ; 9
                                                                    ; reset to default value.
                                    mov     word [cs:stack_size],defaultsize ; 128
                                    mov     word [cs:stack_addr],0

                                    mov     dx,badstack
                                    call    print
                                    call    error_line
                                    jmp     short sr79

                                  %endif

                                  ; 11/12/2022
                                  %if 0
                                  if94:
                                    mov     ax,[cs:p_stack_count]
                                    mov     [cs:stack_count],ax
                                    mov     ax,[cs:p_stack_size]
                                    mov     [cs:stack_size],ax
                                    mov     word [cs:stack_addr],-1       ; stacks= been accepted.
                                  sr79:
                                    jmp     coff
                                  %endif
                                    ;endif

                                  ;--------------------------------------------------------------------------
                                  ; shell command
                                  ;--------------------------------------------------------------------------

                                  trys:
                                          cmp     ah,CONFIG_SHELL ; 'S'
00002CA7 80FC53                     jne     short tryx
00002CAA 755A

                                  ; 31/12/2022 - Retro DOS v4.2 (Modified MSDOS 6.21 IO.SYS)
                                  ; (SYSINIT:2BE1h)
                                  ; 30/10/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
                                  ;%if 0
                                  ;ifdef     MULTI_CONFIG
00002CAC E86D19                     call    query_user              ; query the user if config_cmd
00002CAF 7255                       jc      short tryx              ; has the CONFIG_OPTION_QUERY bit set
                                    ; 14/04/2024
                                    ; ds = cs
                                    ;mov    byte [cs:newcmd],1
00002CB1 C606[2A4B]01               mov     byte [newcmd],1
                                  ;endif
                                  ;%endif ; 30/10/2022

                                    ;;mov  word [cs:command_line],0 ; zap length,first byte of command-line
```

```
                                    ; 30/10/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
                                    ;mov    byte [cs:command_line+1],0
                                    ; 15/12/2022
                                    ; ds = cs
                                    ; 08/09/2023
                                    ;mov    byte [command_line+1],0
00002CB6 C706[BB4B]0000             mov     word [command_line],0 ; zap length,first byte of command-line

00002CBC BF[2E4B]                       mov     di,commnd+1             ; we already have the first char
00002CBF 8845FF                         mov     [di-1],al               ; of the new shell in AL, save it now
                                storeshell:
00002CC2 E8EA1A                     call    getchr
00002CC5 08C0                           or      al,al                   ; this is the normal case: "organize"
00002CC7 741C                           jz  short getshparms        ; put a ZERO right after the filename

00002CC9 3C20                           cmp     al," "                  ; this may happen if there are no args
00002CCB 7209                           jb  short endofshell        ; I suppose...
00002CCD 8805                       mov     [di],al
00002CCF 47                         inc     di
                                        ;cmp    di,commnd+63            ; this makes sure we don't overflow
                                        ;jb short storeshell        ; commnd (the filename)
                                        ;jmp       short endofshell
                                    ; 30/10/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
                                    ;jmp    short storeshell
                                    ; 03/01/2023
00002CD0 81FF[6C4B]                 cmp     di,commnd+63            ; this makes sure we don't overflow
00002CD4 72EC                           jb  short storeshell        ; commnd (the filename)
                                    ;jmp    short endofshell

                                ; 30/10/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
                                ;getshparms:
                                ;   mov     byte [di],0             ; zero-terminate the filename
                                ;   mov     di,command_line+1       ; prepare to process the command-line
                                ;
                                ;parmloop:
                                ;   call    getchr
                                ;   cmp     al," "
                                ;   jb      short endofparms
                                ;   mov     [di],al
                                ;   inc     di
                                ;   cmp     di,command_line+126
                                ;   jb      short parmloop
                                ;endofparms:
                                ;   mov     cx,di
                                ;   sub     cx,command_line+1
                                ;   mov     [cs:command_line],cl
                                ;
                                ;endofshell:
                                ;   mov     byte [di],0             ; zero-terminate the filename (or
                                ;                                   ; the command-line as the case may be)
                                ;skipline:
                                ;       cmp     al,lf       ; 0Ah           ; the safest way to eat the rest of
                                ;       je  short endofline             ; the line: watch for ever-present LF
                                ;call       getchr
                                ;       jnc short skipline          ; keep it up as long as there are chars
                                ;
                                ;endofline:
                                ;       jmp     conflp

                                    ; 30/10/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
                                endofshell:
                                        mov     byte [di],0             ; zero-terminate the filename (or
                                                                        ; the command-line as the case may be)
                                    ; 11/07/2023 - Retro DOS v4.2 IO.SYS (optimization)
                                    ; MSDOS 6.21 IO.SYS - SYSINIT:2C33h
                                    ; PCDOS 7.1 IBMBIO.COM - SYSINIT:314Eh
                                    ;call   getchr
                                skipline:           ; MSDOS 6.21 IO.SYS - SYSINIT:2C33h
00002CD9 3C0A                       cmp     al,lf   ; 0Ah           ; the safest way to eat the rest of
00002CDB 7405                       je      short endofline             ; the line: watch for ever-present LF
00002CDD E8CF1A                     call    getchr
                                    ; 03/01/2023 - Retro DOS v4.2 (Modified MSDOS 6.1 IO.SYS)
                                    ; (SYSINIT:2C3Ah)
00002CE0 73F7                       jnb     short skipline

                                endofline:
00002CE2 E94AF8                     jmp     conflp

                                    ; 30/10/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
                                getshparms:
                                    ; 18/12/2022
                                    ; al = 0
00002CE5 8805                       mov     [di],al ; 0
                                    ;mov    byte [di],0             ; zero-terminate the filename
00002CE7 BF[BC4B]                   mov     di,command_line+1       ; prepare to process the command-line
                                parmloop:
00002CEA E8C21A                     call    getchr
00002CED 3C20                       cmp     al," " ; 20h
                                    ;jb     short endofshell
                                    ; 03/01/2023
00002CEF 7209                       jb      short endofparms

00002CF1 8805                       mov     [di],al
00002CF3 47                         inc     di
                                    ;jmp    short parmloop
                                    ; 03/01/2023 - Retro DOS v4.2
00002CF4 81FF[394C]                 cmp     di,command_line+126
00002CF8 72F0                       jb      short parmloop

                                    ; 03/01/2023 - Retro DOS v4.2
                                endofparms:
00002CFA 89F9                       mov     cx,di
00002CFC 81E9[BC4B]                 sub     cx,command_line+1
                                    ;mov    [cs:command_line],cl
                                    ; 03/01/2023
00002D00 880E[BB4B]                 mov     [command_line],cl
00002D04 EBD0                       jmp     short endofshell

                                ;--------------------------------------------------------------------------
                                ; fcbs command
                                ;--------------------------------------------------------------------------

                                ;**************************************************************************
                                ; function: parse the parameters of fcbs= command.                        *
                                ;                                                                        *
                                ; input :                                                                *
                                ;   es:si -> parameters in command line.                                 *
                                ; output:                                                               *
                                ;   set the variables fcbs,keep.                                         *
                                ;                                                                        *
                                ; subroutines to be called:                                             *
                                ;   sysinit_parse                                                        *
                                ; logic:                                                                *
                                ; {                                                                      *
```

```
34816                                   ;   set di points to fcbs_parms;                                    *
34817                                   ;   set dx,cx to 0;                                                 *
34818                                   ;   while (end of command line)                                     *
34819                                   ;   { sysparse;                                                    *
34820                                   ;     if (no error) then                                            *
34821                                   ;        { if (cx == 1) then /* first positional = fcbs */             *
34822                                   ;             fcbs = result_val._$P_picked_val;                    *
34823                                   ;          if (cx == 2) then /* second positional = keep */         *
34824                                   ;             keep = result_val._$P_picked_val;                    *
34825                                   ;        }                                                          *
34826                                   ;     else /*error exit*/                                           *
34827                                   ;        error exit;                                                *
34828                                   ;   };                                                             *
34829                                   ; };                                                               *
34830                                   ;*********************************************************************
34831                                   
34832                                   tryx:
34833 00002D06 80FC58                         cmp     ah,CONFIG_FCBS  ; 'X'
34834                                       ; 31/12/2022 - Retro DOS v4.2 (Modified MSDOS 6.21 IO.SYS)
34835 00002D09 7534                       jne     short try1
34836                                       ; 30/10/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
34837                                       ;jne    short tryy      ; comment command
34838                                   
34839                                   ; 31/12/2022 - Retro DOS v4.2
34840                                   ; 30/10/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
34841                                   ;%if 0
34842                                   ;ifdef      MULTI_CONFIG
34843 00002D0B E80E19                      call    query_user      ; query the user if config_cmd
34844 00002D0E 722F                       jc      short try1      ; has the CONFIG_OPTION_QUERY bit set
34845                                   ;endif
34846                                   ;%endif ; 30/10/2022
34847                                   
34848 00002D10 BF[9E22]                   mov     di,fcbs_parms
34849 00002D13 31C9                       xor     cx,cx
34850                                       ; 03/01/2023
34851                                       ;mov    dx,cx
34852                                   do98:
34853 00002D15 E84F01                      call    sysinit_parse
34854                                          ; 03/01/2023
34855                                       ;jnc    short if98      ; parse error
34856                                          ;call      badparm_p      ;  and show messages and end the search loop.
34857                                       ;jmp    short sr98
34858                                       ;-----------------------
34859                                       ; 03/01/2023 - Retro DOS v4.2 (Modified MSDOS 6.21 IO.SYS)
34860 00002D18 723C                       jc      short badparm_p_coff
34861                                   if98:
34862 00002D1A 83F8FF                      cmp     ax,_$P_RC_EOL  ; end of line?
34863 00002D1D 7412                       je      short en98      ; then end the $endloop
34864                                   
34865                                       ;;mov   al,[cs:rv_dword]
34866                                       ;mov    al,[cs:result_val+_$P_Result_Blk.Picked_Val]
34867                                       ; 15/12/2022
34868                                       ; ds = cs
34869 00002D1F A0[1B22]                   mov     al,[result_val+_$P_Result_Blk.Picked_Val]
34870 00002D22 83F901                     cmp     cx,1            ; the first positional?
34871 00002D25 7505                       jne     short if102
34872                                       ;mov    [cs:p_fcbs],al
34873                                       ; 15/12/2022
34874 00002D27 A2[D322]                   mov     [p_fcbs],al
34875                                       ;jmp    short en102
34876 00002D2A EBE9                       jmp     short do98
34877                                   if102:
34878                                       ;mov    [cs:p_keep],al
34879                                       ; 15/12/2022
34880 00002D2C A2[D422]                   mov     [p_keep],al
34881                                   en102:
34882 00002D2F EBE4                       jmp     short do98
34883                                   en98:
34884                                       ; 15/12/2022
34885                                       ; ds = cs
34886 00002D31 A0[D322]                   mov     al,[p_fcbs]
34887 00002D34 A2[A002]                   mov     [FCBS],al
34888 00002D37 C606[A102]00               mov     byte [KEEP],0
34889                                       ;mov    al,[cs:p_fcbs]  ; M017
34890                                       ;mov    [cs:FCBS],al    ; M017
34891                                       ;mov    byte [cs:KEEP],0 ; M017
34892                                   sr98:
34893 00002D3C E947F8                     jmp     coff
34894                                   
34895                                   ; 31/12/2022 - Retro DOS v4.2
34896                                   %if 0
34897                                   
34898                                   ; 30/10/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
34899                                   ;-----------------------------------------------------------------------
34900                                   ; comment= do nothing. just decrease chrptr,and increase count for correct
34901                                   ;          line number
34902                                   ;-----------------------------------------------------------------------
34903                                   
34904                                   tryy:
34905                                       cmp     ah,CONFIG_COMMENT ; 'Y'
34906                                       jne     short try0
34907                                   
34908                                   donothing:
34909                                       ; 15/12/2022
34910                                       ; ds = cs
34911                                       dec     word [chrptr]
34912                                       inc     word [count]
34913                                       ; 02/11/2022
34914                                       ;dec    word [cs:chrptr]
34915                                       ;inc    word [cs:count]
34916                                   
34917                                       jmp     coff
34918                                   
34919                                   ;-----------------------------------------------------------------------
34920                                   ; rem command
34921                                   ;-----------------------------------------------------------------------
34922                                   
34923                                   try0:                           ; do nothing with this line.
34924                                       cmp     ah,CONFIG_REM ; '0'
34925                                       je      short donothing
34926                                   
34927                                   %endif
34928                                   
34929                                   ; 07/04/2019 - Retro DOS v4.0
34930                                   
34931                                   ;-----------------------------------------------------------------------
34932                                   ; switches command
34933                                   ;-----------------------------------------------------------------------
34934                                   
34935                                   ;*********************************************************************
34936                                   ;                                                                    *
34937                                   ; function: parse the option switches specified.                     *
34938                                   ; note - this command is intended for the future use also.              *
34939                                   ; when we need to set system data flag,use this command.              *
```

```
                                ;                                                       *
                                ; input :                                               *
                                ;   es:si -> parameters in command line.                *
                                ; output:                                               *
                                ;   p_swit_k set if /k option chosen.                   *
                                ;                                                       *
                                ; subroutines to be called:                                 *
                                ;   sysinit_parse                                       *
                                ; logic:                                                *
                                ; {                                                     *
                                ;   set di points to swit_parms;  /*parse control definition*/    *
                                ;   set dx,cx to 0;                                     *
                                ;   while (end of command line)                         *
                                ;   { sysinit_parse;                                    *
                                ;     if (no error) then                                *
                                ;          if (result_val._$P_synonym_ptr == swit_k) then             *
                                ;              p_swit_k = 1                             *
                                ;          endif                                        *
                                ;     else {show error message;error exit}                  *
                                ;   };                                                  *
                                ; };                                                    *
                                ;                                                       *
                                ;***************************************************************

                                SUPPRESS_WINA20    EQU 00000010b  ; M025 ; (DOSSYM.INC, MSDOS 6.0)

                                try1:
00002D3F 80FC31                         cmp     ah,CONFIG_SWITCHES ; '1'
00002D42 7403                       je      short do_try1  ; switches= command entered?
                                skip_it5:
                                    ; 31/12/2022 - Retro DOS v4.2 (Modified MSDOS 6.21 IO.SYS)
                                    ; (SYSINIT:2C8Ah)
00002D44 E98E00                     jmp     tryv
                                    ; 30/10/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
                                    ;jmp    tryz

                                do_try1:
                                ; 31/12/2022 - Retro DOS v4.2
                                ; 30/10/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
                                ;%if 0
                                ;ifdef      MULTI_CONFIG
00002D47 E8D218                     call    query_user      ; query the user if config_cmd
00002D4A 72F8                       jc      short skip_it5 ; has the CONFIG_OPTION_QUERY bit set
                                ;endif
                                ;%endif ; 30/10/2022

00002D4C BF[3823]                   mov     di,swit_parms
00002D4F 31C9                       xor     cx,cx
                                    ; 03/01/2023
                                    ;mov    dx,cx
                                do110:
00002D51 E81301                     call    sysinit_parse
00002D54 7306                       jnc     short if110    ; parse error
                                    ;call   badparm_p      ;  and show messages and end the search loop.
                                    ;jmp    short sr110
                                    ; ----------------------
                                    ; 03/01/2023 - Retro DOS v4.2 (Modified MSDOS 6.21 IO.SYS)
                                badparm_p_coff:
00002D56 E83901                     call    badparm_p
00002D59 E92AF8                     jmp     coff
                                    ;------------------------
                                if110:
00002D5C 83F8FF                     cmp     ax,_$P_RC_EOL  ; end of line?
00002D5F 743C                       je      short en110    ; then jmp to $endloop for semantic check

                                    ; 15/12/2022
                                    ; ds = cs
                                    ;;cmp   word [cs:result_val_swoff],swit_k
                                    ;cmp    word [cs:result_val+_$P_Result_Blk.SYNONYM_Ptr],swit_k
00002D61 813E[1922][5623]           cmp     word [result_val+_$P_Result_Blk.SYNONYM_Ptr],swit_k
00002D67 7507                       jne     short if115    ;                               ;M059
                                    ; 15/12/2022
00002D69 C606[9523]01               mov     byte [p_swit_k],1
                                    ;mov    byte [cs:p_swit_k],1  ; set the flag
00002D6E EBE1                       jmp     short do110
                                if115:
                                    ; 15/12/2022                                                ;M059
                                    ;;cmp   word [cs:result_val_swoff],swit_t
                                    ;cmp    word [cs:result_val+_$P_Result_Blk.SYNONYM_Ptr],swit_t     ;M059
00002D70 813E[1922][7A23]           cmp     word [result_val+_$P_Result_Blk.SYNONYM_Ptr],swit_t
                                    ;jne    short if116                                 ;M059 M063
                                    ; 14/04/2024
                                    ;;;
00002D76 7507                       jne     short if118  ; (PCDOS 7.1 IBMBIO.COM)
                                    ;;;
                                    ; 15/12/2022
00002D78 C606[9623]01               mov     byte [p_swit_t],1
                                    ;mov    byte [cs:p_swit_t],1                        ;M059
00002D7D EBD2                       jmp     short do110                                 ;M059

                                    ; 14/04/2024 - Retro DOS v5.0 (PCDOS 7.1 IBMBIO.COM)
                                    ;;;
                                if118:
                                    ;;cmp   word [cs:result_val_swoff],swit_i ; offset "/I"
                                    ;cmp    word [cs:result_val+_$P_Result_Blk.SYNONYM_Ptr],swit_i
00002D7F 813E[1922][9223]           cmp     word [result_val+_$P_Result_Blk.SYNONYM_Ptr],swit_i
00002D85 7507                       jne     short if116
                                    ;mov    byte [cs:p_swit_i],1  ; set the flag
00002D87 C606[9823]01               mov     byte [p_swit_i],1
00002D8C EBC3                       jmp      short do110
                                    ;;;
                                if116:
                                    ; 15/12/2022
                                    ;;cmp   word [cs:result_val_swoff],swit_w
                                    ;cmp    word [cs:result_val+_$P_Result_Blk.SYNONYM_Ptr],swit_w     ;M063
00002D8E 813E[1922][8623]           cmp     word [result_val+_$P_Result_Blk.SYNONYM_Ptr],swit_w
00002D94 75BB                       jne     short do110                                 ;M063
                                    ; 15/12/2022
00002D96 C606[9723]01               mov     byte [p_swit_w],1
                                    ;mov    byte [cs:p_swit_w],1                        ;M063
00002D9B EBB4                       jmp     short do110                                 ;M063
                                en110:
                                    ; 15/12/2022
                                    ; ds = cs
00002D9D 803E[9523]01               cmp     byte [p_swit_k],1
                                    ;cmp    byte [cs:p_swit_k],1  ; if /k entered,
00002DA2 1E                         push    ds
                                    ;;mov   ax,Bios_Data
                                    ;mov    ax,KERNEL_SEGMENT ; 0070h
                                    ; 21/10/2022
00002DA3 B87000                     mov     ax,DOSBIODATASEG ; 0070h
00002DA6 8ED8                       mov     ds,ax
00002DA8 750A                       jne     short if117
```

```
                                    ; 14/04/2024
00002DAA C606[7E04]00               mov     byte [keyrd_func],0 ; 4E5h ; use the conventional keyboard functions
                                                ; BIOSDATA:047Eh for PCDOS 7.1 IBMBIO.COM
00002DAF C606[7F04]01               mov     byte [keysts_func],1 ; 4E6h (for MSDOS 6.21 IO.SYS)
                                                ; BIOSDATA:047Fh for PCDOS 7.1 IBMBIO.COM
                                if117:
                                    ; 15/12/2022
                                    ; ds <> cs
00002DB4 2EA0[9623]                 mov     al,[cs:p_swit_t]                            ;M059
00002DB8 A2[8B04]                   mov     [t_switch],al  ; 4F2h (for MSDOS 6.21 IO.SYS)       ;M059
                                                ; 14/04/2024   ; BIOSDATA:048Bh for PCDOS 7.1 IBMBIO.COM
00002DBB 2E803E[9723]00             cmp     byte [cs:p_swit_w],0                        ;M063
00002DC1 740E                       je      short skip_dos_flag                         ;M063
00002DC3 06                         push    es
00002DC4 53                         push    bx
00002DC5 B452                       mov     ah,GET_IN_VARS ; 52h                        ;M063
00002DC7 CD21                       int     21h                                         ;M063
                                                ; DOS - 2+ internal - GET LIST OF LISTS
                                                ; Return: ES:BX -> DOS list of lists
                                    ;or     bytes [es:86h],2
00002DC9 26800E860002               or      byte [es:DOS_FLAG_OFFSET],SUPPRESS_WINA20 ; 2       ;M063
00002DCF 5B                         pop     bx
00002DD0 07                         pop     es
                                skip_dos_flag:                                          ;M063
00002DD1 1F                         pop     ds
                                sr110:
00002DD2 E9B1F7                     jmp     coff

                                ; 31/12/2022 - Retro DOS v4.2 (Modified MSDOS 6.21 IO.SYS)
                                ; (SYSINIT:2D14h)
                                ; 30/10/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
                                ;%if 0

                                tryv:

                                ;ifdef      MULTI_CONFIG
                                ;---------------------------------------------------------------------------
                                ; set command (as in "set var=value<cr/lf>")
                                ;---------------------------------------------------------------------------

00002DD5 80FC56                     cmp     ah,CONFIG_SET  ; 'V'
00002DD8 750F                       jne     short tryn
00002DDA E83F18                     call    query_user      ; query the user if config_cmd
00002DDD 720A                       jc      short tryn     ; has the CONFIG_OPTION_QUERY bit set
00002DDF E83614                     call    copy_envvar     ; copy var at ES:SI to "config_wrkseg"
00002DE2 73EE                       jnc     short sr110    ; no error
                                err:
00002DE4 E8D300                     call    error_line      ; whoops, display error in line XXX
00002DE7 EBE9                       jmp     short sr110     ; jump to coff (to skip to next line)

                                ;---------------------------------------------------------------------------
                                ; numlock command (as in "numlock=on|off")
                                ;---------------------------------------------------------------------------
                                tryn:
00002DE9 80FC4E                     cmp     ah,CONFIG_NUMLOCK  ;'N'
                                    ;jne    short tryy
                                    ; 14/04/2024 - Retro DOS v5.0 (PCDOS 7.1 IBMBIO.COM)
00002DEC 750C                       jne     short tryt

00002DEE E82B18                     call    query_user      ; query the user if config_cmd
00002DF1 7238                       jc      short tryy     ; has the CONFIG_OPTION_QUERY bit set
00002DF3 E8B710                     call    set_numlock
00002DF6 72EC                       jc      short err
00002DF8 EBD8                       jmp     short sr110    ; all done

                                ;endif      ;MULTI_CONFIG

                                ; 14/04/2024 - Retro DOS v5.0 (PCDOS 7.1 IBMBIO.COM)
                                ;---------------------------------------------------------------------------
                                ; dosdata command
                                ;---------------------------------------------------------------------------
                                tryt:
                                    ;cmp    ah,54h ; 'T'
00002DFA 80FC54                     cmp     ah,CONFIG_DOSDATA ; 'T' ; PCDOS 7 new config cmd
00002DFD 752C                       jne     short tryy

00002DFF E81A18                     call    query_user
00002E02 7227                       jc      short tryy

00002E04 BF[C823]                   mov     di,dosdata_parms
00002E07 31C9                       xor     cx,cx
                                    ; 14/04/2024 - Retro DOS v5.0
                                    ;mov    dx,cx ; 0
                                do120:
00002E09 E85B00                     call    sysinit_parse
00002E0C 7303                       jnc     short if120

                                    ;call   badparm_p
                                    ;jmp    short en120
                                    ; 14/04/2024 - Retro DOS v5.0
00002E0E E945FF                     jmp     badparm_p_coff
                                if120:
                                    ;cmp    ax,0FFFFh
00002E11 83F8FF                     cmp     ax,_$P_RC_EOL ; -1 ; end of line?
00002E14 7422                       jz      short en120
00002E16 803E[1822]01               cmp     byte [result_val_itag],1 ; tag 1 (UMB)
                                                ; [result_val+_$P_Result_Blk.Item_Tag]
00002E1B 7507                       jnz     short if121
00002E1D C606[6E03]01               mov     byte [dosdata_umb],1 ; DOSDATA=UMB (1) NOUMB (0)
                                    ;jmp    short sr120
                                    ; 14/04/2024
00002E22 EBE5                       jmp     short do120
                                if121:
00002E24 C606[6E03]00               mov     byte [dosdata_umb],0 ; DOSDATA=UMB (1) NOUMB (0)
                                sr120:
00002E29 EBDE                       jmp     short do120
                                    ; 14/04/2024
                                ;en120:
                                    ;jmp    coff

                                ; 30/10/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
                                ;---------------------------------------------------------------------------
                                ; comment= do nothing. just decrease chrptr,and increase count for correct
                                ;           line number
                                ;---------------------------------------------------------------------------

                                    ; 31/12/2022
                                tryy:
00002E2B 80FC59                     cmp     ah,CONFIG_COMMENT ; 'Y'
00002E2E 750B                       jne     short try0

                                donothing:
                                    ; 15/12/2022
                                    ; ds = cs
```

```
35188 00002E30 FF0E[5A03]               dec     word [chrptr]
35189 00002E34 FF06[5603]               inc     word [count]
35190                                   ; 02/11/2022
35191                                   ;dec    word [cs:chrptr]
35192                                   ;inc    word [cs:count]
35193                                en120:             ; 14/04/2024
35194 00002E38 E94BF7                   jmp     coff
35195
35196                                ;--------------------------------------------------------------------------
35197                                ; rem command
35198                                ;--------------------------------------------------------------------------
35199
35200                                try0:                           ; do nothing with this line.
35201 00002E3B 80FC30                   cmp     ah,CONFIG_REM ; '0'
35202 00002E3E 74F0                     je      short donothing
35203
35204                                ;%endif    ; 30/10/2022
35205
35206                                ; 30/10/2022
35207                                ; (MSSOS 5.0 IO.SYS - SYSINIT:29D7h)
35208
35209                                ;--------------------------------------------------------------------------
35210                                ; bogus command
35211                                ;--------------------------------------------------------------------------
35212
35213                                tryz:
35214 00002E40 80FCFF                       cmp     ah,0FFh           ;null command? (BUGBUG - who sets FFh anyway?)
35215                                   ; 31/12/2022
35216 00002E43 74EB                     je      short donothing
35217                                   ; 02/11/2022
35218                                   ;je     short tryz_donothing
35219
35220 00002E45 FF0E[5A03]               dec     word [chrptr]
35221 00002E49 FF06[5603]               inc     word [count]
35222 00002E4D EB37                     jmp     short badop
35223
35224                                ; 31/12/2022
35225                                ;   ; 02/11/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 IO.SYS)
35226                                ;tryz_donothing:
35227                                ;   jmp     donothing
35228
35229                                ; 07/04/2019 - Retro DOS v4.0
35230
35231                                ;--------------------------------------------------------------------------
35232
35233                                ; 31/12/2022 - Retro DOS v4.2 (Modified MSDOS 6.21 IO.SYS)
35234                                ; (SYSINIT:2D5Dh)
35235
35236                                ; 11/07/2023 - Retro DOS v4.2 IO.SYS (optimization)
35237
35238                                ; 30/10/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
35239                                ;
35240                                ;***CheckProtmanArena -- special hack for adjusting alloclim with Protman$
35241                                ;
35242                                ;   adjusts alloclim if Protman$ reduced our arena through a manual hack.
35243                                ;
35244                                CheckProtmanArena:
35245                                    ; 08/09/2023
35246                                    ; ds = cs
35247 00002E4F 06                      push    es
35248                                   ;mov    ax,[cs:area]   ; get our arena header
35249 00002E50 A1[6803]                 mov     ax,[area] ; 08/09/2023
35250 00002E53 48                       dec     ax
35251 00002E54 8EC0                     mov     es,ax
35252                                   ;add    ax,[es:ARENA.SIZE]
35253 00002E56 2603060300               add     ax,[es:3]       ; find end of arena
35254 00002E5B 40                       inc     ax
35255                                   ; 08/09/2023
35256 00002E5C 3B06[A502]               cmp     ax,[ALLOCLIM]
35257                                   ;cmp    ax,[cs:ALLOCLIM] ; is it less than alloclim?
35258 00002E60 7703                     ja      short CheckProtmanDone
35259
35260                                   ;mov    [cs:ALLOCLIM],ax ; reduce alloclim then
35261                                   ; 08/09/2023
35262 00002E62 A3[A502]                 mov     [ALLOCLIM],ax
35263                                CheckProtmanDone:
35264 00002E65 07                       pop     es
35265 00002E66 C3                       retn
35266
35267                                ;--------------------------------------------------------------------------
35268
35269                                sysinit_parse:
35270
35271                                ;--------------------------------------------------------------------------
35272                                ;set up registers for sysparse
35273                                ;in)es:si -> command line in confbot
35274                                ;   di -> offset of the parse control definition.
35275                                ;
35276                                ;out)       calls sysparse.
35277                                ;   carry will set if parse error.
35278                                ;   *** the caller should check the eol condition by looking at ax
35279                                ;   *** after each call.
35280                                ;   *** if no parameters are found,then ax will contain a error code.
35281                                ;   *** if the caller needs to look at the synomym@ of the result,
35282                                ;   ***  the caller should use cs:@ instead of es:@.
35283                                ;   cx register should be set to 0 at the first time the caller calls this
35284                                ;    procedure.
35285                                ;   ax - exit code
35286                                ;   bl - terminated delimeter code
35287                                ;   cx - new positional ordinal
35288                                ;   si - set to pase scanned operand
35289                                ;   dx - selected result buffer
35290                                ;--------------------------------------------------------------------------
35291
35292                                    ; 03/01/2023 - Retro DOS v4.2 (Modified MSDOS 6.21 IO.SYS)
35293                                    ; (SYSINIT:2D78h)
35294
35295                                    ; 14/04/2024 - Retro DOS v5.0
35296                                    ; (PCDOS 7.1 IBMBIO.COM - SYSINIT:32F3h)
35297
35298                                    ; 09/09/2023 - Retro DOS v4.2 IO.SYS (Optimization)
35299                                    ; ds = cs
35300 00002E67 8C06[6D19]               mov     [badparm_seg],es      ;save the pointer to the parm
35301 00002E6B 8936[6B19]               mov     [badparm_off],si      ;we are about to parse for badparm msg.
35302
35303                                    ; 24/10/2022
35304 00002E6F 06                       push    es                    ;save es,ds
35305 00002E70 1E                       push    ds
35306
35307 00002E71 06                       push    es
35308 00002E72 1F                       pop     ds                    ;now ds:si -> command line
35309
35310 00002E73 0E                       push    cs
35311 00002E74 07                       pop     es                    ;now es:di -> control definition
```

```
                                    ; 09/09/2023
                                    ;mov    [cs:badparm_seg],ds   ;save the pointer to the parm
                                    ;mov    [cs:badparm_off],si   ;we are about to parse for badparm msg.

                                    ;mov    dx,0
                                    ; 04/01/2023
00002E75 29D2                       sub     dx,dx ; 0
00002E77 E89BEB                     call    SysParse
                                    ;cmp    ax,_$P_No_Error       ; 0    ;no error
                                    ; 06/09/2023
00002E7A 21C0                       and     ax,ax

                                ;**cas note: when zero true after cmp, carry clear

                                    ;je     short ll4
                                    ; 24/10/2022 (MSDOS 5.0 IO.SYS compatibility, SYSINIT:2A02h)
                                    ; 12/12/2022
00002E7C 7405                       je      short en4 ; cf=0
00002E7E 83F8FF                     cmp     ax,_$P_RC_EOL ; 0FFFFh;or the end of line?
                                    ;jne    short if4
                                    ; 12/12/2022
00002E81 7400                       je      short en4 ; cf=0
                                    ; 06/09/2023
                                    ; cf=1

                                ; 12/12/2022
                                ;ll4:
                                ;   ; 12/12/2022
                                ;   ; cf=0
                                ;   ;clc
                                ;   jmp     short en4

                                if4:
                                    ; 24/10/2022
                                    ; 06/09/2023 (cf=1)
                                    ;stc
                                en4:
00002E83 1F                         pop     ds
00002E84 07                         pop     es
00002E85 C3                         retn

                                ; 11/12/2022
                                %if 0

                                ;--------------------------------------------------------------------------
                                ;
                                ; procedure : badop_p
                                ;
                                ;             same thing as badop,but will make sure to set ds register back
                                ;             to sysinitseg and return back to the caller.
                                ;
                                ;--------------------------------------------------------------------------

                                badop_p:
                                    push    cs
                                    pop     ds              ;set ds to configsys seg.
                                    mov     dx,badopm
                                    call    print
                                       ;call      error_line
                                    ;retn
                                    ; 11/12/2022
                                    jmp     error_line

                                %endif

                                ;--------------------------------------------------------------------------
                                ;
                                ; label : badop
                                ;
                                ;--------------------------------------------------------------------------

                                badop:
00002E86 BA[4C50]                   mov     dx,badopm       ;want to print command error "unrecognized command..."
00002E89 E8C81B                     call    print
00002E8C E82B00                     call    error_line      ;show "error in config.sys ..." .
00002E8F E9F4F6                     jmp     coff

                                ;--------------------------------------------------------------------------
                                ;
                                ; procedure : badparm_p
                                ;
                                ;             show "bad command or parameters - xxxxxx"
                                ;             in badparm_seg,badparm_off -> xxxxx
                                ;
                                ;--------------------------------------------------------------------------

                                    ; 24/10/2022
                                badparm_p:
                                    ; 11/12/2022
                                    ; ds = cs
                                    ; 11/12/2022
                                    ;push   ds ; *
00002E92 52                         push    dx
00002E93 56                         push    si

                                    ; 11/12/2022
                                    ; ds = cs
                                    ;push   cs
                                    ;pop    ds

00002E94 BA[7350]                   mov     dx,badparm
00002E97 E8BA1B                     call    print                   ; "bad command or parameters - "
00002E9A C536[6B19]                 lds     si,[badparm_ptr]

                                ;   print "xxxx" until cr.

                                do1:
00002E9E 8A14                       mov     dl,[si]                 ; get next character
00002EA0 80FA0D                     cmp     dl,cr ; 0Dh             ; is a carriage return?
00002EA3 7407                       je      short en1               ; exit loop if so

00002EA5 B402                       mov     ah,2 ; STD_CON_OUTPUT   ; function 2
00002EA7 CD21                       int     21h                     ; display character
00002EA9 46                         inc     si                      ; next character
00002EAA EBF2                       jmp     short do1
                                en1:
00002EAC 0E                         push    cs
00002EAD 1F                         pop     ds

00002EAE BA[7050]                   mov     dx,crlfm
00002EB1 E8A01B                     call    print
00002EB4 E80300                     call    error_line
```

```
                                    pop     si
                                    pop     dx
                                    ; 11/12/2022
                                    ;pop    ds ; *
                                badparmp_ret:
                                    retn

                                ; 10/09/2023 - Retro DOS v4.2 IO.SYS (Optimization)
                                %if 0

                                ;--------------------------------------------------------------------------
                                ;
                                ; procedure : getchr
                                ;
                                ;--------------------------------------------------------------------------

                                    ; 24/10/2022
                                getchr:
                                    ; 12/12/2022
                                    ;push   cx
                                    ;mov    cx,[count]
                                    ;jcxz   nochar
                                    ; 12/12/2022
                                    cmp     word [count],1
                                    jb      short nochar ; cf=1 ([count] = 0)

                                    mov     si,[chrptr]
                                    mov     al,[es:si]
                                    dec     word [count]
                                    inc     word [chrptr]
                                    ; 12/12/202
                                    ; cf=0
                                    ;clc
                                ;get_ret:
                                    ;pop    cx
                                    ;retn
                                nochar:
                                    ; 12/12/2022
                                    ; cf=1
                                    ;stc
                                    ;jmp    short get_ret

                                    retn
                                %endif

                                ; 11/12/2022
                                %if 0

                                ;--------------------------------------------------------------------------
                                ;
                                ; procedure : incorrect_order
                                ;
                                ;             show "incorrect order in config.sys ..." message.
                                ;
                                ;--------------------------------------------------------------------------

                                incorrect_order:
                                    mov     dx,badorder
                                    call    print
                                    call    showlinenum
                                    retn

                                %endif

                                ;--------------------------------------------------------------------------
                                ;
                                ; procedure : error_line
                                ;
                                ;             show "error in config.sys ..." message.
                                ;
                                ;--------------------------------------------------------------------------

                                    ; 11/12/2022
                                    ; 24/10/2022
                                error_line:
                                    ; 11/12/2022
                                    ; ds = cs
                                    ;push   cs
                                    ;pop    ds

 00002EBA BA[A851]                  mov     dx,errorcmd
 00002EBD E8941B                    call    print
                                    ;call   showlinenum
                                    ;retn
                                    ; 11/12/2022
                                    ;jmp    short shortlinemum

                                ;--------------------------------------------------------------------------
                                ;
                                ; procedure : showlinenum
                                ;
                                ; convert the binary linecount to decimal ascii string in showcount
                                ; and display showcount at the current curser position.
                                ; in.) linecount
                                ;
                                ; out) the number is printed.
                                ;
                                ;--------------------------------------------------------------------------

                                    ; 11/12/2022
                                    ; ds = cs
                                    ; 24/10/2022
                                showlinenum:
 00002EC0 06                        push    es
                                    ; 11/12/2022
                                    ;push   ds
 00002EC1 57                        push    di

 00002EC2 0E                        push    cs
 00002EC3 07                        pop     es              ; es=cs

                                    ; 11/12/2022
                                    ;push   cs
                                    ;pop    ds

 00002EC4 BF[B502]                  mov     di,showcount+4 ; di -> the least significant decimal field.
 00002EC7 B90A00                    mov     cx,10          ; decimal divide factor
                                    ;mov    ax,[cs:linecount]
                                    ; 11/12/2022
 00002ECA A1[AF02]                  mov     ax,[linecount]
                                sln_loop:
                                    ; 11/12/2022
 00002ECD 39C8                      cmp     ax,cx ; < 10 ?
                                    ;cmp    ax,10          ; < 10?
```

```
35560 00002ECF 720C                     jb      short sln_last
35561
35562 00002ED1 31D2                     xor     dx,dx
35563 00002ED3 F7F1                     div     cx      ; cx = 10
35564 00002ED5 80CA30                   or      dl,30h          ; add "0" (= 30h) to make it an ascii.
35565 00002ED8 8815                     mov     [di],dl
35566 00002EDA 4F                       dec     di
35567 00002EDB EBF0                     jmp     short sln_loop
35568
35569                                sln_last:
35570 00002EDD 0C30                     or      al,30h ; "0"
35571 00002EDF 8805                     mov     [di],al
35572 00002EE1 89FA                     mov     dx,di
35573 00002EE3 E86E1B                   call    print           ; show it.
35574 00002EE6 5F                       pop     di
35575                                   ; 11/12/2022
35576                                   ;pop    ds
35577 00002EE7 07                       pop     es
35578 00002EE8 C3                       retn
35579
35580                                ; 07/04/2019 - Retro DOS v4.0
35581                                ; (MSDOS 6.21 IO.SYS, SYSINIT:2E44h)
35582
35583                                ;--------------------------------------------------------------------------
35584                                ;
35585                                ; procedure : ProcDOS
35586                                ;
35587                                ;   Process the result of DOS= parsing
35588                                ;
35589                                ;   result_val._$P_item_tag       = 1 for DOS=HIGH
35590                                ;                                 = 2 for DOS=LOW
35591                                ;                                 = 3 for DOS=UMB
35592                                ;                                 = 4 for DOS=NOUMB
35593                                ;--------------------------------------------------------------------------
35594
35595                                    ; 01/11/2022 - Retro DOS v4.0 (Modififed MSDOS 5.0 IO.SYS)
35596                                    ; (SYTSINIT:2AB5h)
35597                                ProcDOS:
35598                                    ; 01/01/2023
35599                                    ; ds = cs
35600 00002EE9 30E4                     xor     ah,ah
35601                                   ;;mov   al,[cs:result_val_itag]
35602                                   ;mov    al,[cs:result_val+_$P_Result_Blk.Item_Tag]
35603                                   ; 01/01/2023
35604 00002EEB A0[1822]                 mov     al,[result_val+_$P_Result_Blk.Item_Tag]
35605 00002EEE 48                       dec     ax
35606 00002EEF 7415                     jz      short pd_hi
35607 00002EF1 48                       dec     ax
35608 00002EF2 740E                     jz      short pd_lo
35609 00002EF4 48                       dec     ax
35610 00002EF5 7405                     jz      short pd_umb
35611                                   ;;mov   byte [cs:DevUMB],0
35612                                   ; 18/12/2022
35613                                   ;mov    byte [cs:DevUMB],ah ; 0
35614                                   ; 01/01/2023
35615 00002EF7 8826[4224]               mov     byte [DevUMB],ah ; 0
35616 00002EFB C3                       retn
35617                                pd_umb:
35618                                   ; 01/01/2023
35619 00002EFC C606[4224]FF             mov     byte [DevUMB],0FFh
35620                                   ;mov    byte [cs:DevUMB],0FFh
35621 00002F01 C3                       retn
35622                                pd_lo:
35623                                   ; 01/01/2023
35624 00002F02 A2[6C02]                 mov     [runhigh],al ; 0
35625                                   ; 18/12/2022
35626                                   ;mov    [cs:runhigh],al ; 0
35627                                   ;;mov   byte [cs:runhigh],0
35628 00002F05 C3                       retn
35629                                pd_hi:
35630                                   ; 01/01/2023
35631 00002F06 C606[6C02]FF             mov     byte [runhigh],0FFh
35632                                   ;mov    byte [cs:runhigh],0FFh
35633                                limx:       ; 11/12/2022
35634 00002F0B C3                       retn
35635
35636                                ;--------------------------------------------------------------------------
35637                                ;
35638                                ; procedure : LieInt12Mem
35639                                ;
35640                                ;   Input : DevEntry points to Device Start address (offset == 0)
35641                                ;           alloclim set to the limit of low memory.
35642                                ;
35643                                ;   Output : none
35644                                ;
35645                                ;   Changes the ROM BIOS variable which stores the total low memory
35646                                ;   If a 3com device driver (any character device with name 'PROTMAN$')
35647                                ;   is being loaded alloclim is converted into Ks and stored in 40:13h
35648                                ;   Else if a device driver being loaded into UMB the DevLoadEnd is
35649                                ;   converted into Ks and stored in 40:13h
35650                                ;
35651                                ;--------------------------------------------------------------------------
35652
35653                                LieInt12Mem:
35654                                    ; 11/12/2022
35655                                    ; ds = cs
35656 00002F0C A1[A502]                 mov     ax,[ALLOCLIM]
35657                                   ;mov    ax,[cs:ALLOCLIM]      ; lie INT 12 as alloclim
35658                                                                 ; assuming that it is 3Com
35659 00002F0F E84200                   call    IsIt3Com              ; Is it 3Com driver?
35660 00002F12 740A                     jz      short lim_set         ; yes, lie to him differently
35661                                   ; 13/05/2019
35662                                   ;cmp    byte [cs:DeviceHi],0  ; Is the DD being loaded in UMB
35663                                   ;je     short limx            ; no, don't lie
35664                                   ;mov    ax,[cs:DevLoadEnd]    ; lie INT 12 as end of UMB
35665                                   ; 11/12/2022
35666                                   ; ds = cs
35667 00002F14 803E[5124]00             cmp     byte [DeviceHi],0
35668 00002F19 74F0                     je      short limx
35669 00002F1B A1[3724]                 mov     ax,[DevLoadEnd]
35670                                lim_set:
35671                                   ; 01/11/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
35672                                   ; 11/12/2022
35673                                   ;call   SetInt12Mem
35674                                ;limx:
35675                                   ;retn
35676
35677                                   ;jmp    short SetInt12Mem
35678
35679                                ;--------------------------------------------------------------------------
35680                                ;
35681                                ; procedure : SetInt12Mem
35682                                ;
35683                                ;   Input : AX = Memory size to be set (in paras)
```

```
                                ;   Output : none
                                ;
                                ;   Sets the variable 40:13 to the memory size passed in AX
                                ;   It saves the old value in 40:13 in OldInt12Mem,
                                ;   It also sets a flag Int12Lied to 0ffh, which is checked before
                                ;   restoring the value of 40:13
                                ;
                                ;--------------------------------------------------------------------------

                                    ; 01/11/2022
                                SetInt12Mem:
00002F1E 1E                         push    ds
00002F1F BB4000                     mov     bx,40h
00002F22 8EDB                       mov     ds,bx                   ; ROM BIOS Data Segment
00002F24 8B1E1300                   mov     bx,[13h]                ; INT 12 memory variable
                                    ;mov    [cs:OldInt12Mem],bx     ; save it
00002F28 B106                       mov     cl,6
00002F2A D3E8                       shr     ax,cl                   ; convert paras into Ks
00002F2C A31300                     mov     [13h],ax                ; Lie
                                    ;mov    byte [cs:Int12Lied],0FFh ; mark that we are lying
00002F2F 1F                         pop     ds
                                    ; 14/04/2024
                                    ; ds = cs
00002F30 891E[5524]                 mov     [OldInt12Mem],bx
00002F34 C606[5424]FF               mov     byte [Int12Lied],0FFh
                                ;limx:
00002F39 C3                         retn

                                ;--------------------------------------------------------------------------
                                ;
                                ; procedure : TrueInt12Mem
                                ;
                                ;   Input : Int12Lied = 0 if we are not lying currently
                                ;                     = 0ffh if we are lying
                                ;           OldInt12Mem = Saved value of 40:13h
                                ;
                                ;   Output : none
                                ;
                                ;   Resets the INT 12 Memory variable if we were lying about int 12
                                ;   and resets the flag which indicates that we were lying
                                ;
                                ;--------------------------------------------------------------------------

                                TrueInt12Mem:
                                    ; 11/12/2022
                                    ; ds = cs
00002F3A 803E[5424]00               cmp     byte [Int12Lied],0
                                    ;cmp    byte [cs:Int12Lied],0 ; were we lying so far?
                                    ; 01/11/2022 (MSDOS 5.0 IO.SYS, SYS.INIT:2B1Dh)
                                    ;mov    byte [cs:Int12Lied],0 ; reset it anyway
00002F3F 7412                       je      short timx              ; no, we weren't
                                    ; 18/12/2022
00002F41 B84000                     mov     ax,40h
00002F44 8826[5424]                 mov     [Int12Lied],ah ; 0
                                    ;mov    byte [Int12Lied],0
                                    ;mov    byte [cs:Int12Lied],0
00002F48 1E                         push    ds
                                    ;mov    ax,40h
00002F49 8ED8                       mov     ds,ax
00002F4B 2EA1[5524]                 mov     ax,[cs:OldInt12Mem]
00002F4F A31300                     mov     [13h],ax                ; restore INT 12 memory
00002F52 1F                         pop     ds
                                timx:
00002F53 C3                         retn

                                ;--------------------------------------------------------------------------
                                ;
                                ; procedure : IsIt3Com?
                                ;
                                ;   Input : DevEntry = Seg:0 of device driver
                                ;   Output : Zero flag set if device name is 'PROTMAN$'
                                ;            else Zero flag is reset
                                ;
                                ;--------------------------------------------------------------------------

                                IsIt3Com:
                                    ; 11/12/2022
                                    ; ds = cs
00002F54 1E                         push    ds
00002F55 06                         push    es
00002F56 56                         push    si
                                    ; 11/12/2022
00002F57 C536[3924]                 lds     si,[DevEntry]
                                    ;lds    si,[cs:DevEntry]        ; ptr to device header
00002F5B 83C60A                     add     si,SYSDEV.NAME ; 10     ; ptr device name
00002F5E 0E                         push    cs
00002F5F 07                         pop     es
00002F60 BF[5724]                   mov     di,ThreeComName
00002F63 B90800                     mov     cx,8                    ; name length
00002F66 F3A6                       rep     cmpsb
00002F68 5E                         pop     si
00002F69 07                         pop     es
00002F6A 1F                         pop     ds
00002F6B C3                         retn

                                ;M020 : BEGIN
                                ;--------------------------------------------------------------------------

                                UpdatePDB:
00002F6C 1E                         push    ds
00002F6D B462                       mov     ah,62h
00002F6F CD21                       int     21h     ; DOS - 3+ - GET PSP ADDRESS
00002F71 8EDB                       mov     ds,bx
00002F73 2E8B1E[A502]               mov     bx,[cs:ALLOCLIM]
                                    ;mov    [2],bx
00002F78 891E0200                   mov     [PDB.BLOCK_LEN],bx
00002F7C 1F                         pop     ds
00002F7D C3                         retn

                                ; M020 : END

                                ;--------------------------------------------------------------------------

                                ; 01/11/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
                                ;%if 0

                                ; 31/12/2022 - Retro DOS v4.2 (Modified MSDOS 6.21 IO.SYS)
                                ; (SYSINIT:2EEEh)

                                ;include highload.inc      ; Routines for devicehigh parsing, control of HIDDEN
                                ;include highexit.inc      ; umb's, etc

                                ; ----------------------------------------------------------------------
                                ; HIGHLOAD.INC (MSDOS 6.0 - 1991)
```

```
                        ; ----------------------------------------------------------------------
                        ; 07/04/2019 - Retro DOS v4.0

                        ;******************************************************************************
                        ;
                        ; This file contains routines needed to parse and implement user-given
                        ; command-line options of the form "/S/L:3,0x500;2;7,127;0x0BE4". InitVar()
                        ; and Parsevar() are used to parse this data and place it in encoded form into
                        ; the variables in highvar.inc, for use by the rest of the routines.
                        ;
                        ; DeviceHigh accepts this command-line (handled in sysconf.asm, not here):
                        ;    DEVICEHIGH SIZE=hhhhhh module opts
                        ; Or, DeviceHigh and LoadHigh accept any of the following:
                        ;    DH/LH module opts
                        ;    DH/LH [/S][/L:umb[,size][;umb[,size]]*] module opts
                        ;    DH/LH [/L:umb[,size][;umb[,size]]*][/S] module opts
                        ; The initial UMB,SIZE pair designates the module's load address; the remainder
                        ; of the UMB and SIZE pairs are used to indicate specific UMBs to be left
                        ; available during the load.
                        ;
                        ; When an actual load is ready to be performed, a call to HideUMBs() will
                        ; temporarily allocate (as owner 8+"HIDDEN  ") all free elements in any
                        ; upper-memory block which was not specified by the user... in addition, if
                        ; UMBs were marked to shrink (/S option) to a certain size ("umb,size"), any
                        ; elements in that umb SAVE the lower-half of the newly-shrunken one are also
                        ; allocated. After the load, the function UnHideUMBs() (in highexit.inc) will
                        ; free any UMBs so allocated.
                        ;
                        ; When a device driver loads, there is the additional problem of allocating its
                        ; initial load site; this should be restricted to the first UMB specified on
                        ; the command-line. The function FreezeUM temporarily allocates all remaining
                        ; free upper-memory elements (as owner 8+"FROZEN  "), except those in the load
                        ; UMB. Then the initial allocation may be made, and a call to UnFreeze will
                        ; return any so-allocated memory elements to FREE, for the true load. Note
                        ; that UnFreeze leaves HIDDEN elements allocated; it only frees FROZEN ones.
                        ;
                        ;******************************************************************************

                        SWTCH       equ     '/'             ; Switch character

                        DOS_CHECK_STRATEGY  equ    5800h    ; Int 21h, Func 58h, Svc 0 = check alloc strat
                        DOS_SET_STRATEGY    equ    5801h    ; Int 21h, Func 58h, Svc 1 = set alloc strategy
                        DOS_CHECK_UMBLINK   equ    5802h    ; Int 21h, Func 58h, Svc 2 = check link state
                        DOS_GET_UMBLINK        equ 5802h ; 20/04/2019
                        DOS_SET_UMBLINK     equ    5803h    ; Int 21h, Func 58h, Svc 3 = set link state
                        DOS_GET_DOS_LISTS   equ      52h    ; Int 21h, Func 52h = return list of lists
                        DOS_UMB_HEAD        equ      8Ch    ; Offset from ES (after func52h) to get UMBHead

                        CR  equ     0Dh             ; Carriage Return
                        LF  equ     0Ah             ; Line Feed
                        TAB equ     09h             ; Tab character (^I)

                        ; ----------------------------------------------------------------------------
                        ;*** InitVar - initializes all the variables used in ParseVar and HideUMBs
                        ; ----------------------------------------------------------------------------
                        ; ENTRY:       None
                        ; EXIT:        Variables listed in highvar.inc are initialized
                        ; ERROR EXIT:  None
                        ; USES:        Flags, variables in highvar.inc
                        ; ----------------------------------------------------------------------------
                        ; Note that element 0 references UMB 0 (conventional), not UMB 1. Its contents
                        ; are largely ignored, but it is initialized nonetheless.
                        ; ----------------------------------------------------------------------------

                        ; 01/01/2023 - Retro DOS v4.2 (Modified MSDOS 6.21 IO.SYS)
                        ; (SYSINIT:2EEEh)

                        InitVar:
                            ; 01/01/2023
                            ; ds = cs

                            ;pushreg <ax, cx, di, es>
                            ; 03/01/2023
                            ;push   ax
                            ;push   cx
                            ;push   di
00002F7E 06                 push    es

                            ;dataseg es                 ;Point ES into appropriate data segment
00002F7F 0E                 push    cs
00002F80 07                 pop     es

00002F81 31C0               xor     ax,ax
                            ;mov    [es:fUmbTiny],al    ;Shrink UMBs? (made 1 if /S given)
                            ;mov    [es:fInHigh],al             ;Set to 1 when DH/LH has been called
                            ;mov    [es:SegLoad],ax             ;Load Address (seg), used for DH only
                            ;mov    byte [es:UmbLoad],UNSPECIFIED ; 0FFh
                            ;                           ;Later is the # of the 1st spec'd UMB
                            ;mov    [es:fm_argc], al    ;Start with zero args having been read

                            ; 01/01/2023
                            ; ds = cs
00002F83 A2[FC23]           mov     [fUmbTiny],al       ;Shrink UMBs? (made 1 if /S given)
00002F86 A2[FB23]           mov     [fInHigh],al        ;Set to 1 when DH/LH has been called
00002F89 A3[FD23]           mov     [SegLoad],ax        ;Load Address (seg), used for DH only
00002F8C C606[FF23]FF       mov     byte [UmbLoad],UNSPECIFIED ; 0FFh
                                                        ;Later is the # of the 1st spec'd UMB
00002F91 A2[3224]           mov     [fm_argc], al       ;Start with zero args having been read

00002F94 FC                 cld

00002F95 B91000             mov     cx,MAXUMB ; 16      ;For each entry
00002F98 BF[0024]           mov     di,UmbUsed          ;on the UmbUsed array,
00002F9B F3AA               rep     stosb               ;       Store 0

                            ;mov    cx,MAXUMB ; 16      ;Okay... for each entry
                            ; 01/01/2033
00002F9D B110               mov     cl,MAXUMB ; 16
00002F9F BF[1024]           mov     di,UmbSize          ;on the UmbSize array,
00002FA2 F3AB               rep     stosw               ;       Store 0

                            ;normseg es                 ; Return ES

                            ;popreg <es, di, cx, ax>
00002FA4 07                 pop     es
                            ; 03/01/2023
                            ;pop    di
                            ;pop    cx
                            ;pop    ax

00002FA5 C3                 retn

                        ; ----------------------------------------------------------------------------
                        ;*** FixMem - scans the upper memory chain and concatenates adjacent free MCBs
```

```
                            ; ----------------------------------------------------------------------------
                            ; ENTRY   : None
                            ; EXIT    : None
                            ; ERROR   : None
                            ; USES    : Flags, fm_umb, fm_strat
                            ; ----------------------------------------------------------------------------

                                ; 01/01/2023 - Retro DOS v4.2 (Modified MSDOS 6.21 IO.SYS)
                                ; (SYSINIT:2F22h)
                            FixMem:
                                ; 01/01/2023
                                ;push  ax
                                ;push  bx
                                ;push  cx
                                ;push  dx
00002FA6 06                     push   es

00002FA7 E84900                 call   fm_link        ; Link in UMBs

00002FAA E80002                 call   UmbHead        ; Get first upper-memory MCB address (0x9FFF)
00002FAD 723F                   jc     short fmX      ; (if couldn't get it, leave now).

00002FAF 8EC0                   mov    es,ax          ; It returns in AX, so move it to ES.

                            ; - Walk MCB Chain -------------------------------------------------------------

00002FB1 31D2                   xor    dx,dx          ; We're keeping the address of the last MCB
00002FB3 89D1                   mov    cx,dx          ; in CX... and the last owner
00002FB5 42                     inc    dx             ; in dx as we go through the loop:

                            ; ----------------------------------------
                            ; FM10--DX  = last MCB's owner's PSP address
                            ;       CX  = last MCB's address (segment)
                            ; ----------------------------------------

00002FB6 26A00000           fm10:       mov    al,[es:ARENA.SIGNATURE] ; if 'Z', don't repeat loop
00002FBA 268B1E0100             mov    bx,[es:ARENA.OWNER]    ; if not zero, do nothing
00002FBF 09D3                   or     bx,dx                  ; dx was owner of previous MCB
00002FC1 7516                   jnz    short fm30             ; If not both zero, don't cat.

                                ; - Coalesce memory blocks at ES:00 and CX:00 -------------------------

00002FC3 268B1E0300         fm20:       mov    bx,[es:ARENA.SIZE]    ; Grab this block's Size,
00002FC8 8EC1                   mov    es,cx                  ; Go back to prev MCB's address
00002FCA 26A20000               mov    [es:ARENA.SIGNATURE],al ; & move the SECOND sig here

00002FCE 26031E0300             add    bx,[es:ARENA.SIZE]     ; Size += first MCB's size
                                ;add   bx,1                   ; And add one for the header
                                ; 11/07/2023
00002FD3 43                     inc    bx
00002FD4 26891E0300             mov    [es:ARENA.SIZE],bx     ; Write the size

                                ; ----------------------------------------------------------------------

00002FD9 8CC1               fm30:       mov    cx,es                  ; Put this address on the stack
00002FDB 268B160100             mov    dx,[es:ARENA.OWNER]    ; And remember its owner

00002FE0 8CC3                   mov    bx,es                  ; Move to the next MCB
00002FE2 26031E0300             add    bx,[es:ARENA.SIZE]
00002FE7 43                     inc    bx
00002FE8 8EC3                   mov    es,bx

                                ;cmp   al,'Z'
00002FEA 3C5A                   cmp    al,arena_signature_end
00002FEC 75C8                   jne    short fm10             ; If signature != 'Z', there are more.
                            fmX:
00002FEE E81300                 call   fm_unlink              ; Unlink UMBs

00002FF1 07                     pop    es
                                ; 01/01/2023
                                ;pop   dx
                                ;pop   cx
                                ;pop   bx
                                ;pop   ax

00002FF2 C3                     retn

                            ; ----------------------------------------------------------------------------
                            ;*** fm_link - links UMBs not already linked in
                            ; ----------------------------------------------------------------------------
                            ; ENTRY:    None
                            ; EXIT:     fm_umb == 0 if not linked in previously, 1 if already linked in
                            ; ERROR:    None
                            ; USES:     AX, BX, fm_umb
                            ; ----------------------------------------------------------------------------

                                ; 01/01/2023 - Retro DOS v4.2
                            fm_link:
00002FF3 B80258                 mov    ax,DOS_CHECK_UMBLINK ; 5802h
00002FF6 CD21                   int    21h                  ; Current link-state is now in al

                                ;putdata fm_umb,al           ; So store it in fm_umb for later
                                ;
                                ;push  es
                                ;push  cs
                                ;pop   es
                                ;mov   [es:fm_umb],al
                                ;pop   es

                                ; 01/01/2023
                                ; ds = cs
                                ;mov   [cs:fm_umb],al
00002FF8 A2[3024]               mov    [fm_umb],al

00002FFB B80358                 mov    ax,DOS_SET_UMBLINK ; 5803h
00002FFE BB0100                 mov    bx,1
00003001 CD21                   int    21h
00003003 C3                     retn

                            ; ----------------------------------------------------------------------------
                            ;*** fm_unlink - unlinks UMBs if fm_umb is set to 0
                            ; ----------------------------------------------------------------------------
                            ; ENTRY:    fm_umb == 1 : leave linked, else unlink
                            ; EXIT:     None
                            ; ERROR:    None
                            ; USES:     AX, BX
                            ; ----------------------------------------------------------------------------

                                ; 01/01/2023 - Retro DOS v4.2
                            fm_unlink:
00003004 31DB                   xor    bx,bx

                                ;getdata bl,fm_umb           ; fm_umb already has the old link-state
                                ;
```

```
                                    ;push   ds
                                    ;push   cs
                                    ;pop    ds
                                    ;mov    bl,[fm_umb]
                                    ;pop    ds

                                    ; 01/01/2023
                                    ; ds = cs
                                    ;mov    bl,[cs:fm_umb]
00003006 8A1E[3024]                 mov     bl,[fm_umb]

0000300A B80358                     mov     ax,DOS_SET_UMBLINK ; 5803h
0000300D CD21                       int     21h                     ; so just use that, and call int 21h
0000300F C3                         retn

                                ; 08/04/2019 - Retro DOS v4.0

                                ; ----------------------------------------------------------------------------
                                ;*** ParseVar - parses [/S][/L:umb[,size][;umb[,size]]*] and builds the table
                                ; laid out in highvar.inc
                                ; ----------------------------------------------------------------------------
                                ; ENTRY:    ES:SI points to command tail of LoadHigh/DeviceHigh (whitespace ok)
                                ; EXIT:     ES:SI points to first character in child program name
                                ; ERROR:    ES:SI points to character which caused error, carry set, AX == code
                                ; USES:     ES:SI, AX, flags, variables in highvar.inc
                                ; ----------------------------------------------------------------------------
                                ; Error codes (in AX if carry set on return):
                                ;
                                PV_InvArg   equ     1       ; Invalid argument passed
                                PV_BadUMB   equ     2       ; Bad UMB number passed (duplicate?)
                                PV_InvSwt   equ     3       ; Unrecognized switch passed
                                ;
                                ; This routine exects ES:SI to point to a string much like the following:
                                ;    "/S/L:1,200;2 module options"
                                ; Optionally, the string can begin with whitespace; neither /S nor /L is
                                ; required, though that's what this routine is supposed to parse.
                                ;
                                optS        equ     'S'     ; /S
                                optL        equ     'L'     ; /L:...
                                ;
                                ; ----------------------------------------------------------------------------
                                ; LoadHigh has a list of arguments, returned by cparse, which is used to create
                                ; a command-line for spawning a child process. For a typical LH command, say,
                                ;     lh /l:1,1000;2 print/d:lpt2
                                ; the arguments would look like (one per line):
                                ;     lh
                                ;     /l
                                ;     1
                                ;     1000
                                ;     2
                                ;     print
                                ;     /d
                                ;     :lpt2
                                ; In short, if "print" were, say, "43", there'd be no way to determine which
                                ; arg was the filename. So, inside this routine, we keep a running counter
                                ; of the number of arguments LH will need to skip in order to get to the
                                ; program name. The "lh" is implicit--it'll always have to skip that. So if
                                ; there's no "/l" or "/s", fm_argc will be 0 ... other than that, 1 is added
                                ; for:
                                ;    Each /L
                                ;    Each /S (there should be only one)
                                ;    Each UMB number (they follow ":" or ";")
                                ;    Each UMB size   (they follow ",")
                                ; So, in the above example, fm_argc would be 4-- and LH would skip right to
                                ; "print".  Note that InitVar initializes fm_argc to zero.
                                ; ----------------------------------------------------------------------------

                                    ; 01/01/2023 - Retro DOS v4.2 (Modified MSDOS 6.21 IO.SYS)
                                    ; (SYSINIT:2F9Fh)

                                ParseVar:
                                    ;pushreg <di, ds, es>
                                    ; 01/01/2023
                                    ;push   di ; * ; (not required) ; 01/01/2023
00003010 1E                         push    ds
00003011 06                         push    es

00003012 06                         push    es              ; Make DS:SI point to it, as well as ES:SI
00003013 1F                         pop     ds              ; (regardless if we're in devhigh or loadhigh)
00003014 FC                         cld

                                ; ------------------------------------------------
                                ; PV10--ES:SI = any whitespace on the command-line
                                ; ------------------------------------------------

00003015 AC                     pv10:       lodsb                   ; here, ES:SI=="  /L..."--must eat whitespace
00003016 E8A200                     call    isWhite
00003019 74FA                       jz      short pv10      ;       ES:SI==" /L..."--keep eating.
                                    ;cmp    al,'/'
0000301B 3C2F                       cmp     al,SWTCH
0000301D 7404                       je      short pv20      ;       ES:SI=="/L..."--go process a switch

0000301F 4E                         dec     si              ; Backup--it's now "odule options", and we need
00003020 F8                         clc                     ; that "m" we just read (or whatever it is).
00003021 EB2B                       jmp     short pvX       ; Then return with carry clear == we're done.

00003023 AC                     pv20:       lodsb                   ; Just read 'S' or 'L', hopefully
                                    ;toUpper al             ; So we make it upper-case, and...
00003024 24DF                       and     al,0DFh
                                    ;cmp    al,'S'
00003026 3C53                       cmp     al,optS         ; just read 'S'?
00003028 750D                       jne     short pv30

                                    ;call   incArgc         ; If it's /S, it's another arg for LH to skip.
0000302A 2EFE06[3224]               inc     byte [cs:fm_argc] ; 19/04/2019

                                    ;putdata fUmbTiny,1     ; /S, so ES:SI=="  /L..." or " module opts", or
                                    ;
                                    ;push   es
                                    ;push   cs
                                    ;pop    es
                                    ;mov    [es:fUmbTiny],1
                                    ;pop    es

0000302F 2EC606[FC23]01             mov     byte [cs:fUmbTiny],1

00003035 EBDE                       jmp     short pv10      ; possibly even "/L...".

                                pv30:       ;cmp    al,'L'
00003037 3C4C                       cmp     al,optL         ; If it's not 'L' either, then 'tis a bad
00003039 750D                       jne     short pvE1              ; switch!

                                    ;call   incArgc         ; If it's /L, it's another arg for LH to skip.
0000303B 2EFE06[3224]               inc     byte [cs:fm_argc] ; 19/04/2019
```

```
36180
36181 00003040 E80E00                  call    parseL
36182 00003043 73D0                    jnc     short pv10      ; If no carry, go back and look for more
36183
36184 00003045 4E                      dec     si              ; Else, back up and exit.
36185 00003046 EB03                    jmp     short pvErr     ; AX has already been set by parseL
36186
36187                                  pvE1:       ;mov    ax,3
36188 00003048 B80300                  mov     ax,PV_InvSwt    ; Unrecognized switch passed
36189 0000304B 4E                      pvErr:      dec     si
36190 0000304C 4E                      dec     si
36191 0000304D F9                      stc
36192                                  pvX:;popreg <es, ds, di>
36193 0000304E 07                      pop     es
36194 0000304F 1F                      pop     ds
36195                                  ; 01/01/2023
36196                                  ;pop    di ; * ; (not required) ; 01/01/2023
36197 00003050 C3                      retn
36198
36199                                  ; ---------------------------------------------------------------------------
36200                                  ;*** parseL - parses ":nnnn[,nnnn][;nnnn[,nnnn]]*" for ParseVar
36201                                  ; ---------------------------------------------------------------------------
36202                                  ; ENTRY:    ES:SI points to colon
36203                                  ; EXIT:     ES:SI points to first character not parsed
36204                                  ; ERROR:    Carry set; rewind three characters and return (see ParseVar)
36205                                  ; USES:     ES:SI, flags, AX, CX, DX, variables in highvar.inc
36206                                  ; ---------------------------------------------------------------------------
36207                                  ; If the string here is terminated with anything other than whitespace or a
36208                                  ; switchchar (perhaps it's /S or another /L:... ), then we return with carry
36209                                  ; set, indicating that they've screwed up the syntax.  The 3-character rewind
36210                                  ; makes sure the app /L: is reported as being the culprit.
36211                                  ; ---------------------------------------------------------------------------
36212
36213                                  parseL:
36214 00003051 AC                      lodsb
36215 00003052 3C3A                    cmp     al,':'          ; Make sure they did /L:
36216 00003054 754E                    jne     short plE1      ; If they didn't, return with carry set.
36217
36218                                  ; ----------------------------------------
36219                                  ; PL10--ES:SI = a UMB number, after /L: or ;
36220                                  ; ----------------------------------------
36221
36222 00003056 E8DB00                  pl10:       call    GetXNum         ; After this, 'tis ",size" or ";umb" or " mod"
36223 00003059 724F                        jc      short plE2      ; And error if it's a bad number.
36224 0000305B E89D01                      call    convUMB         ; Convert any address to a UMB number
36225
36226 0000305E 88C1                        mov     cl,al           ; Remember the UMB number
36227 00003060 E87600                      call    stowUMB         ; Mark this UMB # as used;
36228 00003063 7245                        jc      short plE2      ; If it was already marked, it'll error
36229
36230                                      ;call   incArgc         ; Each UMB number is another arg for LH to skip
36231 00003065 2EFE06[3224]                inc     byte [cs:fm_argc] ; 08/04/2019 - Retro DOS v4.0
36232
36233 0000306A AC                          lodsb
36234 0000306B 3C3B                        cmp     al,';'          ; Did "umb;" ?
36235 0000306D 74E7                        je      short pl10      ; Yep: go back and get another UMB.
36236
36237 0000306F E84900                      call    isWhite         ; Did "umb " ?
36238 00003072 743B                        jz      short plX       ; Yep: return (it'll go back to whitespace)
36239
36240 00003074 E83900                      call    isEOL           ; Did "umb" ?
36241 00003077 7435                        jz      short plSwX     ; If so, backup and exit like everything's ok
36242
36243                                      ;cmp    al,'/'
36244 00003079 3C2F                        cmp     al,SWTCH        ; Did "umb/" ? (as in, "/L:1,100;2/S")
36245 0000307B 7431                        je      short plSwX     ; If so, back up ES:SI one character and return
36246
36247 0000307D 3C2C                        cmp     al,','          ; Did "umb," ?
36248 0000307F 7523                        jne     short plE1      ; Just what the heck DID they do? Return error.
36249
36250                                  ; --- Read a size -----------------------------------------------------------
36251
36252 00003081 E8B000                      call    GetXNum         ; Stop on "size;" or "size " or anything else
36253 00003084 721E                        jc      short plE1      ; And error if it's a bad size.
36254
36255 00003086 E81601                      call    toPara          ; Convert from bytes to paragraphs
36256
36257 00003089 E8CE01                      call    stowSiz         ; CL still has the UMB number for this routine
36258
36259                                      ;call   incArgc         ; Each UMB size is another arg for LH to skip
36260 0000308C 2EFE06[3224]                inc     byte [cs:fm_argc] ; 08/04/2019 - Retro DOS v4.0
36261
36262 00003091 AC                          lodsb
36263 00003092 3C3B                        cmp     al,';'          ; They did "umb,size;", so get another UMB.
36264 00003094 74C0                        je      short pl10      ;
36265
36266 00003096 E82200                      call    isWhite         ; Did it end with whitespace?
36267 00003099 7414                        jz      short plX       ; If so, we're done here--go back.
36268
36269 0000309B E81200                      call    isEOL           ; Did they do "umb,size" and end??? (stupid)
36270 0000309E 740E                        jz      short plSwX     ; If so, backup and exit like everything's ok
36271
36272                                      ;cmp    al,'/'
36273 000030A0 3C2F                        cmp     al,SWTCH        ; Did they do "umb,size/" ?
36274 000030A2 740A                        je      short plSwX     ; If so, again, we're done here.
36275                                  plE1:
36276                                      ;mov    ax,1
36277 000030A4 B80100                      mov     ax,PV_InvArg    ; If not, we don't know WHAT they did...
36278 000030A7 4E                          dec     si
36279 000030A8 F9                          stc
36280 000030A9 C3                          retn
36281
36282                                  plE2:       ;mov    ax,2
36283 000030AA B80200                      mov     ax,PV_BadUMB    ; In this case, they've specified a UMB twice
36284                                      ; 12/12/2022
36285                                      ; cf=1
36286                                      ;stc
36287 000030AD C3                          retn
36288                                  plSwX:
36289 000030AE 4E                          dec     si              ; If we hit a '/' character, back up one char
36290                                                              ; so the whitespace checker will see it too.
36291                                  plX:; 12/12/2022
36292                                      ; cf=0
36293                                      ;clc                    ; Then just return with carry clear, so
36294 000030AF C3                          retn                    ; ParseVar will go about its business.
36295
36296                                  ; ---------------------------------------------------------------------------
36297                                  ;*** incArgc - increments fm_argc, for use with LoadHigh command-line parsing
36298                                  ; ---------------------------------------------------------------------------
36299                                  ; ENTRY:    None
36300                                  ; EXIT:     None
36301                                  ; ERROR:    None
36302                                  ; USES:     fm_argc, flags
36303                                  ; ---------------------------------------------------------------------------
```

```
                                ;incArgc:
                                    ;push   ax
                                    ;;getdata al, fm_argc ; Obtain previous value of fm_argc,
                                    ;mov    al,[cs:fm_argc]
                                    ;inc    al              ; Increment it,
                                    ;;putdata fm_argc, al ; And store it right back.
                                    ;mov    [cs:fm_argc],al
                                    ;pop    ax
                                    ;retn
                                ; ------------------------------------------------------------------------------
                                ;*** isEOL - returns with ZF set if AL contains CR or LF, or 0
                                ; ------------------------------------------------------------------------------
                                ; ENTRY:    AL contains character to test
                                ; EXIT:     ZF set iff AL contains CR or LF, or 0
                                ; ERROR:    None
                                ; USES:     ZF
                                ; ------------------------------------------------------------------------------
                                isEOL:
000030B0 3C00                       cmp     al,0            ; Null-terminator
000030B2 7406                       je      short ieX
000030B4 3C0D                       cmp     al,CR ; 0Dh     ; Carriage Return
000030B6 7402                       je      short ieX
000030B8 3C0A                       cmp     al,LF ; 0Ah     ; LineFeed
                                ieX:
000030BA C3                         retn
                                ; ------------------------------------------------------------------------------
                                ;*** isWhite - returns with ZF set if AL contains whitespace (or "=")
                                ; ------------------------------------------------------------------------------
                                ; ENTRY:    AL contains character to test
                                ; EXIT:     ZF set iff AL contains space, tab, or equals
                                ; ERROR:    None
                                ; USES:     ZF
                                ; ------------------------------------------------------------------------------
                                isWhite:
000030BB 3C20                       cmp     al,' '          ; Space
000030BD 7406                       je      short iwX
000030BF 3C3D                       cmp     al,'='          ; Equals (treat as whitespace)
000030C1 7402                       je      short iwX
000030C3 3C09                       cmp     al,tab ; 9      ; Tab
                                iwX:
000030C5 C3                         retn
                                ; ------------------------------------------------------------------------------
                                ;*** unMarkUMB - marks a given UMB as unused, even if previously marked used
                                ; ------------------------------------------------------------------------------
                                ; ENTRY:    AL contains UMB number
                                ; EXIT:     None
                                ; ERROR:    None
                                ; USES:     Flags, variables in highvar.inc
                                ; ------------------------------------------------------------------------------
                                    ; 02/01/2023 - Retro DOS v4.2 (Modified MSDOS 6.21 IO.SYS)
                                unMarkUMB:
                                    ; 02/01/2023
                                    ;push   ax
                                    ;push   bx
                                    ;push   di
                                    ;push   es
                                    ;
                                    ;push   cs
                                    ;pop    es
000030C6 30E4                       xor     ah,ah
000030C8 89C3                       mov     bx,ax
                                    ; 19/04/2019
                                    ;;mov   byte [es:bx+UmbUsed],0
                                    ;mov    [es:bx+UmbUsed],ah ; 0
                                    ; 02/01/2023
                                    ; ds= cs
                                    ;mov    [cs:bx+UmbUsed],ah ; 0
000030CA 88A7[0024]                 mov     [bx+UmbUsed],ah ; 0
000030CE 3806[FF23]                 cmp     [UmbLoad],al
                                    ;cmp    [cs:UmbLoad],al
                                    ;;cmp   [es:UmbLoad],al
000030D2 7504                       jne     short umu10
                                    ;;mov   [es:UmbLoad],0 ; If unmarked the load UMB, load into convent.
                                    ;mov    [es:UmbLoad],ah ; 0
                                    ; 02/01/2023
                                    ; ds = cs
                                    ;mov    [cs:UmbLoad],ah ; 0
000030D4 8826[FF23]                 mov     [UmbLoad],ah ; 0
                                umu10:
                                    ;pop    es
                                    ;pop    di
                                    ;pop    bx
                                    ;pop    ax
000030D8 C3                         retn
                                ; ------------------------------------------------------------------------------
                                ;*** stowUMB - marks a given UMB as used, if it hasn't been so marked before
                                ;              -- accepts a UMB # in AL, and makes sure it hasn't yet been
                                ; listed in the /L:... chain. If it's the first one specified, it sets UmbLoad
                                ; to that UMB #... and in any case, it marks the UMB as specified.
                                ; ------------------------------------------------------------------------------
                                ; ENTRY:    AL contains UMB number, as specified by the user
                                ; EXIT:     None
                                ; ERROR:    Carry set if UMB # is less than 0 or >= MAXUMB (see highvar.inc)
                                ; USES:     AX, Flags, variables in highvar.inc
                                ; ------------------------------------------------------------------------------
                                    ; 01/01/2023 - Retro DOS v4.2
                                stowUMB:
000030D9 3C10                       cmp     al,MAXUMB ; 16
000030DB 7202                       jb      short su10
000030DD F9                         stc
000030DE C3                         retn                    ; Ooops-- UMB>=MAXUMB
                                su10:
                                    ; 01/01/2023
```

```
                                    ;push   bx
                                    ;push   di
                                    ;push   si
                                    ;push   ds
                                    ;push   es
                                    ;push   cs
                                    ;pop    es
                                    ;push   cs
                                    ;pop    ds

                                    ; 01/01/2023
                                    ; ds <> cs
                                    ;cmp    byte [cs:UmbLoad],0FFh
000030DF 2E803E[FF23]FF             cmp     byte [cs:UmbLoad],UNSPECIFIED
                                                        ; If this, we haven't been here before
000030E5 7504                       jne     short su20
000030E7 2EA2[FF23]                 mov     [cs:UmbLoad],al       ; So remember this UMB as the load UMB slot.

                                    ;;cmp   byte [UmbLoad],0FFh
                                    ;cmp    byte [UmbLoad],UNSPECIFIED ; If this, we haven't been here before
                                    ;jne    short su20
                                    ;mov    [UmbLoad],al   ; So remember this UMB as the load UMB slot.
                                su20:
000030EB 08C0                       or      al,al           ; If they gave UMB 0, there's really nothing
000030ED 740E                       jz      short su30      ; that we should do here.

                                    ;mov    bl,al
                                    ;xor    bh,bh
                                    ;mov    ax,1            ; Now, AX = 1, and BX = UMB Number
                                    ; 01/01/2023
000030EF 30E4                       xor     ah,ah
000030F1 89C3                       mov     bx,ax
000030F3 B001                       mov     al,1

                                    ;xchg   [es:bx+UmbUsed],al
                                    ; 01/01/2023
000030F5 2E8687[0024]               xchg    [cs:bx+UmbUsed],al

                                    ;or     al,al           ; If it was already 1, then al==1... and that
                                    ;jz     short su30      ; means an error.
                                    ;
                                    ;stc                    ; OOOPS!  This one's been used before.  :(

                                    ; 01/01/2023
000030FA 3C01                       cmp     al,1
000030FC F5                         cmc     ; if al > 0 -> cf = 1
                                su30:
                                    ; 01/01/2023
                                    ;pop    es
                                    ;pop    ds
                                    ;pop    si
                                    ;pop    di
                                    ;pop    bx
000030FD C3                         retn

                                ; 08/09/2023 - Retro DOS v4.2 IO.SYS (Optimization)
                                %if 0
                                ; ---------------------------------------------------------------------------
                                ;*** stowSiz - marks a given UMB as having a given minimum size
                                ; ---------------------------------------------------------------------------
                                ; ENTRY:    CL contains UMB number, AX contains size
                                ; EXIT:     None
                                ; ERROR:    None
                                ; USES:     AX, DX, Flags, variables in highvar.inc
                                ; ---------------------------------------------------------------------------

                                ; 13/05/2019

                                    ; 01/01/2023 - Retro DOS v4.2
                                stowSiz:
                                    ; 01/01/2023
                                    ;push   bx
                                    ;;push  di ; ?
                                    ;push   es

                                    ;push   cs
                                    ;pop    es

                                    mov     bl,cl               ; Now bl==UMB number, AX==size
                                    mov     bh,0                ;     bx==UMB number, AX==size
                                    shl     bl,1                ;     bx==offset into array, AX=size
                                    ;mov    [es:bx+UmbSize],ax  ; Store the size
                                    ; 01/01/2023
                                    mov     [cs:bx+UmbSize],ax  ; Store the size

                                    ; 01/01/2023
                                    ;pop    es
                                    ;;pop   di ; ?
                                    ;pop    bx

                                    retn
                                %endif

                                ; ---------------------------------------------------------------------------
                                ;*** toDigit - converts a character-digit to its binary counterpart
                                ;           -- verifies that CL contains a valid character-digit; if so, it
                                ; changes CL to its counterpart binary digit ((CL-'0') or (CL-'A'+10)).
                                ; A-F are considered valid iff gnradix is 16.
                                ; ---------------------------------------------------------------------------
                                ; ENTRY:    CL contains a digit ('0' to '9' or, if gnradix==16, 'A' to 'F')
                                ; EXIT:     CL contains digit in binary (0 to 9 or, if gnradix==16, 0 to 15)
                                ; ERROR:    Carry set indicates invalid digit; carry clear indicates good digit
                                ; USES:     CL, Flags
                                ; ---------------------------------------------------------------------------
                                ; If the string is preceeded with "0x", the value is read as hexadecimal; else,
                                ; as decimal. After a read, you may check the radix by examining gnradix--it
                                ; will be 10 or 16.
                                ; ---------------------------------------------------------------------------

                                gnradix:
000030FE 0000                       dw      0               ; Must be a word--16x16 multiplication

                                toDigit:
00003100 2E833E[FE30]10             cmp     word [cs:gnradix],16
00003106 751C                       jne     short td20      ; Don't check hex digits if radix isn't 16

                                toDigit_hex:
00003108 80F961                     cmp     cl,'a'  ; 61h
0000310B 7209                       jb      short td10
0000310D 80F966                     cmp     cl,'f'  ; 66h
00003110 7720                       ja      short tdE       ; Nothing valid above 'z' at all...
00003112 80E957                     sub     cl,'a'-10 ; 57h         ; Make 'a'==10 and return.
                                    ;clc                    ; <- CLC is implicit from last SUB
00003115 C3                         retn
```

```
                                td10:
00003116 80F941                     cmp     cl,'A'  ; 41h
00003119 7209                       jb      short td20     ; Below 'A'?  Not a letter...
0000311B 80F946                     cmp     cl,'F'  ; 46h
0000311E 7712                       ja      short tdE      ; Above 'F'?  Not a digit.
00003120 80E937                     sub     cl,'A'-10 ; 37h       ; Make 'A'==10 and return.
                                    ;clc                   ; <- CLC is implicit from last SUB
00003123 C3                         retn
                                toDigit_dec:
                                td20:
00003124 80F930                     cmp     cl,'0'         ; If less than zero,
                                    ;jb     short tdE      ; Done.
00003127 720A                       jb      short tdEr ; 08/04/2019
00003129 80F939                     cmp     cl,'9'         ; Or, if greater than nine,
0000312C 7704                       ja      short tdE      ; Done.
0000312E 80E930                     sub     cl,'0'  ; 30h  ; Okay--make '0'==0 and return.
                                    ;clc                   ; <- CLC is implicit from last SUB
00003131 C3                         retn
                                tdE:
00003132 F9                         stc
                                tdEr:               ; 08/04/2019 - Retro DOS v4.0
00003133 C3                         retn

                                ; ---------------------------------------------------------------------------
                                ;*** GetXNum - reads a 32-bit ASCII number at ES:SI and returns it in DX:AX
                                ; ---------------------------------------------------------------------------
                                ; ENTRY:    ES:SI points to an ascii string to scan
                                ; EXIT:     ES:SI moved to first invalid digit, DX:AX contains value read
                                ; ERROR:    Carry set if # is too big, or has no digits (EOL possibly)
                                ; USES:     ES:SI, DX, AX, Flags, gnradix
                                ; ---------------------------------------------------------------------------
                                ; If the string is preceeded with "0x", the value is read as hexadecimal; else,
                                ; as decimal. After a read, you may check the radix by examining gnradix--it
                                ; will be 10 or 16.
                                ; ---------------------------------------------------------------------------

                                ; 08/04/2019 - Retro DOS v4.0

                                ; 01/01/2023 - Retro DOS v4.2 (Modified MSDOS 6.21 IO.SYS)
                                ; (SYSINIT:3109h)

                                GetXNum:
                                    ;pushreg <bx, cx, ds>
                                    ; 01/01/2023
                                    ;push   bx
00003134 51                         push    cx ; *
                                    ;push   ds

00003135 FC                         cld
00003136 31C0                       xor     ax,ax
00003138 31DB                       xor     bx,bx
0000313A 31C9                       xor     cx,cx
0000313C 31D2                       xor     dx,dx                   ; Start with 0 (makes sense)

0000313E 2EC706[FE30]0A00           mov     word [cs:gnradix],10  ; And default to a radix of 10 (dec)

00003145 268A0C                     mov     cl,[es:si]              ; Now AX=0, BX=0, CH=0/CL=char, DX=0
                                    ;call   toDigit
00003148 E8D9FF                     call    toDigit_dec
                                    ;jc     short gxnE              ; If it's not a digit, leave now.
                                    ; 01/01/2023
0000314B 7233                       jc      short gxnX

0000314D 08C9                       or      cl,cl
0000314F 7517                       jnz     short gxn20             ; Doesn't have '0x'
00003151 268A4C01                   mov     cl,[es:si+1]
00003155 80F978                     cmp     cl,'x'                  ; Either 'x'...
00003158 7405                       je      short gxn10
0000315A 80F958                     cmp     cl,'X'                  ; ...or 'X' means it's hexadecimal
0000315D 7509                       jne     short gxn20

                                gxn10:
0000315F 2EC706[FE30]1000           mov     word [cs:gnradix], 16
00003166 46                         inc     si                      ; Since we read "0x", march over it.
00003167 46                         inc     si

                                    ; ---------------------------------------------------
                                    ; GXN20--ES:SI = a digit in a number; if not, we're done
                                    ;       DX:AX = current total
                                    ;       BX    = 0
                                    ;       CH    = 0
                                    ; ---------------------------------------------------

                                gxn20:
00003168 268A0C                     mov     cl,[es:si]      ; Now DX:AX=current total, CH=0/CL=char
0000316B 46                         inc     si

0000316C E891FF                     call    toDigit         ; Accepts only valid digits, A-F -> 10-16
0000316F 720D                       jc      short gxnQ      ; <- Ah... wasn't a digit. Stop.

00003171 E80E00                     call    mul32           ; Multiply DX:AX by gnradix
00003174 720A                       jc      short gxnX      ; (if it's too big, error out)

00003176 01C8                       add     ax,cx           ; Add the digit
00003178 11DA                       adc     dx,bx           ; (BX is 0!)--Adds 1 iff last add wrapped
                                    ;jc     short gxnX      ; If _that_ wrapped, it's too big.
                                    ;jmp    short gxn20
0000317A 73EC                       jnc     short gxn20
                                gxnE:
                                    ;stc                    ; In this case, we need to set the carry
0000317C EB02                       jmp     short gxnX      ; and leave--there were no digits given.
                                gxnQ:
0000317E 4E                         dec     si              ; Don't read in the offensive character.
0000317F F8                         clc                     ; And clear carry, so they know it's okay.
                                gxnX:
                                    ; 01/01/2023
                                    ;pop    ds
00003180 59                         pop     cx ; *
                                    ;pop    bx
00003181 C3                         retn

                                ; ---------------------------------------------------------------------------
                                ;*** mul32 - multiplies the number in DX:AX by gnradix
                                ; ---------------------------------------------------------------------------
                                ; ENTRY:   DX:AX = the number to be multiplied, BX = 0, gnradix = multiplier
                                ; EXIT:    DX:AX has been multiplied by gnradix if carry clear; BX still 0
                                ; ERROR:   Carry set if number was too large
                                ; USES:    Flags, AX, DX
                                ; ---------------------------------------------------------------------------

                                mul32:
00003182 50                         push    ax              ; DX=old:hi, AX=old:lo, TOS=old:lo, BX=0
00003183 89D0                       mov     ax,dx           ; DX=old:hi, AX=old:hi, TOS=old:lo, BX=0
00003185 2EF726[FE30]               mul     word [cs:gnradix] ; DX=?, AX=new:hi, TOS=old:lo, BX=0
```

```
0000318A 7211                       jc      short m32E      ; Too big?
                                    
0000318C 89C2                       mov     dx,ax           ; DX=new:hi, AX=new:hi, TOS=old:lo, BX=0
0000318E 58                         pop     ax              ; DX=new:hi, AX=old:lo, TOS=orig, BX=0
                                    
0000318F 87D3                       xchg    dx,bx           ; DX=0, AX=old:lo, TOS=orig, BX=new:hi
00003191 2EF726[FE30]               mul     word [cs:gnradix] ; DX=carry,  AX=new:lo, TOS=orig, BX=new:hi
00003196 87D3                       xchg    dx,bx           ; DX=new:hi, AX=new:lo, TOS=orig, BX=carry
00003198 01DA                       add     dx,bx           ; DX=new:hi, AX=new:lo, TOS=orig, BX=carry
0000319A 31DB                       xor     bx,bx           ; DX=new:hi, AX=new:lo, TOS=orig, BX=0
0000319C C3                         retn
                                  m32E:
0000319D 58                         pop     ax
0000319E C3                         retn
                                    
                                  ; ---------------------------------------------------------------------------
                                  ;*** toPara - divides DX:AX by 16; result in AX only (discards extra DX data)
                                  ; ---------------------------------------------------------------------------
                                  ; ENTRY:   DX:AX = the number to be divided
                                  ; EXIT:    Interpereting DX:AX as bytes, AX=paragraph equivalent, 0xFFFF max
                                  ; ERROR:   None
                                  ; USES:    Flags, AX, DX
                                  ; ---------------------------------------------------------------------------
                                  ; Note: The 386 has a 32-bit SHR, which would work perfectly for this... but we
                                  ;       can't ensure a 386 host machine. Sorry.
                                  ; ---------------------------------------------------------------------------
                                    
                                      ; 01/01/2023 - Retro DOS v4.2
                                  toPara:
0000319F 51                         push    cx              ; DX:AX=HHHH hhhh hhhh hhhh:LLLL llll llll llll
                                    
000031A0 B104                       mov     cl,4            ;
000031A2 D3E8                       shr     ax,cl           ; DX:AX=HHHH hhhh hhhh hhhh:0000 LLLL llll llll
000031A4 92                         xchg    ax,dx           ; DX:AX=0000 LLLL llll llll:HHHH hhhh hhhh hhhh
000031A5 B10C                       mov     cl,12
000031A7 D3E0                       shl     ax,cl           ; DX:AX=0000 LLLL llll llll:hhhh 0000 0000 0000
000031A9 09D0                       or      ax,dx           ;    AX=hhhh LLLL llll llll
                                    
000031AB 59                         pop     cx
000031AC C3                         retn
                                    
                                  ; ---------------------------------------------------------------------------
                                  ;*** UmbHead - returns in AX the address of the first UMB block (0x9FFF)
                                  ; ---------------------------------------------------------------------------
                                  ; ENTRY:  Nothing
                                  ; EXIT:   AX contains 0x9FFF for most systems
                                  ; ERROR:  Carry set if pointer is 0xFFFF (if not set up yet--DH runs into this)
                                  ; USES:   Flags, AX
                                  ; ---------------------------------------------------------------------------
                                  ; Early in the boot-cycle, the pointer used to obtain this value isn't set up;
                                  ; to be precise, before a UMB provider is around. In this event, the pointer
                                  ; is always set to 0xFFFF; it changes once a provider is around. On most
                                  ; machines (all of 'em I've seen), it changes to 0x9FFF at that point.
                                  ; ---------------------------------------------------------------------------
                                    
                                      ; 01/01/2023 - Retro DOS v4.2 (Modified MSDOS 6.21 IO.SYS)
                                  UmbHead:
                                      ; 13/05/2019 (because of callers, pushs & pops are not needed here)
                                    
                                      ;push  si ; ?
                                      ;push  ds ; ?
                                      ;push  es
                                      ;push  bx ; *
                                    
                                      ; 09/04/2019
                                      ; !!! No need to save es,bx,ds,si above !!! (es,bx are changed here)
                                    
000031AD B452                         mov     ah,GET_IN_VARS        ; Call int 21h, function 52h...
000031AF CD21                         int     21h
                                    
000031B1 26A18C00                     mov     ax,[es:DOS_UMB_HEAD]  ; And read what's in ES:[008C]
                                    
                                      ; 01/01/2023
000031B5 83F8FF                       cmp     ax,0FFFFh
000031B8 F5                           cmc
                                      ; if AX=0FFFFh -> CF=1
000031B9 C3                           retn
                                    
                                  ; 01/01/2023
                                  ;%if 0
                                  ;   cmp     ax,0FFFFh
                                  ;   je      short uhE             ; If it's 0xFFFF, it's an error...
                                  ;
                                  ;   clc                           ; Else, it isn't (CLC done by prev cmp)
                                  ;   ;jmp    short uhX
                                  ;   ; 12/12/2022
                                  ;   retn
                                  ;uhE:
                                  ;   stc
                                  ;uhX:
                                  ;   ;pop    bx ; *
                                  ;   ;pop    es
                                  ;   ;pop    ds ; ?
                                  ;   ;pop    si ; ?
                                  ;   retn
                                  ;%endif
                                    
                                  ; ---------------------------------------------------------------------------
                                  ;*** isSysMCB - sets ZF if ES points to an MCB owned by "SC" + (8 or 9)
                                  ; ---------------------------------------------------------------------------
                                  ; ENTRY:  ES:0 should point to a valid MCB
                                  ; EXIT:   ZF set if owned by SC+8 or SC+9 (for japan)
                                  ; USES:   Flags
                                  ; ---------------------------------------------------------------------------
                                    
                                  isSysMCB:
                                      ;push  ax
                                    
                                      ;mov   ax,[es:ARENA.OWNER]   ; Check the owner...
                                      ;cmp   ax,SystemPSPOwner     ; 8 (for US OR Japan) is valid
                                      ;je    short ism10
                                      ;cmp   ax,JapanPSPOwner      ; 9 (for Japan) is valid
                                      ;;je   short ism10
                                      ;;jmp  short ismX            ; Anything else isn't.
                                      ;jne   short ismX
000031BA 26833E010008                 cmp     word [es:ARENA.OWNER],SystemPSPOwner ; 8  ; 09/04/2019
000031C0 7507                         jne     short ismX
                                  ism10:
                                      ;mov   ax,[es:ARENA.NAME]    ; Check the name...
                                      ;cmp   ax,'SC' ; 4353h
000031C2 26813E08005343               cmp     word [es:ARENA.NAME],'SC'
                                  ismX:
                                      ;pop   ax
000031C9 C3                           retn
```

```
                                ; 09/04/2019 - Retro DOS v4.0
                                ; ----------------------------------------------------------------------------
                                ;*** AddrToUmb - converts a segment address in AX to its appropriate UMB number
                                ; ----------------------------------------------------------------------------
                                ; ENTRY:  AX contains a segment address
                                ; EXIT:   AX will contain the UMB number which contains the address (0==conv)
                                ; ERROR:  If the address is above UM Range, AX will return as FFFF.
                                ; USES:   Flags, AX
                                ; ----------------------------------------------------------------------------
                                ; An address in the following areas is treated as:
                                ;    0      <-> umbhead (0x9FFF)          = Conventional memory
                                ;    0x9FFF <-> addr of first UM sys MCB  = UMB #1
                                ;      ...
                                ;    addr of last UM sys MCB <-> TOM      = invalid; returns #0xFFFF
                                ; ----------------------------------------------------------------------------

                                    ; 01/01/2023 - Retro DOS v4.2
                                AddrToUmb:
                                    ; 01/01/2023
                                    ;push   cx
                                    ;push   dx
000031CA 06                         push    es

000031CB 89C2                       mov     dx,ax           ; DX = address to search for

000031CD E8DDFF                     call    UmbHead         ; AX = first segment
000031D0 7222                       jc      short atuE      ; If it couldn't get it, error out.

                                    ; 22/07/2023
                                    ;mov    es,ax ; *       ; ES = first UMB segment
000031D2 31C9                       xor     cx,cx ; 0       ; Pretend we're on UMB 0 for now... (cx = UMB#)

                                    ; 22/07/2023
                                atu10:
000031D4 8EC0                       mov     es,ax ; * ; ** ; 22/07/2023
                                ; ----------------------------------------
                                ; ATU10--ES - Current MCB address
                                ;        DX - Address given for conversion
                                ;        CX - Current UMB #
                                ; ----------------------------------------

                                ;atu10:
                                    ;mov    ax,es ; * ; 18/07/2023
000031D6 39D0                           cmp ax,dx           ; Present segment >= given segment?
000031D8 731D                       jae     short atuX      ; Yep--done.

000031DA E8DDFF                     call    isSysMCB        ; Returns with ZF set if this is a system MCB
000031DD 7501                       jnz     short atu20

000031DF 41                         inc     cx              ; If it _was_ a system MCB, we're in a new UMB.
                                atu20:
                                    ;mov    al,[es:ARENA.SIGNATURE]
                                    ;cmp    al,arena_signature_end  ; 'Z'
                                    ; 22/07/2023
                                    ; ax = es
                                    ;mov    ax,es ; **
000031E0 2603060300                 add     ax,[es:ARENA.SIZE]
000031E5 26803E00005A               cmp     byte [es:ARENA.SIGNATURE],arena_signature_end
000031EB 7403                       je      short atu30             ; 'Z' means this was the last MCB... that's it.

                                    ;NextMCB es,ax

                                    ;mov    ax,es ; **
                                    ;;add   ax,[es:3]
                                    ;add    ax,[es:ARENA.SIZE]
000031ED 40                         inc     ax
                                    ; 22/07/2023
                                    ;mov    es,ax ; *
000031EE EBE4                       jmp     short atu10

                                ; ----------------------------------------------------------------------------
                                ; if we get to atu30, they specified a number that was past the last MCB.
                                ; make sure it's not _inside_ that MCB before we return an error condition.
                                ; ----------------------------------------------------------------------------

                                atu30:
                                    ; 22/07/2023
                                    ; ax = es + [es:ARENA.SIZE]
                                    ;mov    ax,es ; **
                                    ;add    ax,[es:ARENA.SIZE] ; **
000031F0 39D0                       cmp     ax,dx           ; Present >= given?
000031F2 7303                       jae     short atuX      ; Yep! It _was_ inside.
                                atuE:
000031F4 31C9                       xor     cx,cx ; 0       ; Else, fall through with UMB # == -1
000031F6 49                         dec     cx              ; (that makes it return 0xFFFF and sets CF)
                                atuX:
000031F7 89C8                       mov     ax,cx           ; Return the UMB number in AX

000031F9 07                         pop     es
                                    ; 01/01/2023
                                    ;pop    dx
                                    ;pop    cx
000031FA C3                         retn

                                ; ----------------------------------------------------------------------------
                                ;*** convUMB - checks after GetXNum to convert an address to a UMB number
                                ;              -- if GetXNum read a hex number, we interperete that as a segment
                                ; address rather than a UMB number... and use that address to look up a UMB.
                                ; This routine checks for that condition and calls AddrToUmb if necessary.
                                ; ----------------------------------------------------------------------------
                                ; ENTRY:  AX contains a UMB number or segment, gnradix has been set by GetXNum
                                ; EXIT:   AX will contain a UMB number
                                ; ERROR:  None
                                ; USES:   Flags, AX
                                ; ----------------------------------------------------------------------------

                                    ; 01/01/2023 - Retro DOS v4.2
                                convUMB:
000031FB 2E833E[FE30]10             cmp     word [cs:gnradix],16
00003201 7507                       jne     short cu10      ; If it didn't read in hex, it's not an address
00003203 E8C4FF                     call    AddrToUmb       ; Else, convert the address to a UMB number
                                    ;cmp    ax,0FFFFh
                                    ;jne    short cu10
                                    ;inc    ax              ; If too high, ignore it (make it conventional)
                                    ; 01/01/2023
00003206 40                         inc     ax
00003207 7401                       jz      short cu10      ; If too high, ignore it (make it conventional)
00003209 48                         dec     ax
                                cu10:
0000320A C3                         retn

                                ; 01/01/2023 - Retro DOS v4.2
```

```
                        ;%if 0
                        ;
                        ;; ---------------------------------------------------------------------------
                        ;;*** setUMBs - links umbs and sets allocation strategy for a load
                        ;;              -- if LoadHigh, the allocation strategy MAY be LOW_FIRST instead
                        ;; of the usual HIGH_FIRST. See the code.
                        ;; ---------------------------------------------------------------------------
                        ;; ENTRY:  None
                        ;; EXIT:   None
                        ;; ERROR:  None
                        ;; USES:   Flags, fm_umb, fm_strat
                        ;; ---------------------------------------------------------------------------
                        ;
                        ;setUMBs:
                        ;   push    ax
                        ;   push    bx
                        ;   call    fm_link
                        ;   pop     bx
                        ;   pop     ax
                        ;   retn
                        ;
                        ;%endif

                        ; 18/07/2023 - Retro DOS v4.2 IO.SYS (optimization)
                        ; loadLow subroutine is not used anywhere of IO.SYS 6.22 (& 5.0)
                        %if 0

                        ; ---------------------------------------------------------------------------
                        ;*** loadLow - returns AL==0 if UMB0 == 0, else AL==1
                        ; ---------------------------------------------------------------------------
                        ; ENTRY:  None
                        ; EXIT:   AL==0 if mem strategy should be set to LOW_FIRST, else AL==1
                        ;         Carry set if UMB0 not specified (_NOT_ an error)
                        ; ERROR:  None
                        ; USES:   Flags, fm_strat, fm_umb
                        ; ---------------------------------------------------------------------------
                        ; We want to set the memory strategy to LOW_FIRST if the user specified a
                        ; load UMB, and it is 0. That 0 can be either from the user having _specified_
                        ; zero (/L:0;...), or from having specified a too-big min size (/L:1,99999999)
                        ; such that the load UMB is too small, and shouldn't be used.
                        ; ---------------------------------------------------------------------------

                        loadLow:
                            ;push  ds
                            ;push  cs              ; Point DS into appropriate data segment
                            ;pop   ds

                            ;mov   al,[UmbLoad]
                            mov     al,[cs:UmbLoad]
                            cmp     al,UNSPECIFIED ; 0FFh, -1
                            jne     short ll10

                            stc
                        ll15:
                            mov     al,1           ; Return with AL==1 && STC if no UMBs specified
                            ;stc
                            ;jmp    short llX
                            retn
                        ll10:
                            or      al,al          ; AL=the load UMB: Is it == 0?
                            ;jz     short llX      ; Yep... CF==0 (from OR) && AL=0, so just exit

                            jnz     short ll15     ; 09/04/2019 - Retro DOS v4.0
                            retn

                            ;mov    al,1
                            ;clc
                        ;llX:
                            ;pop    ds             ; Return DS to where it was
                            ;retn

                        %endif

                        ; ---------------------------------------------------------------------------
                        ;*** HideUMBs - links UMBs and hides upper-memory as appropriate
                        ; ---------------------------------------------------------------------------
                        ; ENTRY:  None
                        ; EXIT:   None
                        ; ERROR:  None
                        ; USES:   Flags, fm_strat, fm_umb
                        ; ---------------------------------------------------------------------------

                            ; 01/01/2023 - Retro DOS v4.2 (Modified MSDOS 6.21 IO.SYS)
                            ; (SYSINIT:322Fh)
                        HideUMBs:
                            ; 01/01/2023
                            ;push  ax
                            ;push  cx
                            ;push  ds
0000320B 06                 push    es

                            ; 01/01/2023
                            ; ds = cs

0000320C E86E02             call    UmbTest        ; See if we REALLY linked in anything...
0000320F 7232               jc      short husX     ; ...if not, there's nothing for us to do.

00003211 E892FD             call    FixMem         ; Concatenate adjacent free MCBs in upper mem

                            ;call  setUMBs        ; Link UMBs and set memory-allocation strategy
                            ; 01/01/2023
00003214 E8DCFD             call    fm_link

                            ;putdata fInHigh,1     ; Remember that we're now running high
                            ;mov    byte [cs:fInHigh],1
                            ; 01/01/2023
00003217 C606[FB23]01       mov     byte [fInHigh],1

                            ;call  GetLoadUMB     ; See if they gave us a list to leave free
                            ;mov    al,[cs:UmbLoad] ; 09/04/2019 - Retro DOS v4.0
                            ; 01/01/2023
0000321C A0[FF23]           mov     al,[UmbLoad]

0000321F 3CFF               cmp     al,UNSPECIFIED ; If they didn't,
00003221 7420               je      short husX     ; then we shouldn't do this loop:

00003223 31C9               xor     cx,cx

                        ; ---------------------------------------------
                        ; HUS10-CX - UMB number (after inc, 1==first UMB)
                        ; ---------------------------------------------

00003225 41             hus10:      inc     cx             ; For each UMB:
                            ; 01/01/2023
```

```
37048 00003226 80F910                   cmp     cl,MAXUMB
37049                                   ;cmp    cx,MAXUMB ; 16
37050 00003229 730E                     jae     short hus20
37051
37052 0000322B 88C8                     mov     al,cl           ; (stopping as soon as we're outside of the
37053 0000322D 06                       push    es
37054 0000322E E8A200                   call    findUMB         ; valid range of UMBs)
37055 00003231 07                       pop     es              ; push/pop: trash what findumb finds. :-)
37056 00003232 7205                     jc      short hus20
37057
37058                                   ; 02/01/2023
37059                                   ;push   cx ; *
37060 00003234 E84F01                   call    _hideUMB_       ; hide what we need to hide.
37061                                   ;pop    cx ; *
37062
37063 00003237 EBEC                     jmp     short hus10
37064                                 hus20:
37065                                   ;call   GetLoadUMB      ; Now check if they offered /L:0
37066                                   ; 01/01/2023
37067                                   ; ds = cs
37068                                   ;mov    al,[UmbLoad]
37069                                   ;;mov   al,[cs:UmbLoad] ; 09/04/2019 - Retro DOS v4.0
37070 00003239 800E[FF23]00             or      byte [UmbLoad],0
37071                                   ;or     al,al           ; --Is the load UMB 0? (-1==unspecified)
37072 0000323E 7503                     jnz     short husX      ; If not, we're done.
37073
37074 00003240 E86802                   call    hl_unlink       ; If so, however, fix UMBs and strategy.
37075                                 husX:
37076 00003243 07                       pop     es
37077                                   ; 01/01/2023
37078                                   ;pop    ds
37079                                   ;pop    cx
37080                                   ;pop    ax
37081 00003244 C3                       retn
37082
37083                                 ; ------------------------------------------------------------------------------
37084                                 ;*** GetLoadUMB - Returns the load UMB number in AL (-1 if not specified)
37085                                 ; ------------------------------------------------------------------------------
37086                                 ; ENTRY:  None
37087                                 ; EXIT:   AL == load UMB
37088                                 ; ERROR:  None
37089                                 ; USES:   Flags, AX
37090                                 ; ------------------------------------------------------------------------------
37091
37092                                 ;GetLoadUMB:
37093                                 ;   ;getdata al, UmbLoad
37094                                 ;   push    ds
37095                                 ;   push    cs
37096                                 ;   pop     ds
37097                                 ;   mov     al,[UmLoad]
37098                                 ;   pop     ds
37099                                 ;   retn
37100
37101                                 ; ------------------------------------------------------------------------------
37102                                 ;*** GetLoadSize - Returns the load UMB minimum size (0 if not specified)
37103                                 ; ------------------------------------------------------------------------------
37104                                 ; ENTRY:  None
37105                                 ; EXIT:   AX == load UMB minimum size
37106                                 ; ERROR:  None
37107                                 ; USES:   Flags, AX
37108                                 ; ------------------------------------------------------------------------------
37109
37110                                 ; 08/09/2023 - Retro DOS v4.2 IO.SYS (Optimization)
37111                                 %if 0
37112                                     ; 01/01/2023 - Retro DOS v4.2
37113                                 GetLoadSize:
37114                                     ; 09/04/2019 - Retro DOS v4.0
37115                                     ;mov    al,[cs:UmbLoad]
37116                                     ; 01/01/2023
37117                                     ; ds = cs
37118                                     mov     al,[UmbLoad]
37119                                     ;jmp    short GetSize
37120
37121                                     ;push   bx
37122                                     ;;push  si
37123                                     ;push   ds
37124                                     ;push   cs
37125                                     ;pop    ds
37126
37127                                     ;mov    al,[UmbLoad]
37128
37129                                     ;xor    ah,ah                   ;    ax==UMB
37130                                     ;mov    bx,UmbSize              ;    bx==array
37131                                     ;shl    al,1                     ;    ax==offset
37132                                     ;;add   ax,bx                   ;    ax==element index
37133                                     ;;mov   si,ax                   ; ds:si==element index
37134
37135                                     ;;lodsw                         ;    hh
37136
37137                                     ;add    bx,ax
37138                                     ;mov    ax,[bx]
37139
37140                                     ;pop    ds
37141                                     ;;pop   si
37142                                     ;pop    bx
37143                                     ;retn
37144                                 %endif
37145
37146                                 ; ------------------------------------------------------------------------------
37147                                 ;*** GetSize - Returns the UMB in AL's minimum size (0 if not specified)
37148                                 ; ------------------------------------------------------------------------------
37149                                 ; ENTRY:  AL == a UMB number
37150                                 ; EXIT:   AX == UMB minimum size, as specified by the user
37151                                 ; ERROR:  None
37152                                 ; USES:   Flags, AX
37153                                 ; ------------------------------------------------------------------------------
37154
37155                                 ; 08/09/2023 - Retro DOS v4.2 IO.SYS (Optimization)
37156                                 GetLoadSize:
37157                                     ; ds = cs
37158                                     ;mov    al,[UmbLoad]
37159                                     ; al = [UmbLoad]
37160                                     ; ....
37161
37162                                     ; 01/01/2023 - Retro DOS v4.2
37163                                 GetSize:
37164                                     ; 09/04/2019 - Retro DOS v4.0
37165
37166                                     ;push   bx ; 01/01/2023
37167                                     ;push   si
37168                                     ;push   ds
37169                                     ;push   cs
37170                                     ;pop    ds
37171
```

```
00003245 30E4                       xor     ah,ah                   ;    ax==UMB
00003247 BB[1024]                   mov     bx,UmbSize              ;    bx==array
0000324A D0E0                       shl     al,1                     ;    ax==offset
                                    ;add    ax,bx                   ;    ax==element index
                                    ;mov    si,ax                   ; ds:si==element index

                                    ;lodsw                          ;    ax==size

0000324C 01C3                       add     bx,ax
                                    ; 01/01/2023
                                    ; ds = cs
0000324E 8B07                       mov     ax,[bx]
                                    ;mov    ax,[cs:bx]

                                    ;pop    ds
                                    ;pop    si
                                    ;pop    bx ; 01/01/2023
                                sls10:      ; 08/09/2023
00003250 C3                         retn

                                ; ----------------------------------------------------------------------------
                                ;*** StoLoadUMB - Overrides the load UMB number with what's in AL
                                ; ----------------------------------------------------------------------------
                                ; ENTRY:   AL == new load UMB
                                ; EXIT:    None
                                ; ERROR:   None
                                ; USES:    Flags, AX
                                ; ----------------------------------------------------------------------------
                                ; CAUTION: Should only be used if /L:... was used.  Logically, that is the only
                                ;          time you would ever need this, so that's okay.
                                ; ----------------------------------------------------------------------------

                                ; StoLoadUMB subroutine is not used anywhere
                                ; of PCDOS 7.1 IBMBIO.COM (& MSDOS 6.21 IO.SYS)
                                ; Erdogan Tan - 18/07/2023

                                ;StoLoadUMB:
                                ;   ;putdata UmbLoad, al
                                ;   push    es
                                ;   push    cs
                                ;   pop     es              ; mov [cs:UmbLoad], al !!!! ; 08/09/2023
                                ;   mov     [es:UmbLoad],al
                                ;   pop     es
                                ;   retn

                                ; ----------------------------------------------------------------------------
                                ;*** StoLoadSize - Overrides the load UMB minimum size with what's in AX
                                ; ----------------------------------------------------------------------------
                                ; ENTRY:  AL == new load size
                                ; EXIT:   None
                                ; ERROR:  None
                                ; USES:   Flags, AX
                                ; ----------------------------------------------------------------------------
                                    ; 08/09/2023 - Retro DOS v4.2 IO.SYS (Optimization)
                                    ; 01/01/2023 - Retro DOS v4.2
                                StoLoadSize:
                                    ; 01/01/2023
                                    ;push  dx

                                    ;getdata dl, UmbLoad         ; Put UMB# in DL and size in AX
                                    ;
                                    ;push  ds
                                    ;push  cs
                                    ;pop   ds
                                    ;mov   dl,[UmbLoad]
                                    ;pop   ds

                                    ; 08/09/2023
                                    ; MSDOS 6.21 IO.SYS - SYSINIT:32B6h
                                    ; PCDOS 7.1 IBMBIO.COM - SYSINIT:3831h

                                    ;;mov  dl,[UmbLoad]          ; BUG ! CL would/must be used here
                                                                 ; instead of DL (*) ; 18/07/2023
                                    ;mov   dl,[cs:UmbLoad] ; Retro DOS v4.0, v4.1, v4.2
                                    ;cmp   dl,UNSPECIFIED ; 0FFh
                                    ;je    short sls10

                                           ; BUG ! stowSiz uses CL instead of DL !
                                           ; (CL is set in ParseL which calls stowSiz)
                                           ; (This BUG existing in PCDOS 7.1 IBMBIO.COM also)
                                           ; Erdogan Tan - 18/07/2023

                                    ; 08/09/2023 (BugFix)
                                    ;mov    cl,[cs:UmbLoad]
                                    ; 08/09/2023
                                    ; ds = cs
00003251 8A0E[FF23]                 mov     cl,[UmbLoad]
00003255 80F9FF                     cmp     cl,UNSPECIFIED ; 0FFh
00003258 74F6                       je      short sls10

                                    ; 08/09/2023
                                ;   call    stowSiz             ; We've got a function to do just this
                                ;sls10:
                                ;   ; 01/01/2023
                                ;   ;pop    dx
                                ;   retn

                                    ; 08/09/2023
                                    ;;jmp   stowSiz
                                    ;jmp    short stowSiz

                                ; 08/09/2023 - Retro DOS v4.2 IO.SYS (Optimization)
                                %if 1
                                ; ----------------------------------------------------------------------------
                                ;*** stowSiz - marks a given UMB as having a given minimum size
                                ; ----------------------------------------------------------------------------
                                ; ENTRY:    CL contains UMB number, AX contains size
                                ; EXIT:     None
                                ; ERROR:    None
                                ; USES:     AX, DX, Flags, variables in highvar.inc
                                ; ----------------------------------------------------------------------------

                                ; 13/05/2019

                                    ; 01/01/2023 - Retro DOS v4.2
                                stowSiz:
                                    ; 01/01/2023
                                    ;push  bx
                                    ;;push di ; ?
                                    ;push  es

                                    ;push  cs
                                    ;pop   es
```

```
37296 0000325A 88CB                     mov     bl,cl                   ; Now bl==UMB number, AX==size
37297 0000325C B700                     mov     bh,0                    ;     bx==UMB number, AX==size
37298 0000325E D0E3                     shl     bl,1                    ;     bx==offset into array, AX=size
37299                                   ;mov    [es:bx+UmbSize],ax      ; Store the size
37300                                   ; 01/01/2023
37301 00003260 2E8987[1024]             mov     [cs:bx+UmbSize],ax      ; Store the size
37302
37303                                   ; 01/01/2023
37304                                   ;pop    es
37305                                   ;;pop   di ; ?
37306                                   ;pop    bx
37307
37308 00003265 C3                       retn
37309                                 %endif
37310
37311                                 ; ----------------------------------------------------------------------------
37312                                 ;*** hideUMB - marks as HIDDEN all FREE elements in UMB passed as AL
37313                                 ; ----------------------------------------------------------------------------
37314                                 ; ENTRY:    AL must indicate a valid UMB; 0==conv && is invalid.
37315                                 ; EXIT:     None; free elements in UMB marked as hidden
37316                                 ; ERROR:    None
37317                                 ; USES:     Flags
37318                                 ; ----------------------------------------------------------------------------
37319
37320                                     ; 01/01/2023 - Retro DOS v4.2
37321                                 hideUMB:
37322                                     ; 02/01/2023
37323 00003266 52                       push    dx ; (*)
37324                                   ; 01/01/2023
37325                                   ;push   ax
37326 00003267 06                       push    es
37327
37328 00003268 E86800                   call    findUMB ; (*)   ; Returns with carry if err, else ES == MCB
37329 0000326B 7224                     jc      short huX
37330
37331                                 ; ------------------------------------------------
37332                                 ; HU10--ES - MCB inside UMB; if it's a system MCB,
37333                                 ;            we're not in the same UMB, so exit.
37334                                 ; ------------------------------------------------
37335
37336 0000326D E84AFF                 hu10:       call    isSysMCB        ; Returns with ZF set if owner is SYSTEM
37337 00003270 741F                       jz      short huX       ; If it is, we've finished the UMB.
37338                                     ;call   isFreeMCB       ; Returns with ZF set if owner is 0
37339 00003272 26830E010000               or      word [es:ARENA.OWNER],0
37340 00003278 7503                       jnz     short hu20
37341
37342 0000327A E81700                     call    hideMCB
37343                                 hu20:
37344                                     ;mov    al,[es:ARENA.SIGNATURE]
37345                                     ;cmp    al,arena_signature_end  ;'Z'
37346                                     ; 19/07/2023
37347 0000327D 26803E00005A               cmp     byte [es:ARENA.SIGNATURE],'Z'
37348 00003283 740C                       jz      short huX       ; 'Z' means this was the last MCB... that's it.
37349
37350                                     ;NextMCB es,ax          ; Go on forward.
37351 00003285 8CC0                       mov     ax,es
37352                                     ;add    ax,[es:3]
37353 00003287 2603060300                 add     ax,[es:ARENA.SIZE]
37354 0000328C 40                         inc     ax
37355 0000328D 8EC0                       mov     es,ax
37356
37357 0000328F EBDC                       jmp     short hu10
37358                                 huX:
37359 00003291 07                       pop     es
37360                                   ; 01/01/2023
37361                                   ;pop    ax
37362                                   ; 02/01/2023
37363 00003292 5A                       pop     dx ; (*)
37364 00003293 C3                       retn
37365
37366                                 ; 02/01/2023
37367                                 %if 0
37368
37369                                 ; ----------------------------------------------------------------------------
37370                                 ;*** isTiny - returns with ZF set if user didn't specify /S
37371                                 ; ----------------------------------------------------------------------------
37372                                 ; ENTRY:    None
37373                                 ; EXIT:     ZF set if user DIDN'T specify /S
37374                                 ; ERROR:    None
37375                                 ; USES:     Flags
37376                                 ; ----------------------------------------------------------------------------
37377
37378                                     ; 02/01/2023 - Retro DOS v4.2 (Modified MSDOS 6.21 IO.SYS)
37379                                 isTiny:
37380                                     ; 02/01/2023
37381                                     ;push   ax
37382
37383                                     ;getdata al,fUmbTiny
37384                                     ;
37385                                     ;push   ds
37386                                     ;push   cs
37387                                     ;pop    ds
37388                                     ;mov    al,[fUmbTiny]
37389                                     ;pop    ds
37390
37391                                     ; 09/09/2023
37392                                     ;mov    al,[cs:fUmbTiny]
37393                                     ; 02/01/2023
37394                                     ; ds = cs
37395                                     mov     al,[fUmbTiny]
37396
37397                                     or      al,al
37398                                     ; 02/01/2023
37399                                     ;pop    ax
37400                                     retn
37401
37402                                 %endif
37403
37404                                 ; ----------------------------------------------------------------------------
37405                                 ;*** isFreeMCB - returns with ZF set if current MCB (ES:0) is FREE
37406                                 ; ----------------------------------------------------------------------------
37407                                 ; ENTRY:    ES:0 should point to an MCB
37408                                 ; EXIT:     ZF set if MCB is free, else !ZF
37409                                 ; ERROR:    None
37410                                 ; USES:     Flags
37411                                 ; ----------------------------------------------------------------------------
37412
37413                                 ;isFreeMCB:
37414                                 ;   or      word [es:ARENA.OWNER],0
37415                                 ;   retn
37416
37417                                 ; ----------------------------------------------------------------------------
37418                                 ;*** hideMCB - marks as HIDDEN the MCB at ES:0
37419                                 ; ----------------------------------------------------------------------------
```

```
                                ; ENTRY:    ES:0 should point to an MCB
                                ; EXIT:     None; MCB marked as HIDDEN
                                ; ERROR:    None
                                ; USES:     None
                                ; ----------------------------------------------------------------------------

                                hideMCB:
00003294 26C70601000800             mov     word [es:ARENA.OWNER],SystemPSPOwner ; 8
0000329B 26C70608004849             mov     word [es:ARENA.NAME+0], 'HI' ; 4948h
000032A2 26C7060A004444             mov     word [es:ARENA.NAME+2], 'DD' ; 4444h
000032A9 26C7060C00454E             mov     word [es:ARENA.NAME+4], 'EN' ; 4E45h
000032B0 26C7060E002020             mov     word [es:ARENA.NAME+6], '  ' ; 2020h
000032B7 C3                         retn

                                ; ----------------------------------------------------------------------------
                                ;*** unHideMCB - marks as FREE the MCB at ES:0
                                ; ----------------------------------------------------------------------------
                                ; ENTRY:    ES:0 should point to an MCB
                                ; EXIT:     None; MCB marked as FREE
                                ; ERROR:    None
                                ; USES:     None
                                ; ----------------------------------------------------------------------------

                                    ; 03/01/2023 - Retro DOS v4.2 (Modified MSDOS 6.21 IO.SYS)

                                unHideMCB:
                                    ; 03/01/2023
                                    ;push   ax
000032B8 26C70601000000             mov     word [es:ARENA.OWNER],FreePSPOwner ; 0
000032BF B82020                     mov     ax,'  ' ; 2020h
000032C2 26A30800                   mov     [es:ARENA.NAME+0],ax
000032C6 26A30A00                   mov     [es:ARENA.NAME+2],ax
000032CA 26A30C00                   mov     [es:ARENA.NAME+4],ax
000032CE 26A30E00                   mov     [es:ARENA.NAME+6],ax
                                    ; 03/01/2023
                                    ;pop    ax
000032D2 C3                         retn

                                ; ----------------------------------------------------------------------------
                                ;*** findUMB - makes ES:0 point to the first MCB in UMB given as AL
                                ;           -- returns UmbHEAD pointer (0x9FFF) if passed AL==0
                                ; ----------------------------------------------------------------------------
                                ; ENTRY:    AL should be to a valid UMB number
                                ; EXIT:     ES:0 points to first MCB in UMB (_not_ the 8+SC MCB that heads it)
                                ; ERROR:    Carry set if couldn't reach UMB (too high)
                                ; USES:     Flags, ES
                                ; ----------------------------------------------------------------------------

                                    ; 01/01/2023 - Retro DOS v4.2 (Modified MSDOS 6.21 IO.SYS)
                                    ; (SYSINIT:3344h)
                                findUMB:
                                    ; 01/01/2023
                                    ;push   ax
                                    ; 02/01/2023
000032D3 51                         push    cx ; *
                                    ;push   dx

000032D4 30E4                       xor     ah,ah           ; Zap ah, so al==ax

000032D6 89C2                       mov     dx,ax           ; Store the to-be-found UMB number in DX

000032D8 E8D2FE                     call    UmbHead         ; Returns first UMB segment in AX
                                    ; 22/07/2023
                                    ;mov    es,ax ; *
000032DB 31C9                       xor     cx,cx           ; Pretend we're on UMB 0 for now...

                                    ; 22/07/2023
                                fu10:
000032DD 8EC0                       mov     es,ax ; * ; **
                                ; ------------------------------------------
                                ; FU10--CX - This UMB number; 0 == conventional
                                ;       DX - The UMB number they're looking for
                                ;       ES - The current MCB address
                                ; ------------------------------------------

                                ;fu10:
000032DF 39D1                       cmp     cx,dx           ; If CX==DX, we've found the UMB we're
000032E1 7417                       je      short fuX       ; searching for--so exit.

000032E3 E8D4FE                     call    isSysMCB        ; Returns with ZF set if owner is SYSTEM
000032E6 7501                       jnz     short fu20

000032E8 41                         inc     cx              ; If it _was_ SYSTEM, we're in a new UMB.
                                fu20:
                                    ;mov    al,[es:ARENA.SIGNATURE]
                                    ;cmp    al,arena_signature_end ; 'Z'
                                    ; 19/07/2023
000032E9 26803E00005A               cmp     byte [es:ARENA.SIGNATURE],arena_signature_end
000032EF 7408                       je      short fuE       ; 'Z' means this was the last MCB... that's it.

                                    ;NextMCB es,ax          ; Go on forward.
                                    ; 22/07/2023
                                    ; ax = es
                                    ;mov    ax,es ; * ; 22/07/2023
                                    ;add    ax,[es:3]
000032F1 2603060300                 add     ax,[es:ARENA.SIZE]
000032F6 40                         inc     ax
                                    ; 22/07/2023
                                    ;mov    es,ax ; **
000032F7 EBE4                       jmp     short fu10
                                fuE:
000032F9 F9                         stc
                                fuX:
                                    ; 01/01/2023
                                    ;pop    dx
                                    ; 02/01/2023
000032FA 59                         pop     cx ; *
                                    ;pop    ax              ; The address is already in ES.
000032FB C3                         retn

                                ; ----------------------------------------------------------------------------
                                ;*** BigFree - makes ES:0 point to the largest free MCB in UMB given as AL
                                ; ----------------------------------------------------------------------------
                                ; ENTRY:    AL should be to a valid UMB number
                                ; EXIT:     ES:0 points to largest free MCB in UMB, AX returns its size
                                ; ERROR:    Carry set if couldn't reach UMB (0 or too high)
                                ; USES:     Flags, ES
                                ; ----------------------------------------------------------------------------

                                    ; 01/01/2023 - Retro DOS v4.2
                                BigFree:
                                    ; 01/01/2023
                                    ;push   bx
000032FC 51                         push    cx
```

```
                                    call    findUMB                 ; Returns with CF if err, else ES==MCB
                                    jc      short bfX               ; (would be "jc bfE"; it just does stc)

                                    xor     bx,bx                   ; Segment address of largest free MCB
                                    xor     cx,cx                   ; Size of largest free MCB

                                ; ---------------------------------------------
                                ; BF10--ES - Current MCB address
                                ;       BX - Address of largest free MCB so far
                                ;       CX - Size of largest free MCB so far
                                ; ---------------------------------------------

                                bf10:
                                    call    isSysMCB                ; If we've left the MCB, we're done.
                                    jz      short bf30

                                    ;call   isFreeMCB               ; Returns with ZF set if owner is 0
                                    or      word [es:ARENA.OWNER],0
                                    jnz     short bf20

                                    mov     ax,[es:ARENA.SIZE]
                                    ;cmp    cx,[es:ARENA.SIZE]      ; Compare sizes...
                                    cmp     cx,ax
                                    ;jg     short bf20              ; Unless we're bigger,
                                    ; 19/07/2023
                                    jge     short bf20

                                    mov     bx,es                   ; Store this new element's address,
                                    ;mov    cx,[es:ARENA.SIZE]      ; and its size.
                                    mov     cx,ax

                                bf20:
                                    ;mov    al,[es:ARENA.SIGNATURE]
                                    ;cmp    al,arena_signature_end  ; 'Z'
                                    ; 19/07/2023
                                    ;cmp    byte [es:0],'Z'
                                    cmp     byte [es:ARENA.SIGNATURE],arena_signature_end
                                    jz      short bf30              ; 'Z' means this was the last MCB.

                                    ;NextMCB es,ax                 ; Go on forward.
                                    mov     ax,es
                                    ;add    ax,[es:3]
                                    add     ax,[es:ARENA.SIZE]
                                    inc     ax
                                    mov     es,ax

                                    jmp     short bf10

                                bf30:       mov     es,bx                   ; Return the address
                                    mov     ax,cx                   ; Return the size
                                    or      bx,bx
                                    jnz     short bfX               ; (if size==0, there's nothing free)
                                bfE:
                                    stc
                                bfX:
                                    pop     cx
                                    ; 01/01/2023
                                    ;pop    bx
                                    retn

                                ; ---------------------------------------------------------------------------
                                ;*** isSpecified - sets ZF if UMB in AL wasn't specified in DH/LH line.
                                ; ---------------------------------------------------------------------------
                                ; ENTRY:    AL should be to a valid UMB number
                                ; EXIT:     ZF set if UMB wasn't specified, ZF clear if it was
                                ; ERROR:    None
                                ; USES:     Flags
                                ; ---------------------------------------------------------------------------

                                    ; 02/01/2023 - Retro DOS v4.2

                                isSpecified:
                                    ; 02/01/2023
                                    ;push   ax

                                    xor     bh,bh
                                    mov     bl,al

                                    ;getdata al,DS:UmbUsed[bx]
                                    ;
                                    ;push   ds
                                    ;push   cs
                                    ;pop    ds
                                    ;mov    al,[bx+UmbUsed]
                                    ;pop    ds

                                    ;mov    al,[cs:bx+UmbUsed]
                                    ; 02/01/2023
                                    ; ds = cs
                                    mov     al,[bx+UmbUsed]

                                    or      al,al                   ; Sets ZF if al==0 (ie, if unspecified)

                                    ; 09/09/2023
                                    ; 02/01/2023
                                    ;pop    ax

                                    retn

                                ; ---------------------------------------------------------------------------
                                ;*** shrinkMCB - breaks an MCB into two pieces, the lowest one's size==AX
                                ; ---------------------------------------------------------------------------
                                ; ENTRY:    AX == new size, ES:0 == current MCB
                                ; EXIT:     None; MCB broken if carry clear
                                ; ERROR:    Carry set if MCB isn't as large as AX+0x20 (not a useful split)
                                ; USES:     Flags
                                ; ---------------------------------------------------------------------------
                                ; If the size of the to-be-split MCB isn't at least 0x20 bytes greater than
                                ; the specified new size, the split is useless; if it's only 0x10 bytes, that
                                ; 0x10 will be used to make a header that mentions a 0-byte free space, and
                                ; that just sucks up 0x10 bytes for nothing. So we make 0x20 bytes the
                                ; minimum for performing a split.
                                ; ---------------------------------------------------------------------------

                                MIN_SPLIT_SIZE      equ     20h

                                    ; 02/01/2023 - Retro DOS v4.2 (Modified MSDOS 6.21 IO.SYS)

                                shrinkMCB:
                                    ;pushreg <bx,cx,es>
                                    ; 02/01/2023
                                    ;push   bx
                                    push    cx
```

```
                                push    es
                                
0000334B 89C3                   mov     bx,ax                   ; Move things around... and
                                ; 02/01/2023
                                ;mov    ax,es                   ; save this one for later.
                                
0000334D 268B0E0300             mov     cx,[es:ARENA.SIZE]
                                ; 02/01/2023
00003352 89C8                   mov     ax,cx
                                
00003354 83E820                 sub     ax,MIN_SPLIT_SIZE ; 32
                                ;sub    cx,MIN_SPLIT_SIZE ; 32
                                ;;cmp   bx,cx                   ; {New size} vs {Current Size-20h}
                                ;ja     short smE               ; if wanted_size > cur-20h, abort.
                                ; 18/12/2022
                                ;cmp    cx,bx
                                ; 02/01/2023
00003357 39D8                   cmp     ax,bx
00003359 7228                   jb      short smE ; (*)
                                
0000335B 268A160000             mov     dl,[es:ARENA.SIGNATURE]
                                
                                ;mov    cx,[es:ARENA.SIZE]
                                ; 02/01/2023
00003360 8CC0                   mov     ax,es
                                
00003362 26891E0300             mov     [es:ARENA.SIZE],bx
00003367 26C60600004D           mov     byte [es:ARENA.SIGNATURE],'M'
                                
0000336D 01D8                   add     ax,bx
0000336F 40                     inc     ax
00003370 8EC0                   mov     es,ax                   ; Move to new arena area
                                
00003372 89C8                   mov     ax,cx
00003374 29D8                   sub     ax,bx
                                ; 12/12/2022
                                ; ax > 0
00003376 48                     dec     ax                      ; And prepare the new size
                                
                                ; 18/12/2022
00003377 2688160000             mov     [es:ARENA.SIGNATURE],dl
                                ;mov    word [es:ARENA.OWNER],0 ; (**)
0000337C 26A30300               mov     [es:ARENA.SIZE],ax
                                ;mov    ax,'  ' ; 2020h
                                ;mov    [es:ARENA.NAME+0],ax ; (**)
                                ;mov    [es:ARENA.NAME+2],ax ; (**)
                                ;mov    [es:ARENA.NAME+4],ax ; (**)
                                ;mov    [es:ARENA.NAME+6],ax ; (**)
                                
                                ; 18/12/2022
00003380 E8A801                 call    freeMCB ; (**)
                                
                                ; 12/12/2022
                                ; cf=0
                                ;clc
                                ; 18/12/2022
                                ;jmp    short smX
                            smE:
                                ; 18/12/2022
                                ; cf=1 (*)
                                ;stc
                            smX:
                                ;popreg <es,cx,bx>
00003383 07                     pop     es
00003384 59                     pop     cx
                                ; 02/01/2023
                                ;pop    bx
00003385 C3                     retn
                                
                            ; ---------------------------------------------------------------------------
                            ;*** hideUMB? - hides as appropriate the UMB in CL
                            ; ---------------------------------------------------------------------------
                            ; ENTRY:    CL should be to a valid UMB number, and AX to its address (findUMB)
                            ; EXIT:     None; UMB is hidden as necessary
                            ; ERROR:    None
                            ; USES:     Flags, AX, CX
                            ; ---------------------------------------------------------------------------
                            ; PRIMARY LOGIC:
                            ;
                            ; If the UMB is specified in the DH/LH statement, then:
                            ;    If the largest free segment is too small (check specified size), then:
                            ;       Pretend it wasn't ever specified, and fall out of this IF.
                            ;    Else, if largest free segment is LARGER than specified size, then:
                            ;       If /S was given on the command-line, then:
                            ;          Break that element into two pieces
                            ;          Set a flag that we're shrinking
                            ;       Endif
                            ;    Endif
                            ; Endif
                            ; If the UMB is NOT specified (or was removed by the above):
                            ;    Hide all free elements in the UMB
                            ;    If the flag that we're shrinking was set, then:
                            ;       UN-hide the lower portion of the shrunken UMB
                            ;    ENDIF
                            ; ENDIF
                            ; ---------------------------------------------------------------------------
                                
                                ; 01/01/2023 - Retro DOS v4.2 (Modified MSDOS 6.21 IO.SYS)
                                ; (SYSINIT:3426h)
                            _hideUMB_:
                                ; 02/01/2023
                                ; ds = cs
                                
                                ; 01/01/2023
                                ;push   bx
                                ;push   dx
00003386 06                     push    es
                                
00003387 88C8                   mov     al,cl
00003389 E8B2FF                 call    isSpecified     ; Returns ZF set if al's umb was NOT specified
0000338C 742D                   jz      short hu_20
                                
0000338E 88C8                   mov     al,cl           ; Retrieve the size of the largest
00003390 E869FF                 call    BigFree         ; free element in AX; put its address in ES
00003393 7226                   jc      short hu_20     ; Oops. Errors mean skip this part.
                                
00003395 50                     push    ax              ; TOS==size of BigFree in UMB (popped as BX)
00003396 88C8                   mov     al,cl           ; Retrieve the user's specified
00003398 E8AAFE                 call    GetSize         ; minimum size for this umb (into AX)
0000339B 5B                     pop     bx              ; Now BX==BigFree, AX==Specified Size
                                
0000339C 09C0                   or      ax,ax           ; If they didn't specify one,
0000339E 741B                   jz      short hu_20     ; Skip over all this.
                                
```

```
37792 000033A0 39D8                     cmp     ax,bx           ; Ah... if (specified > max free)
37793 000033A2 7607                     jbe     short hu_10
37794
37795 000033A4 88C8                     mov     al,cl           ;  Then mark that UMB as unused. Nya nya.
37796 000033A6 E81DFD                   call    unMarkUMB
37797 000033A9 EB10                     jmp     short hu_20
37798                                 hu_10:
37799                                   ;call   isTiny          ; Returns ZF clear if user specified /S
37800                                   ;jz     short hu_20
37801                                   ; 02/01/2023
37802                                 ;isTiny:
37803                                   ;mov    al,[fUmbTiny] ; ds = cs
37804                                   ;or     al,al
37805 000033AB 800E[FC23]00             or      byte [fUmbTiny],0
37806 000033B0 7409                     jz      short hu_20
37807
37808 000033B2 E894FF                   call    shrinkMCB       ; They specified /S, so shrink the MCB to AX
37809 000033B5 7204                     jc      short hu_20     ; Ah... if didn't shrink after all, skip this:
37810
37811 000033B7 8CC2                     mov     dx,es
37812 000033B9 EB09                     jmp     short hu_30     ; Skip the spec check.. we wanna hide this one.
37813
37814 000033BB 89C8                   hu_20:    mov     ax,cx
37815 000033BD E87EFF                   call    isSpecified     ; If they specified this UMB, we're done...
37816 000033C0 7510                     jnz     short hu_X      ; so leave.
37817
37818 000033C2 31D2                     xor     dx,dx
37819                                 hu_30:
37820 000033C4 88C8                     mov     al,cl
37821
37822 000033C6 E89DFE                   call    hideUMB         ; Hides everything in UMB #al
37823
37824 000033C9 09D2                     or      dx,dx           ; Did we shrink a UMB? If not, DX==0,
37825 000033CB 7405                     jz      short hu_X      ; So we should leave.
37826
37827 000033CD 8EC2                     mov     es,dx           ; Ah, but if it isn't, DX==the MCB's address;
37828 000033CF E8E6FE                   call    unHideMCB       ; Un-hides the lower portion of that MCB.
37829                                 hu_X:
37830 000033D2 07                       pop     es
37831                                   ; 01/01/2023
37832                                   ;pop    dx
37833                                   ;pop    bx
37834 000033D3 C3                       retn
37835
37836                                 ; ---------------------------------------------------------------------------
37837                                 ;*** UnFreeze - Marks FROZEN elements as FREE
37838                                 ; ---------------------------------------------------------------------------
37839                                 ; Entry:  None
37840                                 ; Exit:   None; all 8+FROZEN elements are marked as FREE, from any UMB.
37841                                 ; Error:  None
37842                                 ; Uses:   Flags
37843                                 ; ---------------------------------------------------------------------------
37844
37845                                   ; 03/01/2023 - Retro DOS v4.2 (Modified MSDOS 6.21 IO.SYS)
37846                                 UnFreeze:
37847                                   ; 03/01/2023
37848                                   ;push   ax
37849 000033D4 06                       push    es
37850
37851 000033D5 E8D5FD                   call    UmbHead         ; Returns with carry if err, else ES == MCB
37852 000033D8 721C                     jc      short ufX
37853
37854                                   ; 22/07/2023
37855                                 uf10:
37856 000033DA 8EC0                     mov     es,ax ; *
37857
37858                                 ; ------------------------------
37859                                 ; UF10--ES - Current MCB address
37860                                 ; ------------------------------
37861
37862                                 ;uf10:
37863 000033DC E81900                   call    isFrozMCB       ; Returns with ZF set if MCB is FROZEN
37864 000033DF 7505                     jnz     short uf20
37865 000033E1 E8D4FE                   call    unHideMCB
37866                                   ; 09/09/2023
37867                                   ; ax <> es
37868 000033E4 8CC0                     mov     ax,es ; *
37869                                 uf20:
37870                                   ;mov    al,[es:ARENA.SIGNATURE]
37871                                   ;cmp    al,arena_signature_end ; 'Z'
37872                                   ; 22/07/2023
37873 000033E6 26803E00005A             cmp     byte [es:ARENA.SIGNATURE],arena_signature_end ; 'Z'
37874 000033EC 7408                     jz      short ufX       ; 'Z' means this was the last MCB.. that's it.
37875
37876                                   ;NextMCB es,ax          ; Go on forward.
37877                                   ; 22/07/2023
37878                                   ; ax = es
37879                                   ;mov    ax,es ; *
37880                                   ;add    ax,[es:3]
37881 000033EE 2603060300              add     ax,[es:ARENA.SIZE]
37882 000033F3 40                       inc     ax
37883                                   ; 22/07/2023
37884                                   ;mov    es,ax
37885 000033F4 EBE4                     jmp     short uf10
37886                                 ufX:
37887 000033F6 07                       pop     es
37888                                   ; 03/01/2023
37889                                   ;pop    ax
37890 000033F7 C3                       retn
37891
37892                                 ; ---------------------------------------------------------------------------
37893                                 ;*** isFrozMCB - returns with ZF set if current MCB (ES:0) is FROZEN
37894                                 ; ---------------------------------------------------------------------------
37895                                 ; ENTRY:    ES:0 should point to an MCB
37896                                 ; EXIT:     ZF set if MCB is frozen, else !ZF
37897                                 ; ERROR:    None
37898                                 ; USES:     Flags
37899                                 ; ---------------------------------------------------------------------------
37900
37901                                 isFrozMCB:
37902                                   ;push   ax
37903
37904                                   ;mov    ax,[es:ARENA.OWNER]   ; Check the owner...
37905                                   ;cmp    ax,SystemPSPOwner     ; 8 (for US OR Japan) is valid
37906 000033F8 26833E010008             cmp     word [es:ARENA.OWNER],SystemPSPOwner
37907 000033FE 7522                     jne     short ifmX
37908
37909                                   ;mov    ax,[es:ARENA.NAME+0]
37910                                   ;cmp    ax,'FR' ; 5246h
37911 00003400 26813E08004652           cmp     word [es:ARENA.NAME+0],'FR'
37912 00003407 7519                     jne     short ifmX
37913                                   ;mov    ax,[es:ARENA.NAME+2]
37914                                   ;cmp    ax,'OZ' ; 5A4Fh
37915 00003409 26813E0A004F5A           cmp     word [es:ARENA.NAME+2],'OZ'
```

```
37916 00003410 7510                     jne     short ifmX
37917                                   ;mov    ax,[es:ARENA.NAME+4]
37918                                   ;cmp    ax,'EN' ; 4E45h
37919 00003412 26813E0C00454E           cmp     word [es:ARENA.NAME+4],'EN'
37920 00003419 7507                     jne     short ifmX
37921                                   ;mov    ax,[es:ARENA.NAME+6]
37922                                   ;cmp    ax,'  ' ; 2020h
37923 0000341B 26813E0E002020           cmp     word [es:ARENA.NAME+6],'  '
37924                                 ifmX:
37925                                   ;pop    ax
37926 00003422 C3                       retn
37927
37928                                 ; ------------------------------------------------------------------------------
37929                                 ;*** frezMCB - marks as 8+FROZEN the MCB at ES:0
37930                                 ; ------------------------------------------------------------------------------
37931                                 ; ENTRY:    ES:0 should point to an MCB
37932                                 ; EXIT:     None; MCB frozen
37933                                 ; ERROR:    None
37934                                 ; USES:     None
37935                                 ; ------------------------------------------------------------------------------
37936
37937                                 frezMCB:
37938 00003423 26C70601000800           mov     word [es:ARENA.OWNER],SystemPSPOwner ; 8
37939 0000342A 26C70608004652           mov     word [es:ARENA.NAME+0],'FR'
37940 00003431 26C7060A004F5A           mov     word [es:ARENA.NAME+2],'OZ'
37941 00003438 26C7060C00454E           mov     word [es:ARENA.NAME+4],'EN'
37942 0000343F 26C7060E002020           mov     word [es:ARENA.NAME+6],'  '
37943 00003446 C3                       retn
37944
37945                                 ; ------------------------------------------------------------------------------
37946                                 ;*** FreezeUM - Marks FROZEN all UM elements now FREE, save those in load UMB
37947                                 ; ------------------------------------------------------------------------------
37948                                 ; Entry:  None
37949                                 ; Exit:   None; all free elements not in load UMB marked as 8+FROZEN
37950                                 ; Error:  None
37951                                 ; Uses:   Flags
37952                                 ; ------------------------------------------------------------------------------
37953
37954                                   ; 01/01/2023 - Retro DOS v4.2
37955                                 FreezeUM:
37956                                   ; 01/01/2023
37957                                   ;push   ax
37958                                   ;push   cx
37959                                   ;push   dx
37960 00003447 06                       push    es
37961
37962                                   ;;call  GetLoadUMB
37963                                   ; 01/01/2023
37964                                   ; ds = cs
37965                                   ;mov    al,[cs:UmbLoad] ; 19/04/2019 - Retro DOS v4.0
37966 00003448 A0[FF23]                 mov     al,[UmbLoad]
37967
37968 0000344B 30E4                     xor     ah,ah           ; Zap ah, so al==ax
37969 0000344D 89C2                     mov     dx,ax           ; Store the load UMB in DX, so we can skip it
37970
37971 0000344F E85BFD                   call    UmbHead         ; Returns first UMB segment in AX
37972                                   ; 22/07/2023
37973                                   ;mov    es,ax ; *
37974 00003452 31C9                     xor     cx,cx           ; Pretend we're on UMB 0 for now...
37975
37976                                   ; 22/07/2023
37977                                 fum10:
37978 00003454 8EC0                     mov     es,ax ; *
37979
37980                                 ; ----------------------------------------
37981                                 ; FUM10--ES - Current MCB address
37982                                 ;        CX - Current UMB number
37983                                 ;        DX - UMB number to skip (load UMB)
37984                                 ; ----------------------------------------
37985
37986                                 ;fum10:
37987 00003456 E861FD                   call    isSysMCB        ; Returns with ZF set if owner is SYSTEM
37988 00003459 7501                     jnz     short fum20
37989
37990 0000345B 41                       inc     cx              ; If it _was_ SYSTEM, we're in a new UMB.
37991                                 fum20:
37992 0000345C 39D1                     cmp     cx,dx           ; If this is the load UMB, we don't want to
37993 0000345E 740B                     je      short fum30     ; freeze anything... so skip that section.
37994
37995                                   ;call   isFreeMCB       ; Oh. If it's not free, we can't freeze it
37996 00003460 26830E010000             or      word [es:ARENA.OWNER],0
37997 00003466 7503                     jnz     short fum30     ; either.
37998
37999 00003468 E8B8FF                   call    frezMCB
38000                                 fum30:
38001                                   ;mov    al,[es:ARENA.SIGNATURE]
38002                                   ;cmp    al,arena_signature_end ; 'Z'
38003                                   ; 22/07/2023
38004 0000346B 26803E00005A             cmp     byte [es:ARENA.SIGNATURE],arena_signature_end ; 'Z'
38005 00003471 7408                     je      short fumX      ; 'Z' means this was the last MCB.. that's it.
38006
38007                                   ;NextMCB es, ax               ; Go on forward.
38008                                   ; 22/07/2023
38009                                   ; ax = es
38010                                   ;mov    ax,es
38011                                   ;add    ax,[es:3]
38012 00003473 2603060300               add     ax,[es:ARENA.SIZE]
38013 00003478 40                       inc     ax
38014                                   ; 22/07/2023
38015                                   ;mov    es,ax ; *
38016 00003479 EBD9                     jmp     short fum10
38017
38018 0000347B 07                     fumX:     pop     es
38019                                   ; 01/01/2023
38020                                   ;pop    dx
38021                                   ;pop    cx
38022                                   ;pop    ax
38023 0000347C C3                       retn
38024
38025                                 ; ------------------------------------------------------------------------------
38026                                 ;*** UmbTest - returns with carry set if UMBs are not available, else CF==false
38027                                 ; ------------------------------------------------------------------------------
38028                                 ; ENTRY:    None
38029                                 ; EXIT:     Carry is clear if UMBs are available, or set if they are not
38030                                 ; ERROR:    None
38031                                 ; USES:     CF (AX,BX,DS,ES pushed 'cause they're used by others)
38032                                 ; ------------------------------------------------------------------------------
38033
38034                                   ; 01/01/2023 - Retro DOS v4.2 (Modified MSDOS 6.21 IO.SYS)
38035                                 UmbTest:
38036                                   ; 01/01/2023
38037                                   ;push   ax
38038 0000347D 53                       push    bx ; *
38039                                   ;push   ds
```

```
0000347E 06                             push    es ; **
                                        ; 01/01/2023
                                        ; ds = cs
0000347F E871FB                         call    fm_link                 ; Link in UMBs (if not already linked)
00003482 E80800                         call    WalkMem                 ; Check to see if they're really linked
00003485 9C                             pushf                           ; And remember what we found out
00003486 E87BFB                         call    fm_unlink               ; Unlink UMBs (if WE have linked 'em)
00003489 9D                             popf                            ; And restore what we found out.
0000348A 07                             pop     es ; **
                                        ; 01/01/2023
                                        ;pop    ds
0000348B 5B                             pop     bx ; *
                                        ;pop    ax
0000348C C3                             retn
                                ; ----------------------------------------------------------------------------
                                ;*** WalkMem - travels memory chain and returns carry clear iff UMBs are linked
                                ; ----------------------------------------------------------------------------
                                ; ENTRY:    None
                                ; EXIT:     Carry SET if MCB chain stops before 9FFF, CLEAR if stops >= 9FFF.
                                ; ERROR:    None
                                ; USES:     Flags
                                ; ----------------------------------------------------------------------------
                                        ; 01/01/2023 - Retro DOS v4.2 (Modified MSDOS 6.21 IO.SYS)
                                        ; (SYSINIT:3541h)
                                WalkMem:
                                        ;push   ax ; ?
                                        ;push   bx ; ?
                                        ;;push  ds ; ? ; 01/01/2023 (MSDOS 6.21 IO.SYS, SYSINIT:352Fh)
                                        ;push   es ; ? no need to save contents of these registers ?
0000348D B452                           mov     ah,GET_IN_VARS          ; Call int 21h, function 52h...
0000348F CD21                           int     21h
00003491 268B47FE                       mov     ax,[es:bx-2]
                                        ; 22/07/2023
                                UM10:
00003495 8EC0                           mov     es,ax ; * ; **
                                ; ------------------------------
                                ; UM10: ES = Current MCB pointer
                                ; ------------------------------
                                ;um10:
                                        ;mov    al,[es:ARENA.SIGNATURE]
                                        ;cmp    al,arena_signature_end ; 'Z'
                                        ; 22/07/2023
00003497 26803E00005A                   cmp     byte [es:ARENA.SIGNATURE],arena_signature_end ; 'Z'
0000349D 7408                           je      short um20              ; If signature == 'Z', hay no more.
                                        ;NextMCB es,bx                  ; Move to the next MCB
                                        ;mov    bx,es
                                        ;;add   bx,[es:3]
                                        ;add    bx,[es:ARENA.SIZE]
                                        ;inc    bx
                                        ;mov    es,bx
                                        ; 22/07/2023
                                        ; ax = es
                                        ;mov    ax,es ; *
0000349F 2603060300                     add     ax,[es:ARENA.SIZE]
000034A4 40                             inc     ax
                                        ;mov    es,ax ; **
000034A5 EBEE                           jmp     short um10              ; And restart the loop.
                                um20:
                                        ; 22/07/2023
                                        ; ax = es
                                        ;mov    ax,es
000034A7 3DFF9F                         cmp     ax,9FFFh                ; This sets CF if ax < 9FFF.
                                        ;pop    es ; ?
                                        ;;pop   ds ; ? ; 01/01/2023 (MSDOS 6.21 IO.SYS, SYSINIT:353Dh)
                                        ;pop    bx ; ?
                                        ;pop    ax ; ?
000034AA C3                             retn
                                ; ----------------------------------------------------------------------------
                                ;*** hl_unlink - unlinks UMBs if fm_umb is set to 0; restores strategy too
                                ; ----------------------------------------------------------------------------
                                ; ENTRY:    fm_umb == 1 : leave linked, else unlink
                                ; EXIT:     None
                                ; ERROR:    None
                                ; USES:     AX, BX
                                ; ----------------------------------------------------------------------------
                                        ; 01/01/2023 - Retro DOS v4.2
                                hl_unlink:
000034AB 30FF                           xor     bh,bh
                                        ;getdata bl,fm_umb              ; Restore original link-state
                                        ;
                                        ;push   ds
                                        ;push   cs
                                        ;pop    ds
                                        ;mov    bl,[fm_umb]
                                        ;pop    ds
                                        ; 01/01/2023
                                        ; ds = cs
                                        ;mov    bl,[cs:fm_umb]
000034AD 8A1E[3024]                     mov     bl,[fm_umb]
000034B1 B80358                         mov     ax,DOS_SET_UMBLINK ; 5803h
000034B4 CD21                           int     21h
000034B6 C3                             retn
                                ; ----------------------------------------------------------------------------
                                ; HIGHEXIT.INC (MSDOS 6.0 - 1991)
                                ; ----------------------------------------------------------------------------
                                ; 09/04/2019 - Retro DOS v4.0
                                ;   Module:   HIGHEXIT.INC - Code executed after LoadHigh or DeviceHigh
                                ;   Date:     May 14, 1992
                                ;   Modification log:
                                ;
```

```
38164                                 ;     DATE    WHO      DESCRIPTION
38165                                 ;   --------  -------  -------------------------------------------------------
38166                                 ;   05/14/92  t-richj  Original
38167                                 ;   06/21/92  t-richj  Final revisions before check-in
38168                                 
38169                                 UMB_HeadIdx equ     8Ch     ; Offset from ES (after func52h) to get UMBHead
38170                                 
38171                                 ; --------------------------------------------------------------------------
38172                                 ;*** UnHideUMBs - Marks HIDDEN elements as FREE
38173                                 ; --------------------------------------------------------------------------
38174                                 ; ENTRY:  None; perhaps, earlier, HideUMBs was called... if not, we have
38175                                 ;             very little to do, as no elelments will be marked as HIDDEN.
38176                                 ; EXIT:   Sets InHigh to zero; carry clear if HideUMBs was called earlier.
38177                                 ; ERROR:  None
38178                                 ; USES:   fInHigh (from highvar.inc), carry flag
38179                                 ; --------------------------------------------------------------------------
38180                                 
38181                                     ; 31/12/2022 - Retro DOS v4.2 (Modified MSDOS 6.21 IO.SYS)
38182                                     ; (SYSINIT:357Bh)
38183                                 
38184                                 UnHideUMBs:
38185 000034B7 50                       push    ax              ; Save ax for what we're about to do
38186                                 
38187                                 ; --------------------------------------------------------------------------
38188                                 ; BUGBUG t-richj 11-8-92: The following six lines were commented out for a good
38189                                 ;    length of time. Those six constitute a check of whether or not we should
38190                                 ;    indeed clean up the upper-memory chain; without such a check, COMMAND.COM
38191                                 ;    will destroy the current link-state and memory-allocation strategy after
38192                                 ;    every command execution.
38193                                 ; --------------------------------------------------------------------------
38194                                 
38195                                     ;getdata al,fInHigh   ; Get InHigh from data segment
38196                                     ;
38197                                     ;push  ds
38198                                     ;push  cs
38199                                     ;pop   ds
38200                                     ;mov   al,[fInHigh]
38201                                     ;pop   ds
38202                                 
38203                                     ;mov   al,[cs:fInHigh]
38204                                     ; 31/12/2022
38205                                     ; ds = cs
38206 000034B8 A0[FB23]                  mov     al,[fInHigh]
38207                                 
38208 000034BB 08C0                      or      al,al
38209 000034BD 7503                      jnz     short uhu10    ; If didn't call loadhigh/devicehigh earlier,
38210                                 
38211 000034BF 58                        pop     ax             ; then there's nothing to do here... so
38212 000034C0 F9                        stc                    ; restore everything and return. Just like
38213 000034C1 C3                        retn                   ; that.
38214                                 uhu10:
38215 000034C2 E88E00                    call    linkumb        ; Make sure UMBs are linked in.
38216 000034C5 E81200                    call    FreeUMBs
38217                                 
38218                                     ;putdata fInHigh,0    ; We're leaving, so update fInHigh.
38219                                     ;
38220                                     ;push  es
38221                                     ;push  cs
38222                                     ;pop   es
38223                                     ;mov   byte [es:fInHigh],0
38224                                     ;pop   ds
38225                                 
38226                                     ; 31/12/2022
38227                                     ; ds = cs
38228                                     ;mov   byte [cs:fInHigh],0
38229 000034C8 C606[FB23]00              mov     byte [fInHigh],0
38230                                 
38231                                     ;call  he_unlink      ; Unlink UMBs
38232                                     ; 31/12/2022
38233                                 ;;he_unlink:
38234 000034CD 30FF                      xor     bh,bh
38235                                 
38236                                     ;getdata bl,fm_umb    ; Restore original link-state
38237                                     ;mov   bl,[cs:fm_umb]
38238 000034CF 8A1E[3024]                mov     bl,[fm_umb]
38239                                 
38240 000034D3 B80358                    mov     ax,DOS_SET_UMBLINK ; 5803h
38241 000034D6 CD21                      int     21h
38242                                     ;;retn
38243                                 
38244 000034D8 58                        pop     ax
38245                                     ; 12/12/2022
38246                                     ;clc   ; 12/12/2022 (this clc may not be necessary!?)
38247 000034D9 C3                        retn
38248                                 
38249                                 ; 31/12/2022
38250                                 ;%if 0
38251                                 ;
38252                                 ;; --------------------------------------------------------------------------
38253                                 ;;*** he_unlink - unlinks UMBs if fm_umb is set to 0
38254                                 ;; --------------------------------------------------------------------------
38255                                 ;; ENTRY:    fm_umb == 1 : leave linked, else unlink
38256                                 ;; EXIT:     None
38257                                 ;; ERROR:    None
38258                                 ;; USES:     AX, BX
38259                                 ;; --------------------------------------------------------------------------
38260                                 ;
38261                                 ;he_unlink:
38262                                 ;   xor     bh, bh
38263                                 ;
38264                                 ;   ;getdata bl, fm_umb   ; Restore original link-state
38265                                 ;   mov     bl,[cs:fm_umb]
38266                                 ;
38267                                 ;   mov     ax,DOS_SET_UMBLINK ; 5803h
38268                                 ;   int     21h
38269                                 ;   retn
38270                                 ;
38271                                 ;%endif
38272                                 
38273                                 ; --------------------------------------------------------------------------
38274                                 ;*** freeUMBs - frees all HIDDEN memory elements in upper-memory.
38275                                 ; --------------------------------------------------------------------------
38276                                 ; ENTRY:    None
38277                                 ; EXIT:     None; HIDDEN memory elements returned to FREE
38278                                 ; ERROR:    None (ignore CF)
38279                                 ; USES:     Flags
38280                                 ; --------------------------------------------------------------------------
38281                                 
38282                                 FreeUMBs:
38283 000034DA 50                       push    ax
38284 000034DB 06                       push    es
38285                                 
38286 000034DC E86700                    call    HeadUmb        ; Returns with carry if err, else ES == MCB
38287 000034DF 721C                      jc      short fusX
```

```
                                fus10:
000034E1 8EC0                       mov     es,ax           ; Prepare for the loop; ES = current MCB addr.
                                ;fus10:
000034E3 E81A00                     call    isHideMCB       ; Returns with ZF set if owner is 0
000034E6 7505                       jnz     short fus20
000034E8 E84000                     call    freeMCB
                                    ; 09/09/2023
                                    ; ax <> es
000034EB 8CC0                       mov     ax,es
                                fus20:
                                    ;mov    al,[es:ARENA.SIGNATURE]
                                    ;cmp    al,arena_signature_end ; 'Z'
                                    ; 22/07/2023
000034ED 26803E00005A               cmp     byte [es:ARENA.SIGNATURE],arena_signature_end ; 'Z'
000034F3 7408                       jz      short fusX      ; That means this was the last MCB--that's it.

                                    ; 22/07/2023
                                    ; ax = es
                                    ;mov    ax,es
000034F5 2603060300                 add     ax,[es:ARENA.SIZE]
000034FA 40                         inc     ax
                                    ; 22/07/2023
                                    ;mov    es,ax
000034FB EBE4                       jmp     short fus10     ; Go on forward.
                                fusX:
000034FD 07                         pop     es
000034FE 58                         pop     ax
000034FF C3                         retn

                                ; ---------------------------------------------------------------------------
                                ;*** isHideMCB - returns with ZF set if current MCB (ES:0) is HIDDEN
                                ; ---------------------------------------------------------------------------
                                ; ENTRY:    ES:0 should point to an MCB
                                ; EXIT:     ZF set if MCB is hidden, else !ZF
                                ; ERROR:    None
                                ; USES:     Flags
                                ; ---------------------------------------------------------------------------

                                isHideMCB:
                                    ;push   ax

00003500 26833E010008               cmp     word [es:ARENA.OWNER],SystemPSPOwner ; If the owner's SYSTEM
00003506 7522                       jne     short ihm_x                          ; then check for HIDDEN

                                    ;mov    ax,[es:ARENA.NAME]
                                    ;cmp    ax,'HI' ; 4948h
00003508 26813E08004849             cmp     word [es:ARENA.NAME+0],'HI'
0000350F 7519                       jne     short ihm_x
                                    ;mov    ax,[es:ARENA.NAME+2]
                                    ;cmp    ax,'DD' ; 4444h
00003511 26813E0A004444             cmp     word [es:ARENA.NAME+2],'DD'
00003518 7510                       jne     short ihm_x
                                    ;mov    ax,[es:ARENA.NAME+4]
                                    ;cmp    ax,'EN' ; 4E45h
0000351A 26813E0C00454E             cmp     word [es:ARENA.NAME+4],'EN'
00003521 7507                       jne     short ihm_x
                                    ;mov    ax,[es:ARENA.NAME+6]
                                    ;cmp    ax,'  ' ; 2020h
00003523 26813E0E002020             cmp     word [es:ARENA.NAME+6],'  '
                                ihm_x:
                                    ;pop    ax
0000352A C3                         retn

                                ; ---------------------------------------------------------------------------
                                ;*** freeMCB - marks as free the MCB at ES:0
                                ; ---------------------------------------------------------------------------
                                ; ENTRY:    ES:0 should point to an MCB
                                ; EXIT:     None; MCB free'd
                                ; ERROR:    None
                                ; USES:     AX
                                ; ---------------------------------------------------------------------------

                                freeMCB:
0000352B 26C70601000000             mov     word [es:ARENA.OWNER],0
00003532 B82020                     mov     ax,'  ' ; mov ax,2020h ; 31/12/2022
00003535 26A30800                   mov     [es:ARENA.NAME+0],ax
00003539 26A30A00                   mov     [es:ARENA.NAME+2],ax
0000353D 26A30C00                   mov     [es:ARENA.NAME+4],ax
00003541 26A30E00                   mov     [es:ARENA.NAME+6],ax
00003545 C3                         retn

                                ; ---------------------------------------------------------------------------
                                ;*** HeadUmb - returns in AX the address of the first UMB block (0x9FFF)
                                ; ---------------------------------------------------------------------------
                                ; ENTRY:  Nothing
                                ; EXIT:   AX contains 0x9FFF for most systems
                                ; ERROR:  Carry set if pointer is 0xFFFF (if not set up yet--DH runs into this)
                                ; USES:   Flags, AX
                                ; ---------------------------------------------------------------------------

                                HeadUmb:
                                    ; 13/05/2019

                                    ;push   si ; ?
                                    ;push   ds ; ?
                                    ;push   es
                                    ;push   bx ; *

                                    ; 09/04/2019
                                    ; !!! No need to save es,bx,ds,si above !!! (es,bx are changed here)

00003546 B452                       mov     ah,GET_IN_VARS      ; Call int 21h, function 52h...
00003548 CD21                       int     21h
                                                ; DOS - 2+ internal - GET LIST OF LISTS
                                                ; Return: ES:BX -> DOS list of lists
                                    ;mov    ax,[es:8Ch]
0000354A 26A18C00                   mov     ax,[es:UMB_HeadIdx] ; And read what's in ES:008C
0000354E 83F8FF                     cmp     ax,0FFFFh
                                    ;je     short xhu_e         ; If it's 0xFFFF, it's an error...

                                    ;clc                        ; Else, it isn't.
                                    ;jmp    short xhu_x
                                xhu_e:
                                    ;stc
00003551 F5                         cmc     ; 09/04/2019 - Retro DOS v4.0 ; *
                                xhu_x:
                                    ;pop    bx ; *
                                    ;pop    es
                                    ;pop    ds ; ?
                                    ;pop    si ; ?
00003552 C3                         retn

                                ; ---------------------------------------------------------------------------
                                ;*** linkumb - links UMBs not already linked in; updates fm_umb as needed
```

```
                                ; ----------------------------------------------------------------------------
                                ; ENTRY:    None
                                ; EXIT:     fm_umb == 0 if not linked in previously, 1 if already linked in
                                ; ERROR:    None
                                ; USES:     AX, BX, fm_umb
                                ; ----------------------------------------------------------------------------

                                linkumb:
00003553 B80258                     mov     ax,DOS_GET_UMBLINK ; 5802h
00003556 CD21                       int     21h                     ; Current link-state is now in al

00003558 08C0                       or      al,al                   ; BUGBUG: proper check?
0000355A 7508                       jnz     short lumbX             ; Jumps if UMBs already linked in

0000355C B80358                     mov     ax,DOS_SET_UMBLINK ; 5803h
0000355F BB0100                     mov     bx,1
00003562 CD21                       int     21h
                                lumbX:
00003564 C3                         retn

                                ;%endif

                                ; 01/11/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 IO.SYS)
                                ; (SYSINIT:2B5Fh)

                                ; ----------------------------------------------------------------------
                                ; SYSCONF.ASM (MSDOS 6.0 - 1991)
                                ; ----------------------------------------------------------------------
                                ; 09/04/2019 - Retro DOS v4.0

                                ;--------------------------------------------------------------------------
                                ;
                                ; procedure : InitDevLoad
                                ;
                                ;   Input : DeviceHi = 0 indicates load DD in low memory
                                ;                    = 1 indicates load in UMB:
                                ;                       ConvLoad = 0 indicates a new-style load (see below)
                                ;                                = 1 indicates a DOS 5-style load
                                ;           DevSize  = Size of the device driver file in paras
                                ;
                                ;   Output : none
                                ;
                                ;   Initializes DevLoadAddr, DevLoadEnd & DevEntry.
                                ;   Also sets up a header for the Device driver entry for mem utility
                                ;
                                ;--------------------------------------------------------------------------
                                ; For a "new-style load", we break off the current DevEntry and link the umbs
                                ; as we see fit, using HideUMBs (and UnHideUMBs at exit, though _it_ decides
                                ; whether it's entitled to do anything). HideUMBs uses the chart built by
                                ; ParseVar to determine which UMBs to leave FREE, and which not.
                                ;--------------------------------------------------------------------------

                                    ; 01/11/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 IO.SYS)
                                    ; 01/01/2023 - Retro DOS v4.2 (Modified MSDOS 6.21 IO.SYS)
                                    ; (SYSINIT:364Ah)
                                InitDevLoad:
                                    ; 01/01/2023
                                    ;push   es ; *

                                    ; 11/12/2022
                                    ; ds = cs
00003565 803E[5124]00               cmp     byte [DeviceHi],0
                                    ;cmp    byte [cs:DeviceHi],0  ; Are we loading in UMB ?
                                    ;je     short InitForLo               ; no, init for lo mem
0000356A 7439                       je      short initforlo_x ; 09/04/2019

                                ; 01/01/2023 - Retro DOS v4.2 (Modified MSDOS 6.21 IO.SYS)
                                ; 01/11/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
                                ; %if 0
                                    ; 01/01/2023
0000356C 803E[4124]01               cmp     byte [ConvLoad],1     ; Are we loading as per DOS 5?
                                    ;cmp    byte [cs:ConvLoad],1  ; Are we loading as per DOS 5?
00003571 7413                       je      short InitForConv

                                ; There are two stages to preparing upper-memory; first, we mark as 8+HIDDEN
                                ; any areas not specified on the /L:... chain. Second, we mark as 8+FROZEN
                                ; any areas left in upper-memory, except for elements in the load UMB...
                                ; we then malloc space as per Dos-5 style, and mark as free any spaces which
                                ; are 8+FROZEN (but leave 8+HIDDEN still hidden). The load is performed,
                                ; and UnHideUMBs later on marks all 8+HIDDEN as free.

00003573 E85904                     call    ShrinkUMB             ; Stop using the old device arena

00003576 E892FC                     call    HideUMBs              ; Mark up the UM area as we see fit
00003579 E8CBFE                     call    FreezeUM              ; Hide everything BUT the load area
0000357C E85700                     call    GetUMBForDev          ; And grab that load area as needed
0000357F 9C                         pushf
00003580 E851FE                     call    UnFreeze              ; Then unhide everything frozen
00003583 9D                         popf
                                    ;jc     short InitForLo               ; (if carry, it's loading low)
                                    ;jmp    short InitForHi
                                    ; 06/07/2023 - Retro DOS v4.2 IO.SYS (optimization)
00003584 EB0B                       jmp     short idl0

                                ;%endif ; 01/11/2022

                                    ; 01/11/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 IO.SYS)
                                    ; (SYSINIT:2B67h)
                                InitForConv:
                                    ; 11/12/2022
                                    ; ds = cs
00003586 E83700                     call    SpaceInUMB            ; Do we have space left in the
                                                                  ;  current UMB ?
00003589 7308                       jnc     short InitForHi               ; yes, we have
0000358B E84104                     call    ShrinkUMB             ; shrink the current UMB in use
0000358E E84500                     call    GetUMBForDev          ; else try to allocate new UMB
                                idl0: ; 06/07/2023
00003591 720D                       jc      short InitForLo               ; we didn't succeed, so load
                                                                  ;  in low memory
                                InitForHi:
                                    ; 11/12/2022
                                    ; ds = cs
                                    ;mov    ax,[cs:DevUMBFree]    ; get Para addr of free mem
                                    ;mov    dx,[cs:DevUMBAddr]    ; UMB start addr
                                    ;add    dx,[cs:DevUMBSize]    ; DX = UMB End addr
00003593 A1[4724]                   mov     ax,[DevUMBFree]
00003596 8B16[4324]                 mov     dx,[DevUMBAddr]
0000359A 0316[4524]                 add     dx,[DevUMBSize]
0000359E EB0C                       jmp     short idl1

                                InitForLo:
                                    ; 11/12/2022
                                    ; ds = cs
                                    ;mov    byte [cs:DeviceHi],0  ; in case we failed to load
```

```
38536 000035A0 C606[5124]00             mov     byte [DeviceHi],0
38537                                   initforlo_x:
38538                                       ; 11/12/2022
38539                                       ; ds = cs
38540                                                                   ;  into UMB indicate that
38541                                                                   ;  we are loading low
38542                                       ;mov    ax,[cs:memhi]       ; AX = start of Low memory
38543                                       ;mov    dx,[cs:ALLOCLIM]    ; DX = End of Low memory
38544 000035A5 A1[6403]                    mov     ax,[memhi]
38545 000035A8 8B16[A502]                  mov     dx,[ALLOCLIM]
38546                                   idl1:
38547 000035AC E86600                      call    DevSetMark          ; setup a sub-arena for DD
38548                                       ; 11/12/2022
38549                                       ; ds = cs
38550                                       ;mov    [cs:DevLoadAddr],ax ; init the Device load address
38551                                       ;mov    [cs:DevLoadEnd],dx  ; init the limit of the block
38552                                       ;mov    word [cs:DevEntry],0  ; init Entry point to DD
38553                                       ;mov    [cs:DevEntry+2],ax
38554 000035AF A3[3524]                    mov     [DevLoadAddr],ax
38555 000035B2 8916[3724]                  mov     [DevLoadEnd],dx
38556 000035B6 C706[3924]0000              mov     word [DevEntry],0
38557 000035BC A3[3B24]                    mov     [DevEntry+2],ax
38558                                       ; 01/01/2023
38559                                       ;pop    es ; *
38560 000035BF C3                          retn
38561
38562                                   ;--------------------------------------------------------------------------
38563                                   ;
38564                                   ; procedure : SpaceInUMB?
38565                                   ;
38566                                   ;   Input : DevUMBAddr, DevUMBSize, DevUMBFree & DevSize
38567                                   ;   Output : Carry set if no space in UMB
38568                                   ;            Carry clear if Space is available for the device in
38569                                   ;              current UMB
38570                                   ;
38571                                   ;--------------------------------------------------------------------------
38572
38573                                   SpaceInUMB:
38574                                       ; 11/12/2022
38575                                       ; ds = cs
38576                                       ;mov    ax,[cs:DevUMBSize]
38577                                       ;add    ax,[cs:DevUMBAddr]  ; End of UMB
38578                                       ;sub    ax,[cs:DevUMBFree]  ; - Free = Remaining space
38579 000035C0 A1[4524]                    mov     ax,[DevUMBSize]
38580 000035C3 0306[4324]                  add     ax,[DevUMBAddr]             ; End of UMB
38581 000035C7 2B06[4724]                  sub     ax,[DevUMBFree]             ; - Free = Remaining space
38582                                       ; 11/12/2022
38583                                       ;or     ax,ax               ; Nospace ?
38584                                       ;jnz    short spcinumb1
38585                                       ;stc
38586                                       ;retn
38587                                       ; 11/12/2022
38588 000035CB 83F801                      cmp     ax,1
38589 000035CE 7205                        jb      short spcinumb2     ; cf=1
38590                                   spcinumb1:
38591 000035D0 48                          dec     ax                  ; space for sub-arena
38592                                       ; 11/12/2022
38593                                       ; ds = cs
38594 000035D1 3B06[3324]                  cmp     ax,[DevSize]
38595                                       ;cmp    ax,[cs:DevSize]             ; do we have space ?
38596                                   spcinumb2:
38597 000035D5 C3                          retn
38598
38599                                   ;--------------------------------------------------------------------------
38600                                   ;
38601                                   ; procedure : PrepareMark
38602                                   ;
38603                                   ;   Input : AX==Address of MCB (not addr of free space), BX==Size
38604                                   ;   Output : None; MCB marked appropriately and DevUMB* set as needed.
38605                                   ;
38606                                   ;--------------------------------------------------------------------------
38607
38608                                   ;   ; 01/11/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
38609                                   ;
38610                                   ;PrepareMark:
38611                                   ;   push    ds
38612                                   ;   mov     ds,ax
38613                                   ;   mov     word [ARENA.OWNER],8
38614                                   ;   mov     word [ARENA.NAME],'SD' ; 4453h
38615                                   ;   pop     ds
38616                                   ;
38617                                   ;   inc     ax
38618                                   ;   mov     [cs:DevUMBAddr],ax
38619                                   ;   mov     [cs:DevUMBFree],ax
38620                                   ;   mov     [cs:DevUMBSize],bx  ; update the UMB Variables
38621                                   ;   retn
38622
38623                                   ;--------------------------------------------------------------------------
38624                                   ;
38625                                   ; procedure : GetUMBForDev
38626                                   ;
38627                                   ;   Input : DevSize
38628                                   ;   Output : Carry set if couldn't allocate a UMB to fit the
38629                                   ;            the device.
38630                                   ;            If success carry clear
38631                                   ;
38632                                   ;   Allocates the biggest UMB for loading devices and updates
38633                                   ;   DevUMBSize, DevUMBAddr & DevUMBFree if it succeeded in allocating
38634                                   ;   UMB.
38635                                   ;
38636                                   ;   This routine relies on the fact that all of the low memory
38637                                   ;   is allocated, and any DOS alloc calls should return memory
38638                                   ;   from the UMB pool.
38639                                   ;
38640                                   ;--------------------------------------------------------------------------
38641                                       ; 01/11/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 IO.SYS)
38642                                       ; (SYSINIT:2BC6h)
38643
38644                                   GetUMBForDev:
38645                                       ; 11/12/2022
38646                                       ; ds = cs
38647 000035D6 BBFFFF                      mov     bx,0FFFFh
38648 000035D9 B80048                      mov     ax,4800h
38649 000035DC CD21                        int     21h
38650                                               ; DOS - 2+ - ALLOCATE MEMORY
38651                                               ; BX = number of 16-byte paragraphs desired
38652
38653 000035DE 09DB                        or      bx,bx
38654                                       ;jz     short gufd_err
38655                                       ; 09/09/2023
38656 000035E0 742E                        jz      short gufd_error ; bx = 0
38657
38658 000035E2 4B                          dec     bx
38659                                       ; 11/12/2022
```

```
                                    ; ds = cs
000035E3 391E[3324]                 cmp     [DevSize],bx
                                    ;cmp    [cs:DevSize],bx
000035E7 7725                       ja      short gufd_err

000035E9 43                         inc     bx

000035EA B80048                     mov     ax,4800h
000035ED CD21                       int     21h
000035EF 721D                       jc      short gufd_err

                                    ; 01/11/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
                                    ;dec    ax
                                    ;call   PrepareMark
                                    ;
                                PrepareMark:
000035F1 1E                         push    ds
000035F2 48                         dec     ax
000035F3 8ED8                       mov     ds,ax
000035F5 C70601000800               mov     word [ARENA.OWNER],8
000035FB C70608005344               mov     word [ARENA.NAME],'SD' ; 4453h
00003601 40                         inc     ax
00003602 1F                         pop     ds
                                    ; 11/12/2022
                                    ; ds = cs
                                    ;mov    [cs:DevUMBSize],bx    ; update the UMB Variables
                                    ;mov    [cs:DevUMBAddr],ax
                                    ;mov    [cs:DevUMBFree],ax
                                gufd_x:             ; 09/09/2023
00003603 891E[4524]                 mov     [DevUMBSize],bx               ; update the UMB Variables
00003607 A3[4324]                   mov     [DevUMBAddr],ax
0000360A A3[4724]                   mov     [DevUMBFree],ax
                                    ;
                                    ; 11/12/2022
                                    ; cf=0
                                    ;clc                           ; mark no error
0000360D C3                         retn
                                    ; 09/09/2023 - Retro DOS v4.2 IO.SYS (Optimization)
                                %if 1
                                gufd_err:
0000360E 31DB                       xor     bx,bx ; 0
                                gufd_error:
00003610 31C0                       xor     ax,ax ; 0
00003612 F9                         stc     ; cf=1
00003613 EBEE                       jmp     short gufd_x
                                %endif

                                ; 09/09/2023 - Retro DOS v4.2 IO.SYS (Optimization)
                                %if 0
                                gufd_err:
                                    xor     ax,ax ; 0
                                    ; 11/12/2022
                                    ; ds = cs
                                    ;mov    [cs:DevUMBSize],ax    ; erase the previous values
                                    ;mov    [cs:DevUMBAddr],ax
                                    ;mov    [cs:DevUMBFree],ax
                                    mov     [DevUMBSize],ax               ; erase the previous values
                                    mov     [DevUMBAddr],ax
                                    mov     [DevUMBFree],ax
                                    stc
                                    retn
                                %endif

                                ;--------------------------------------------------------------------------
                                ;
                                ; procedure : DevSetMark
                                ;
                                ;   Input : AX - Free segment were device is going to be loaded
                                ;   Output : AX - Segment at which device can be loaded (AX=AX+1)
                                ;
                                ;   Creates a sub-arena for the device driver
                                ;   puts 'D' marker in the sub-arena
                                ;   Put the owner of the sub-arena as (AX+1)
                                ;   Copies the file name into sub-arena name field
                                ;
                                ;   Size field of the sub-arena will be set only at succesful
                                ;   completion of Device load.
                                ;
                                ;--------------------------------------------------------------------------

                                    ; 01/11/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 IO.SYS)
                                    ; (SYSINIT:2C13h)

                                DevSetMark:
00003615 06                         push    es
                                    ; 03/01/2023
                                    ;push   di
00003616 1E                         push    ds
00003617 56                         push    si
00003618 8EC0                       mov     es,ax
0000361A 26C606000044               mov     byte [es:devmark.id],devmark_device ; 'D'
00003620 40                         inc     ax
00003621 26A30100                   mov     [es:devmark.seg],ax

                                ;-------------- Copy file name

00003625 50                         push    ax                      ; save load addr

                                    ; 09/09/2023
                                    ; ds = cs
                                    ;lds    si,[cs:bpb_addr]        ; command line is still there
00003626 C536[8103]                 lds     si,[bpb_addr]

0000362A 89F7                       mov     di,si
0000362C FC                         cld
                                dsm_again:
0000362D AC                         lodsb
0000362E 3C3A                       cmp     al,':'
00003630 7504                       jne     short isit_slash
00003632 89F7                       mov     di,si
00003634 EBF7                       jmp     short dsm_again
                                isit_slash:
00003636 3C5C                       cmp     al, '\'
00003638 7504                       jne     short isit_null
0000363A 89F7                       mov     di,si
0000363C EBEF                       jmp     short dsm_again
                                isit_null:
0000363E 08C0                       or      al,al
00003640 75EB                       jnz     short dsm_again
00003642 89FE                       mov     si,di

00003644 BF0800                     mov     di,devmark.filename ; 8
00003647 B90800                     mov     cx,8                    ; maximum 8 characters
```

```
38784                                   dsm_next_char:
38785 0000364A AC                           lodsb
38786 0000364B 08C0                         or      al, al
38787 0000364D 7407                         jz      short blankout
38788 0000364F 3C2E                         cmp     al, '.'
38789 00003651 7403                         je      short blankout
38790 00003653 AA                           stosb
38791 00003654 E2F4                         loop    dsm_next_char
38792                                   blankout:
38793 00003656 E304                         jcxz    dsm_exit
38794 00003658 B020                         mov     al, ' '
38795 0000365A F3AA                         rep     stosb                   ; blank out the rest
38796                                   dsm_exit:
38797 0000365C 58                           pop     ax                      ; restore load addr
38798 0000365D 5E                           pop     si
38799 0000365E 1F                           pop     ds
38800                                       ; 03/01/2023
38801                                       ;pop    di
38802 0000365F 07                           pop     es
38803 00003660 C3                           retn
38804
38805                                   ;--------------------------------------------------------------------------
38806                                   ;
38807                                   ; procedure : SizeDevice
38808                                   ;
38809                                   ;   Input : ES:SI - points to device file to be sized
38810                                   ;
38811                                   ;   Output : Carry set if file cannot be opened or if it is an OS2EXE file
38812                                   ;
38813                                   ;   Calculates the size of the device file in paras and stores it
38814                                   ;   in DevSize
38815                                   ;
38816                                   ;--------------------------------------------------------------------------
38817
38818                                       ; 01/11/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 IO.SYS)
38819                                   SizeDevice:
38820                                       ; 01/11/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
38821                                       ; 11/12/2022 ; *
38822 00003661 1E                           push    ds ; *
38823 00003662 06                           push    es
38824 00003663 1F                           pop     ds
38825 00003664 89F2                         mov     dx,si                   ; ds:dx -> file name
38826 00003666 B8003D                       mov     ax,3D00h                ; open
38827 00003669 CD21                         int     21h
38828 0000366B 7237                         jc      short sd_err            ; open failed
38829
38830 0000366D 89C3                         mov     bx,ax                   ; BX - file handle
38831 0000366F B80242                       mov     ax,4202h                ; seek
38832 00003672 31C9                         xor     cx,cx
38833 00003674 89CA                         mov     dx,cx                   ; to end of file
38834 00003676 CD21                         int     21h
38835 00003678 7223                         jc      short sd_close          ; did seek fail (impossible)
38836 0000367A 83C00F                       add     ax,15                   ; para convert
38837 0000367D 83D200                       adc     dx,0
38838 00003680 F7C2F0FF                     test    dx,0FFF0h               ; size > 0ffffh paras ?
38839                                       ;jz     short szdev1            ; no
38840                                       ; 22/07/2023
38841 00003684 7409                         jz      short sd_ctp
38842 00003686 2EC706[3324]FFFF             mov     word [cs:DevSize],0FFFFh ; invalid device size
38843                                                                       ; assuming that we fail later
38844 0000368D EB0E                         jmp     short sd_close
38845                                   sd_ctp:
38846                                       ; 22/07/2023
38847                                   ;szdev1:
38848 0000368F B104                         mov     cl,4                    ; convert it to paras
38849 00003691 D3E8                         shr     ax,cl
38850 00003693 B10C                         mov     cl,12
38851 00003695 D3E2                         shl     dx,cl
38852 00003697 09D0                         or      ax,dx ; * ; cf=0
38853                                       ;
38854                                       ; 22/07/2023 - Retro DOS v4.2 IO.SYS (optimized)
38855                                       ; MSDOS 6.21 IO.SYS - SYSINIT:37A6h
38856                                       ; 01/11/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
38857                                       ;cmp    ax,[cs:DevSizeOption]
38858                                       ;ja     short szdev2
38859                                       ;mov    ax,[cs:DevSizeOption]
38860                                       ; 12/12/2022
38861                                       ;clc
38862                                   ;szdev2:
38863 00003699 2EA3[3324]                   mov     [cs:DevSize],ax             ; save file size (in paragraps)
38864                                       ; 22/07/2023
38865                                       ;clc ; cf=0 ; *       ; CLC is not needed here
38866                                                             ; (OR instruction clears CF) - E.TAN 22/07/2023
38867
38868                                       ; 01/11/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
38869                                       ; 12/12/2022
38870                                       ; cf=0
38871                                       ;clc
38872                                   sd_close:
38873 0000369D 9C                           pushf                           ; let close not spoil our
38874                                                                       ;  carry flag
38875 0000369E B8003E                       mov     ax,3E00h                ; close
38876 000036A1 CD21                         int     21h                     ; we are not checking for err
38877 000036A3 9D                           popf
38878                                   sd_err:
38879                                       ; 01/11/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
38880                                       ; 11/12/2022 ; *
38881 000036A4 1F                           pop     ds ; *
38882 000036A5 C3                           retn
38883
38884                                   ;--------------------------------------------------------------------------
38885                                   ;
38886                                   ; procedure : ExecDev
38887                                   ;
38888                                   ;   Input : ds:dx -> device to be executed
38889                                   ;           DevLoadAddr - contains where device has to be loaded
38890                                   ;
38891                                   ;   Output : Carry if error
38892                                   ;            Carry clear if no error
38893                                   ;
38894                                   ;   Loads a device driver using the 4b03h function call
38895                                   ;
38896                                   ;--------------------------------------------------------------------------
38897
38898                                       ; 01/11/2022
38899                                   ExecDev:
38900 000036A6 2E8B1E[3524]                 mov     bx,[cs:DevLoadAddr]
38901 000036AB 2E891E[4D24]                 mov     [cs:DevExecAddr],bx     ; Load the parameter block
38902                                                                       ;  block for exec with
38903                                                                       ;  load address
38904 000036B0 2E891E[4F24]                 mov     [cs:DevExecReloc],bx
38905 000036B5 8CCB                         mov     bx,cs
38906 000036B7 8EC3                         mov     es,bx
38907 000036B9 BB[4D24]                     mov     bx,DevExecAddr          ; es:bx points to parameters
```

```
                                    ;mov    al,3    ; (load program only)
                                    ;mov    ah,EXEC ; 4Bh
                                    ; 04/07/2023
000036BC B8034B                     mov     ax,(EXEC<<8)|03h
000036BF CD21                       int     21h                     ; load in the device driver
                                            ; DOS - 2+ - LOAD OR EXECUTE (EXEC)
                                            ; DS:DX -> ASCIZ filename
                                            ; ES:BX -> parameter block
                                            ; AL = subfunction
000036C1 C3                         retn

                                ;--------------------------------------------------------------------------
                                ;
                                ; procedure : RetFromUM
                                ;
                                ;   Input : None
                                ;   Output : ConvLoad set if didn't previously call HideUMBs
                                ;            ConvLoad clear if did.
                                ;
                                ;   Prepares memory for more devices after returning from loading one
                                ;   using the DOS 6 options (/L:... etc).
                                ;
                                ;--------------------------------------------------------------------------

                                ; 31/12/2022 - Retro DOS v4.2 (Modified MSDOS 6.21 IO.SYS)
                                ;  (SYSINIT:37D1h)

                                ; 01/11/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
                                ;%if 0
                                RetFromUM:
                                    ; 31/12/2022
                                    ; ds = cs
000036C2 9C                         pushf
                                    ;mov    byte [cs:ConvLoad],1
000036C3 C606[4124]01               mov     byte [ConvLoad],1
000036C8 E8ECFD                     call    UnHideUMBs
000036CB 7204                       jc      short rfUM1             ; Skip this if didn't HideUMBs
                                    ; 31/12/2022
                                    ; ds = cs
                                    ;;mov   byte [cs:ConvLoad],0
                                    ;mov    byte [ConvLoad],0
                                    ; 09/09/2023
000036CD FE0E[4124]                 dec     byte [ConvLoad] ; -> 0
                                rfUM1:
000036D1 9D                         popf
000036D2 C3                         retn

                                ;%endif ; 01/11/2022

                                ;--------------------------------------------------------------------------
                                ;
                                ; procedure : RemoveNull
                                ;
                                ;   Input : ES:SI points to a null terminated string
                                ;
                                ;   Output : none
                                ;
                                ;   Replaces the null at the end of a string with blank
                                ;
                                ;--------------------------------------------------------------------------

                                    ; 01/11/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 IO.SYS)
                                    ; (SYSINIT:2CCEh)
                                RemoveNull:
                                    ; 11/12/2022
                                    ; ds = cs
                                rn_next:
000036D3 268A1C                     mov     bl,[es:si]
000036D6 08DB                       or      bl,bl                   ; null ?
000036D8 7403                       jz      short rn_gotnull
000036DA 46                         inc     si                      ; advance the pointer
000036DB EBF6                       jmp     short rn_next
                                rn_gotnull:
                                    ; 11/12/2022
000036DD 8A1E[6624]                 mov     bl,[DevSavedDelim]
                                    ;mov    bl,[cs:DevSavedDelim]
000036E1 26881C                     mov     [es:si],bl              ; replace null with blank
                                    ; 02/11/2022
                                ; 11/12/2022
                                rba_ok:             ; 10/04/2019
000036E4 C3                         retn

                                ;--------------------------------------------------------------------------
                                ;
                                ; procedure : RoundBreakAddr
                                ;
                                ;   Input : DevBrkAddr
                                ;   Output : DevBrkAddr
                                ;
                                ;   Rounds DevBrkAddr to a para address so that it is of the form xxxx:0
                                ;
                                ;--------------------------------------------------------------------------

                                RoundBreakAddr:
000036E5 2EA1[3D24]                 mov     ax,[cs:DevBrkAddr]
000036E9 E829DC                     call    ParaRound
000036EC 2E0106[3F24]               add     [cs:DevBrkAddr+2],ax
000036F1 2EC706[3D24]0000           mov     word [cs:DevBrkAddr],0
000036F8 2EA1[3724]                 mov     ax,[cs:DevLoadEnd]
000036FC 2E3906[3F24]               cmp     [cs:DevBrkAddr+2],ax
00003701 76E1                       jbe     short rba_ok
00003703 E92911                     jmp     mem_err
                                    ; 02/11/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
                                    ; 11/12/2022
                                ;rba_ok:
                                ;   retn

                                ;--------------------------------------------------------------------------
                                ;
                                ; procedure : DevSetBreak
                                ;
                                ;   Input : DevBrkAddr
                                ;   Output : Carry set if Device returned Init failed
                                ;            Else carry clear
                                ;
                                ;--------------------------------------------------------------------------

                                DevSetBreak:
00003706 50                         push    ax

00003707 2EA1[3F24]                 mov     ax,[cs:DevBrkAddr+2]   ;remove the init code
0000370B 2E803E[6619]00             cmp     byte [cs:multdeviceflag],0
00003711 750F                       jne     short set_break_continue ;do not check it.
00003713 2E3B06[3524]               cmp     ax,[cs:DevLoadAddr]
```

```
39032 00003718 7508                     jne     short set_break_continue ;if not same, then o.k.
39033
39034                                   ;cmp    word [cs:DevBrkAddr],0
39035                                   ;je     short break_failed    ;[DevBrkAddr+2]=[memhi] & [DevBrkAddr]=0
39036                                   ; 12/12/2022
39037 0000371A 2E833E[3D24]01           cmp     word [cs:DevBrkAddr],1
39038 00003720 7204                     jb      short break_failed
39039
39040                                 set_break_continue:
39041 00003722 E8C0FF                   call    RoundBreakAddr
39042                                   ; 12/12/2022
39043 00003725 F8                       clc
39044                                 break_failed:
39045 00003726 58                       pop     ax
39046                                   ;clc
39047 00003727 C3                       retn
39048
39049                                   ; 12/12/2022
39050                                 ;break_failed:
39051                                   ;pop    ax
39052                                   ;stc
39053                                   ;retn
39054
39055                                 ;--------------------------------------------------------------------------
39056                                 ;
39057                                 ; procedure : DevBreak
39058                                 ;
39059                                 ;   Input : DevLoadAddr & DevBrkAddr
39060                                 ;   Output : none
39061                                 ;
39062                                 ;   Marks a succesful install of a device driver
39063                                 ;   Sets device size field in sub-arena &
39064                                 ;   Updates Free ptr in UMB or adjusts memhi
39065                                 ;
39066                                 ;--------------------------------------------------------------------------
39067
39068                                   ; 11/12/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 IO.SYS)
39069                                 DevBreak:
39070                                   ;push  ds ; 11/12/2022
39071
39072                                   ; 11/12/2022
39073 00003728 0E                       push    cs
39074 00003729 1F                       pop     ds
39075                                   ;mov    ax,[cs:DevLoadAddr]
39076                                   ;mov    bx,[cs:DevBrkAddr+2]
39077 0000372A A1[3524]                 mov     ax,[DevLoadAddr]
39078 0000372D 8B1E[3F24]               mov     bx,[DevBrkAddr+2]
39079                                   ; 11/12/2022
39080 00003731 1E                       push    ds
39081
39082 00003732 48                       dec     ax                      ; seg of sub-arena
39083 00003733 8ED8                     mov     ds,ax
39084 00003735 40                       inc     ax                      ; Back to Device segment
39085 00003736 29D8                     sub     ax,bx
39086 00003738 F7D8                     neg     ax                      ; size of device in paras
39087 0000373A A30300                   mov     [devmark.size],ax       ; store it in sub-arena
39088
39089                                   ; 11/12/2022
39090 0000373D 1F                       pop     ds
39091                                   ; ds = cs
39092
39093 0000373E 803E[5124]00             cmp     byte [DeviceHi],0
39094                                   ;cmp    byte [cs:DeviceHi],0
39095 00003743 7405                     je      short db_lo
39096                                   ;mov    [cs:DevUMBFree],bx      ; update Free ptr in UMB
39097                                   ;jmp    short db_exit
39098                                   ; 11/12/2022
39099 00003745 891E[4724]               mov     [DevUMBFree],bx
39100 00003749 C3                       retn
39101                                 db_lo:
39102                                   ; 11/12/2022
39103                                   ; ds = cs
39104                                   ;mov    [cs:memhi],bx
39105                                   ;mov    word [cs:memlo],0
39106 0000374A 891E[6403]               mov     [memhi],bx
39107 0000374E C706[6203]0000           mov     word [memlo],0 ; 18/12/2022
39108                                 db_exit:
39109                                   ;pop    ds ; 11/12/2022
39110                                 sd_ret:           ; 09/09/2023
39111 00003754 C3                       retn
39112
39113                                 ; 10/04/2019 - Retro DOS v4.0
39114
39115                                 ;--------------------------------------------------------------------------
39116                                 ;
39117                                 ; procedure : ParseSize
39118                                 ;
39119                                 ;   Parses the command line for SIZE= command
39120                                 ;
39121                                 ;   ES:SI = command line to parsed
39122                                 ;
39123                                 ;   returns ptr to command line after SIZE= option in ES:SI
39124                                 ;   updates the DevSizeOption variable with value supplied
39125                                 ;   in SIZE=option
39126                                 ;   Returns carry if the SIZE option was invalid
39127                                 ;
39128                                 ;--------------------------------------------------------------------------
39129
39130                                   ; 02/11/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 IO.SYS)
39131                                   ; (SYSINIT:2D5Ah)
39132
39133                                   ; 09/09/2023 - Retro DOS v4.2 IO.SYS (Optimization) ((&BugFix))
39134                                   ; (MSDOS 6.21 IO.SYS - SYSINIT:3871h) - Retro DOS v4.2 -
39135                                   ; (PCDOS 7.1 IO.SYS - SYSINIT:3D6Eh)  - Retro DOS v5.0 -
39136                                 ParseSize:
39137                                   ;push  bx
39138                                   ;mov   bx,si
39139
39140                                   ; 09/09/2023
39141 00003755 56                       push    si ; * ; mov bx,si
39142
39143                                   ; 11/12/2022
39144                                   ; ds = cs
39145                                   ;mov    word [cs:DevSizeOption],0 ; init the value
39146                                   ;mov    [cs:DevCmdLine],si
39147                                   ;mov    [cs:DevCmdLine+2],es
39148 00003756 C706[5224]0000           mov     word [DevSizeOption],0 ; init the value
39149 0000375C 8936[6224]               mov     [DevCmdLine],si
39150 00003760 8C06[6424]               mov     [DevCmdLine+2],es
39151 00003764 E82400                   call    SkipDelim
39152 00003767 26813C5349               cmp     word [es:si],'SI' ; 4953h
39153 0000376C 7528                     jne     short ps_no_size
39154 0000376E 26817C025A45             cmp     word [es:si+2],'ZE' ; 455Ah
39155 00003774 7520                     jne     short ps_no_size
```

```
00003776 268A4404                   mov     al,[es:si+4]
0000377A E80D10                     call    delim
                                    ;jne    short ps_no_size
                                    ; 22/07/2023
0000377D 7518                       jne     short ps_no_size_2 ; cf=0 here
0000377F 83C605                     add     si,5
00003782 E81400                     call    GetHexNum
00003785 7210                       jc      short ps_err
                                    ; 11/12/2022
                                    ; ds = cs
                                    ;mov    [cs:DevSizeOption],ax
00003787 A3[5224]                   mov     [DevSizeOption],ax

                                    ; 09/09/2023
0000378A 58                         pop     ax  ; * (discard previous si value on top of stack)

                                ;   call    SkipDelim ; **
                                ;
                                ;   ; 22/07/2023
                                ;;ps_no_size_2:
                                ;   ; cf = 0
                                ;   retn

                                    ; 09/09/2023
                                    ;jmp    short SkipDelim

                                ; 09/09/2023 - Retro DOS v4.2 IO.SYS (Optimization)
                                %if 1
                                    ; 01/11/2022
                                SkipDelim:
                                sd_next_char:
0000378B 268A04                     mov     al,[es:si]
0000378E E8F90F                     call    delim
00003791 75C1                       jnz     short sd_ret ; cf=0 ; 09/09/2023
00003793 46                         inc     si
00003794 EBF5                       jmp     short sd_next_char ; 01/11/2022
                                    ; 11/12/2022
                                    ; 01/11/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
                                ;sd_ret:
                                    ;retn
                                %endif

                                    ;;;call SkipDelim ; **
                                    ; 02/11/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
                                    ;mov    bx,si
                                ps_no_size:
                                    ;mov    si,bx
                                    ;pop    bx
00003796 F8                         clc     ; cf=0
                                    ;retn
                                    ; 11/12/2022
                                ps_err:             ; cf=1
                                ps_no_size_2:       ; 09/09/2023 (cf=0)
                                    ; 09/09/2023
00003797 5E                         pop     si ; * ; mov si,bx
                                ;sd_ret:    ; cf=?
00003798 C3                         retn

                                ;ps_err:
                                    ; 02/11/2022
                                    ;pop    bx
                                    ;stc
                                    ; 01/11/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
                                    ; 11/12/2022
                                    ; cf=1
                                    ;stc
                                    ; 11/12/2022
                                ;sd_ret:
                                    ; 22/07/2023
                                    ; 12/04/2019
                                    ;retn

                                ; 12/04/2019 - Retro DOS v4.0

                                ;--------------------------------------------------------------------------
                                ;
                                ; procedure : SkipDelim
                                ;
                                ;   Skips delimiters in the string pointed to by ES:SI
                                ;   Returns ptr to first non-delimiter character in ES:SI
                                ;
                                ;--------------------------------------------------------------------------

                                ; 09/09/2023 - Retro DOS v4.2 IO.SYS (Optimization)
                                %if 0
                                    ; 01/11/2022
                                SkipDelim:
                                sd_next_char:
                                    mov     al,[es:si]
                                    call    delim
                                    jnz     short sd_ret
                                    inc     si
                                    jmp     short sd_next_char ; 01/11/2022
                                    ; 11/12/2022
                                    ; 01/11/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
                                ;sd_ret:
                                    ;retn
                                %endif

                                ;--------------------------------------------------------------------------
                                ;
                                ; procedure : GetHexNum
                                ;
                                ;   Converts an ascii string terminated by a delimiter into binary.
                                ;   Assumes that the ES:SI points to a Hexadecimal string
                                ;
                                ;   Returns in AX the number number of paras equivalent to the
                                ;   hex number of bytes specified by the hexadecimal string.
                                ;
                                ;   Returns carry in case it encountered a non-hex character or
                                ;   if it encountered crlf
                                ;
                                ;--------------------------------------------------------------------------

                                ; 13/05/2019

                                    ; 03/01/2023 - Retro DOS v4.2 (Modified MSDOS 6.21 IO.SYS)
                                    ; (SYSINIT:38C5h)

                                    ; 01/11/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 IO.SYS)
                                    ; (SYSINIT:2DA5h)
                                GetHexNum:
00003799 31C0                       xor     ax,ax
0000379B 31D2                       xor     dx,dx
```

```
39280                                   ghn_next:
39281 0000379D 268A1C                       mov     bl,[es:si]
39282 000037A0 80FB0D                       cmp     bl,cr  ; 0Dh
39283 000037A3 7436                         je      short ghn_err
39284 000037A5 80FB0A                       cmp     bl,lf  ; 0Ah
39285 000037A8 7431                         je      short ghn_err
39286 000037AA 50                           push    ax
39287 000037AB 88D8                         mov     al,bl
39288 000037AD E8DA0F                       call    delim
39289 000037B0 58                           pop     ax
39290                                       ; 03/01/2023
39291 000037B1 B90400                       mov     cx,4
39292 000037B4 7410                         jz      short ghn_into_paras
39293 000037B6 E82400                       call    GetNibble
39294                                       ;jc     short ghn_err
39295                                       ; 11/12/2022
39296 000037B9 7221                         jc      short ghn_ret ; cf=1
39297                                       ; 03/01/2023
39298                                       ;mov    cx,4
39299                                   ghn_shift1:
39300 000037BB D1E0                         shl     ax,1
39301 000037BD D1D2                         rcl     dx,1
39302 000037BF E2FA                         loop    ghn_shift1
39303 000037C1 08D8                         or      al,bl
39304 000037C3 46                           inc     si
39305 000037C4 EBD7                         jmp     short ghn_next
39306                                   ghn_into_paras:
39307 000037C6 83C00F                       add     ax,15
39308 000037C9 83D200                       adc     dx,0
39309 000037CC F7C2F0FF                     test    dx,0FFF0h
39310 000037D0 7509                         jnz     short ghn_err
39311                                       ; 03/01/2023
39312                                       ;mov    cx,4
39313                                   ghn_shift2:
39314 000037D2 F8                           clc
39315 000037D3 D1DA                         rcr     dx,1
39316 000037D5 D1D8                         rcr     ax,1
39317 000037D7 E2F9                         loop    ghn_shift2
39318 000037D9 F8                           clc
39319 000037DA C3                           retn
39320                                       ; 11/12/2022
39321                                   ghn_err:
39322                                   gnib_err:
39323 000037DB F9                           stc
39324                                   ghn_ret:
39325                                   gnib_ret:
39326 000037DC C3                           retn
39327
39328                                   ;--------------------------------------------------------------------------
39329                                   ;
39330                                   ; procedure : GetNibble
39331                                   ;
39332                                   ;   Convert one nibble (hex digit) in BL into binary
39333                                   ;
39334                                   ;   Returns binary value in BL
39335                                   ;
39336                                   ;   Returns carry if BL contains non-hex digit
39337                                   ;
39338                                   ;--------------------------------------------------------------------------
39339
39340                                   GetNibble:
39341 000037DD 80FB30                       cmp     bl,'0'
39342                                       ;jb     short gnib_err
39343                                       ; 11/12/2022
39344 000037E0 72FA                         jb      short gnib_ret ; cf=1
39345 000037E2 80FB39                       cmp     bl,'9'
39346 000037E5 7704                         ja      short is_it_hex
39347 000037E7 80EB30                       sub     bl,'0'          ; clc
39348 000037EA C3                           retn
39349                                   is_it_hex:
39350 000037EB 80FB41                       cmp     bl,'A'
39351                                       ;jb     short gnib_err
39352                                       ; 11/12/2022
39353 000037EE 72EC                         jb      short gnib_ret ; cf=1
39354 000037F0 80FB46                       cmp     bl,'F'
39355 000037F3 77E6                         ja      short gnib_err ; 11/12/2022
39356 000037F5 80EB37                       sub     bl,'A'- 10     ; clc
39357 000037F8 C3                           retn
39358
39359                                       ; 11/12/2022
39360                                   ;gnib_err:
39361                                   ;   stc
39362                                   ;gnib_ret:
39363                                   ;   retn
39364
39365                                   ;==========================================================================
39366
39367                                   ; 12/04/2019 - Retro DOS v4.0
39368
39369                                   ; umb.inc (MSDOS 6.0, 1991)
39370                                   DOS_ARENA   equ 24h        ; offset of arena_head var in DOS data segm.
39371                                   UMB_ARENA   equ 8Ch        ; offset of umb_head in DOS data
39372
39373                                   XMM_REQUEST_UMB    equ 10h
39374                                   XMM_RELEASE_UMB    equ 11h
39375
39376                                   ; 09/09/2023 - Retro DOS v4.2 IO.SYS (Optimization)
39377
39378                                   ;--------------------------------------------------------------------------
39379                                   ;
39380                                   ; Procedure Name   : umb_insert
39381                                   ;
39382                                   ; Inputs           : DOSDATA:UMB_HEAD = start of umb chain
39383                                   ;                  : BX = seg address of UMB to be linked in
39384                                   ;                  : DX = size of UMB to be linked in paras
39385                                   ;                  ; DS = data
39386                                   ;
39387                                   ; Outputs          : links the UMB into the arena chain
39388                                   ;
39389                                   ; Uses                   : AX, CX, ES, DX, BX
39390                                   ;
39391                                   ;--------------------------------------------------------------------------
39392
39393                                   umb_insert:
39394 000037F9 1E                           push    ds
39395
39396                                       ; 31/12/2022
39397                                       ; ds = cs
39398
39399                                       ;mov    ds,[cs:DevDOSData]
39400 000037FA 8E1E[6024]                   mov     ds,[DevDOSData] ; 31/12/2022
39401                                       ;mov    ds,[8Ch]
39402 000037FE 8E1E8C00                     mov     ds,[UMB_ARENA]         ; es = UMB_HEAD
39403 00003802 8CD8                         mov     ax,ds
```

```
39404 00003804 8EC0                     mov     es,ax
39405                                 ui_next:
39406 00003806 39D8                     cmp     ax,bx                   ; Q: is current block above
39407                                                                   ;    new block
39408 00003808 770F                     ja      short ui_insert                 ; Y: insert it
39409                                                                   ; Q: is current block the
39410                                                                   ;    last
39411 0000380A 26803E00005A             cmp     byte [es:ARENA.SIGNATURE],arena_signature_end ; 'Z'
39412 00003810 745C                     je      short ui_append                 ; Y: append new block to chain
39413                                                                   ; N: get next block
39414 00003812 8ED8                     mov     ds,ax                   ; M005
39415                                   ;call   get_next                ; ax = es = next block
39416 00003814 E83B01                   call    _get_next_ ; 13/04/2019 - Retro DOS v4.0
39417 00003817 EBED                     jmp     short ui_next
39418
39419                                 ui_insert:
39420 00003819 8CD9                     mov     cx,ds                   ; ds = previous arena
39421 0000381B 41                       inc     cx                      ; top of previous block
39422
39423 0000381C 29D9                     sub     cx,bx
39424 0000381E F7D9                     neg     cx                      ; cx = size of used block
39425                                   ;mov    byte [0],'M'
39426 00003820 C60600004D               mov     byte [ARENA.SIGNATURE],arena_signature_normal ; 'M'
39427                                   ;mov    word [1],8
39428 00003825 C70601000800             mov     word [ARENA.OWNER],8  ; mark as system owned
39429                                   ;mov    [3],cx
39430 0000382B 890E0300                 mov     [ARENA.SIZE],cx
39431                                   ;mov    word [8],4353h ; 'SC'
39432 0000382F C70608005343             mov     word [ARENA.NAME],'SC' ; 4353h
39433
39434                                 ; prepare the arena at start of new block
39435
39436 00003835 8EC3                     mov     es,bx
39437 00003837 26C60600004D             mov     byte [es:ARENA.SIGNATURE],arena_signature_normal ; 'M'
39438 0000383D 26C70601000000           mov     word [es:ARENA.OWNER],arena_owner_system ; 0
39439                                                                   ; mark as free
39440 00003844 83EA02                   sub     dx,2                    ; make room for arena at
39441                                                                   ; start & end of new block
39442 00003847 2689160300               mov     [es:ARENA.SIZE],dx
39443
39444                                 ; prepare arena at end of new block
39445
39446 0000384C 01D3                     add     bx,dx
39447 0000384E 43                       inc     bx
39448 0000384F 8EC3                     mov     es,bx                   ; es=arena at top of new block
39449 00003851 43                       inc     bx                      ; bx=top of new block
39450
39451                                                                   ; ax contains arena just above
39452                                                                   ; this block
39453 00003852 29D8                     sub     ax,bx                   ; ax = size of used block
39454
39455 00003854 26C60600004D             mov     byte [es:ARENA.SIGNATURE],arena_signature_normal
39456 0000385A 26C70601000800           mov     word [es:ARENA.OWNER],8       ; mark as system owned
39457 00003861 26A30300                 mov     [es:ARENA.SIZE],ax
39458 00003865 26C70608005343           mov     word [es:ARENA.NAME],'SC' ; 4353h
39459
39460 0000386C EB47                     jmp     short ui_done
39461
39462                                 ui_append:
39463                                                                   ; es = arena of last block
39464 0000386E 2603060300               add     ax,[es:ARENA.SIZE]    ; ax=top of last block-1 para
39465 00003873 26832E030001             sub     word [es:ARENA.SIZE],1; reflect the space we are
39466                                                                   ; going to rsrv on top of this
39467                                                                   ; block for the next arena.
39468                                   ; 13/05/2019
39469 00003879 26C60600004D             mov     byte [es:ARENA.SIGNATURE],arena_signature_normal
39470
39471 0000387F 89C1                     mov     cx,ax                   ; cx=top of prev block-1
39472 00003881 40                       inc     ax
39473 00003882 29D8                     sub     ax,bx                   ; ax=top of prev block -
39474                                                                   ;    seg. address of new block
39475 00003884 F7D8                     neg     ax
39476
39477 00003886 8EC1                     mov     es,cx                   ; ds = arena of unused block
39478
39479 00003888 26C60600004D             mov     byte [es:ARENA.SIGNATURE],arena_signature_normal
39480 0000388E 26C70601000800           mov     word [es:ARENA.OWNER],8       ; mark as system owned
39481 00003895 26A30300                 mov     [es:ARENA.SIZE],ax
39482 00003899 26C70608005343           mov     word [es:ARENA.NAME],'SC'
39483
39484                                 ; prepare the arena at start of new block
39485 000038A0 8EC3                     mov     es,bx
39486 000038A2 26C60600005A             mov     byte [es:ARENA.SIGNATURE],arena_signature_end
39487 000038A8 26C70601000000           mov     word [es:ARENA.OWNER],arena_owner_system
39488                                                                   ; mark as free
39489 000038AF 4A                       dec     dx                      ; make room for arena
39490 000038B0 2689160300               mov     [es:ARENA.SIZE],dx
39491                                 ui_done:
39492                                 uc_done: ; 31/12/2022 ; *!
39493 000038B5 1F                       pop     ds
39494                                   ; ds = cs ; 31/12/2022
39495                                 ;uc_done:   ; 18/12/2022
39496                                 au_exit:    ; 09/09/2023
39497 000038B6 C3                       retn
39498
39499                                 ;--------------------------------------------------------------------------
39500                                 ;
39501                                 ; procedure : AllocUMB
39502                                 ;
39503                                 ;   Allocate all UMBs and link it to DOS arena chain
39504                                 ;
39505                                 ;--------------------------------------------------------------------------
39506
39507                                 AllocUMB:
39508                                     ; 31/12/2022
39509                                     ; ds = cs
39510 000038B7 E84700                   call    InitAllocUMB            ; link in the first UMB
39511 000038BA 72FA                     jc      short au_exit           ; quit on error
39512                                 au_next:
39513 000038BC E87000                   call    umb_allocate            ; allocate
39514 000038BF 7205                     jc      short au_coalesce
39515 000038C1 E835FF                   call    umb_insert              ; & insert till no UMBs
39516 000038C4 EBF6                     jmp     short au_next
39517                                 au_coalesce:
39518                                     ; 09/09/2023
39519                                 ;   call    umb_coalesce            ; coalesce all UMBs
39520                                 ;au_exit:
39521                                 ;   ; 31/12/2022
39522                                 ;   ; ds = cs
39523                                 ;   retn
39524
39525                                     ; 09/09/2023
39526                                     ;jmp    short umb_coalesce
39527
```

```
                                ; 09/09/2023 - Retro DOS v4.2 IO.SYS (Optimization)

                                ; 13/04/2019 - Retro DOS v4.0

                                ;--------------------------------------------------------------------------
                                ;
                                ;** umb_coalesce - Combine free blocks ahead with current block
                                ;
                                ;   Coalesce adds the block following the argument to the argument block,
                                ;   if it's free. Coalesce is usually used to join free blocks, but
                                ;   some callers (such as $setblock) use it to join a free block to it's
                                ;   preceeding allocated block.
                                ;
                                ;   EXIT    'C' clear if OK
                                ;             (ds) unchanged, this block updated
                                ;             (ax) = address of next block, IF not at end
                                ;           'C' set if arena trashed
                                ;   USES    cx, di, ds, es
                                ;
                                ;--------------------------------------------------------------------------

                                umb_coalesce:
                                    ; 31/12/2022
                                    ; ds = cs
000038C6 1E                         push    ds ; *!

000038C7 31FF                       xor     di, di

                                    ;mov    es,[cs:DevDOSData]
                                    ; 31/12/2022
000038C9 8E06[6024]                 mov     es,[DevDOSData]
000038CD 268E068C00                 mov     es,[es:UMB_ARENA]     ; es = UMB_HEAD
                                uc_nextfree:
000038D2 8CC0                       mov     ax,es
000038D4 8ED8                       mov     ds,ax
                                    ;cmp    [es:1],di
000038D6 26393E0100                 cmp     [es:ARENA.OWNER],di   ; Q: is current arena free
000038DB 7407                       je      short uc_again        ; Y: try to coalesce with next block
                                                                  ; N: get next arena
000038DD E86B00                     call    get_next              ; es, ax = next arena
000038E0 72D3                       jc      short uc_done  ; *!
000038E2 EBEE                       jmp     short uc_nextfree
                                uc_again:
000038E4 E86400                     call    get_next              ; es, ax = next arena
000038E7 72CC                       jc      short uc_done  ; *!
                                uc_check:
000038E9 26393E0100                 cmp     [es:ARENA.OWNER],di   ; Q: is arena free
000038EE 75E2                       jne     short uc_nextfree     ; N: get next free arena
                                                                  ; Y: coalesce
000038F0 268B0E0300                 mov     cx,[es:ARENA.SIZE]      ; cx <- next block size
000038F5 41                         inc     cx                      ; cx <- cx + 1 (for header size)
                                    ;add    [3],cx
000038F6 010E0300                   add     [ARENA.SIZE],cx             ; current size <- current size + cx
000038FA 268A0D                     mov     cl,[es:di]              ; move up signature
000038FD 880D                       mov     [di],cl
000038FF EBE3                       jmp     short uc_again        ; try again

                                    ; 18/12/2022
                                ;uc_done:
                                    ;retn

                                ;--------------------------------------------------------------------------
                                ;
                                ; procedure : InitAllocUMB
                                ;
                                ;--------------------------------------------------------------------------

                                InitAllocUMB:
                                    ; 31/12/2022
                                    ; ds = cs
00003901 E8D6D2                     call    IsXMSLoaded
00003904 7527                       jnz     short iau_err         ; quit on no XMS driver
00003906 B452                       mov     ah,52h
00003908 CD21                       int     21h                   ; get DOS DATA seg
                                    ; 31/12/2022
                                    ; ds = cs
                                    ;mov    [cs:DevDOSData],es    ; & save it for later
0000390A 8C06[6024]                 mov     [DevDOSData],es             ; & save it for later
0000390E B81043                     mov     ax,4310h
00003911 CD2F                       int     2Fh
                                    ;mov    [cs:DevXMSAddr],bx    ; get XMS driver address
                                    ;mov    [cs:DevXMSAddr+2],es
00003913 891E[4924]                 mov     [DevXMSAddr],bx             ; get XMS driver address
00003917 8C06[4B24]                 mov     [DevXMSAddr+2],es
                                    ; 31/12/2022
0000391B 803E[5F24]00               cmp     byte [FirstUMBLinked],0
                                    ;cmp    byte [cs:FirstUMBLinked],0 ; have we already linked a UMB?
                                    ;jne    short ia_1            ; quit if we already did it
                                    ; 12/12/2022
00003920 770A                       ja      short ia_1 ; cf=0
00003922 E83900                     call    LinkFirstUMB          ; else link the first UMB
                                    ;jc     short iau_err
                                    ; 12/12/2022
00003925 7207                       jc      short iau_err2  ; cf=1
                                    ; 31/12/2022
                                    ; ds = cs
00003927 C606[5F24]FF               mov     byte [FirstUMBLinked],0FFh ; mark that 1st UMB linked
                                    ;mov    byte [cs:FirstUMBLinked],0FFh ; mark that 1st UMB linked
                                ia_1:
                                    ; 12/12/2022
                                    ; cf=0
                                    ;clc
0000392C C3                         retn
                                iau_err:
0000392D F9                         stc
                                iau_err2:
0000392E C3                         retn

                                ;--------------------------------------------------------------------------
                                ;
                                ; Procedure Name   : umb_allocate
                                ;
                                ; Inputs           : DS = data
                                ;
                                ; Outputs          : if UMB available
                                ;                        Allocates the largest available UMB and
                                ;                        BX = segment of allocated block
                                ;                        DX = size of allocated block
                                ;                        NC
                                ;                    else
                                ;                        CY
                                ;
                                ; Uses                   : BX, DX
                                ;
```

```
                                ;--------------------------------------------------------------------------
                                
                                umb_allocate:
                                    ; 31/12/2022
                                    ; ds = cs
0000392F 50                         push    ax
00003930 B410                       mov     ah,XMM_REQUEST_UMB ; 16
00003932 BAFFFF                     mov     dx,0FFFFh               ; try to allocate largest
                                                                    ;   possible
                                    ; 31/12/2022
00003935 FF1E[4924]                 call    far [DevXMSAddr]
                                    ;call   far [cs:DevXMSAddr]
                                                                    ; dx now contains the size of
                                                                    ; the largest UMB
00003939 09D2                       or      dx,dx
0000393B 740B                       jz      short ua_err
                                
0000393D B410                       mov     ah,XMM_REQUEST_UMB ; 16
                                
                                    ; 31/12/2022
0000393F FF1E[4924]                 call    far [DevXMSAddr]
                                    ;call   far [cs:DevXMSAddr]
                                
00003943 83F801                     cmp     ax,1                    ; Q: was the reqst successful
                                    ;jne    short ua_err            ; N: error
                                    ; 27/07/2023 - Retro DOS v4.2 IO.SYS (optimization)
00003946 7601                       jna     short ua_done ; if ax=1 then cf=0, else cf=1 (ax=0)
                                ua_err:
00003948 F9                         stc
                                
                                    ;clc
                                    ; 02/11/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
                                    ; 12/12/2022
                                    ; cf=0
                                    ;clc
                                ua_done:
00003949 58                         pop     ax
0000394A C3                         retn
                                    ; 27/07/2023
                                ;ua_err:
                                    ;stc
                                    ;jmp    short ua_done
                                
                                ;--------------------------------------------------------------------------
                                ;
                                ;** get_next - Find Next item in Arena
                                ;
                                ;   ENTRY   dS - pointer to block head
                                ;   EXIT    AX,ES - pointers to next head
                                ;           'C' set if arena damaged
                                ;
                                ;--------------------------------------------------------------------------
                                
                                    ; 01/11/2022
                                get_next:
0000394B 803E00005A                 cmp     byte [0],arena_signature_end ; 'Z'
00003950 740A                       je      short gn_err
                                _get_next_:
00003952 8CD8                       mov     ax,ds                   ; ax=current block
00003954 03060300                   add     ax,[ARENA.SIZE]             ; ax=ax + current block length
00003958 40                         inc     ax                      ; remember that header!
00003959 8EC0                       mov     es,ax
                                    ;clc
                                    ; 01/11/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
                                    ; 11/12/2022
                                    ; cf=0
                                    ;clc
0000395B C3                         retn
                                gn_err:
0000395C F9                         stc
                                    ; 11/12/2022
                                lfu_err:    ; cf=1
0000395D C3                         retn
                                
                                ;--------------------------------------------------------------------------
                                ;
                                ; procedure : LinkFirstUMB
                                ;
                                ;--------------------------------------------------------------------------
                                
                                    ; 01/11/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 IO.SYS)
                                    ; (SYSINIT:2F81h)
                                LinkFirstUMB:
                                    ; 31/12/2022
                                    ; ds = cs
0000395E E8CEFF                     call    umb_allocate
00003961 72FA                       jc      short lfu_err  ; ds = cs ; 31/12/2022
                                
                                ; bx = segment of allocated UMB
                                ; dx = size of UMB
                                
                                    ; 31/12/2022
                                    ; ds = cs
                                
00003963 CD12                       int     12h                     ; ax = size of memory
00003965 B106                       mov     cl,6
00003967 D3E0                       shl     ax,cl                   ; ax = size in paragraphs
                                
00003969 89C1                       mov     cx,ax                   ; cx = size in paras
0000396B 29D8                       sub     ax,bx                   ; ax = - size of unused block
                                
0000396D F7D8                       neg     ax
                                
                                    ;sub    cx,1                    ; cx = first umb_arena
                                    ; 09/09/2023
0000396F 49                         dec     cx
00003970 8EC1                       mov     es,cx                   ; es = first umb_arena
                                
00003972 26C60600004D               mov     byte [es:ARENA.SIGNATURE],arena_signature_normal ; 'M'
00003978 26C70601000800             mov     word [es:ARENA.OWNER],8       ; mark as system owned
                                
0000397F 26A30300                   mov     [es:ARENA.SIZE],ax
00003983 26C70608005343             mov     word [es:ARENA.NAME],'SC' ; 4353h
                                
                                ; put in the arena for the first UMB
                                
0000398A 8EC3                       mov     es,bx                   ; es has first free umb seg
0000398C 26C60600005A               mov     byte [es:ARENA.SIGNATURE],arena_signature_end ; 'Z'
00003992 26C70601000000             mov     word [es:ARENA.OWNER],arena_owner_system ; 0
                                                                    ; mark as free
00003999 4A                         dec     dx                      ; make room for arena
0000399A 2689160300                 mov     [es:ARENA.SIZE],dx
                                
                                    ;mov    es,[cs:DevDOSData]
```

```
                                    ; 31/12/2022
000039F F 8E06[6024]                mov     es,[DevDOSData] ; ds = cs
000039A3 BF8C00                     mov     di,UMB_ARENA ; 8Ch
000039A6 26890D                     mov     [es:di],cx              ; initialize umb_head in DOS
                                                                    ;  data segment with the arena
                                                                    ;  just below Top of Mem

                                ; we must now scan the arena chain and update the size of the last arena

000039A9 BF2400                     mov     di,DOS_ARENA ; 24h
000039AC 268E05                     mov     es,[es:di]              ; es = start arena
000039AF 31FF                       xor     di,di
                                ;scan_next
                                ; 09/12/2022
                                scannext:
000039B1 26803D5A                   cmp     byte [es:di],arena_signature_end  ; 'Z'
000039B5 740C                       je      short got_last

000039B7 8CC0                       mov     ax,es
000039B9 2603060300                 add     ax,[es:ARENA.SIZE]
000039BE 40                         inc     ax
000039BF 8EC0                       mov     es,ax
                                    ;jmp    short scan_next
                                    ; 09/12/2022
000039C1 EBEE                       jmp     short scannext
                                got_last:
                                    ;sub    word [es:ARENA.SIZE],1
                                    ; 09/09/2023
000039C3 26FF0E0300                 dec     word [es:ARENA.SIZE]

000039C8 26C60600004D               mov     byte [es:ARENA.SIGNATURE],arena_signature_normal ; 'M'
                                    ;clc
                                    ; 01/11/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
                                    ; 11/12/2022
                                    ; cf=0
                                    ;clc
000039CE C3                         retn
                                    ; 11/12/2022
                                    ;;lfu_err:
                                    ;   ;stc
                                    ;   ; 01/11/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
                                    ;   ; 11/12/2022
                                    ;   ; cf=1
                                    ;   ;stc
                                    ;   retn

                                ;--------------------------------------------------------------------------
                                ;
                                ; procedure : ShrinkUMB
                                ;
                                ;   Shrinks the current UMB in use, so that the unused portions
                                ;   of the UMB is given back to the DOS free mem pool
                                ;
                                ;--------------------------------------------------------------------------

                                ShrinkUMB:
                                    ; 12/12/2022
                                    ; ds = cs
000039CF 833E[4324]00               cmp     word [DevUMBAddr],0
                                    ;cmp    word [cs:DevUMBAddr],0
000039D4 741F                       je      short su_exit
000039D6 06                         push    es
                                    ; 01/01/2023
                                    ;push   bx
                                    ; 12/12/2022
                                    ;mov    bx,[cs:DevUMBFree]
                                    ;sub    bx,[cs:DevUMBAddr]
                                    ;mov    es,[cs:DevUMBAddr]
000039D7 8B1E[4724]                 mov     bx,[DevUMBFree]
000039DB 2B1E[4324]                 sub     bx,[DevUMBAddr]
000039DF 8E06[4324]                 mov     es,[DevUMBAddr]

000039E3 B8004A                     mov     ax,4A00h
000039E6 CD21                       int     21h
                                            ; DOS - 2+ - ADJUST MEMORY BLOCK SIZE (SETBLOCK)
                                            ; ES = segment address of block to change
                                            ; BX = new size in paragraphs
000039E8 8CC0                       mov     ax,es
000039EA 48                         dec     ax
000039EB 8EC0                       mov     es,ax
000039ED 26C70601000800             mov     word [es:ARENA.OWNER],8
                                    ; 01/01/2023
                                    ;pop    bx
000039F4 07                         pop     es
                                su_exit:
000039F5 C3                         retn

                                ;--------------------------------------------------------------------------
                                ;
                                ; procedure : UnlinkUMB
                                ;
                                ;   Unlinks the UMBs from the DOS arena chain
                                ;
                                ;--------------------------------------------------------------------------

                                UnlinkUMB:
                                    ; 12/12/2022
                                    ; ds = cs
000039F6 1E                         push    ds
000039F7 06                         push    es
                                    ; 12/12/2022
000039F8 803E[5F24]00               cmp     byte [FirstUMBLinked],0
                                    ;cmp    byte [cs:FirstUMBLinked],0
000039FD 7420                       je      short ulu_x             ; nothing to unlink
                                    ; 12/12/2022
000039FF 8E06[6024]                 mov     es,[DevDOSData]
                                    ;mov    es,[cs:DevDOSData]    ; get DOS data seg
00003A03 268E1E2400                 mov     ds,[es:DOS_ARENA]
00003A08 268B3E8C00                 mov     di,[es:UMB_ARENA]
                                ulu_next:
00003A0D E83BFF                     call    get_next
00003A10 720D                       jc      short ulu_x
00003A12 39C7                       cmp     di,ax                   ; is the next one UMB ?
00003A14 7404                       je      short ulu_found
00003A16 8ED8                       mov     ds,ax
00003A18 EBF3                       jmp     short ulu_next
                                ulu_found:
                                    ;mov    byte [0],'Z'
00003A1A C60600005A                 mov     byte [ARENA.SIGNATURE],arena_signature_end ; 'Z'
                                ulu_x:
00003A1F 07                         pop     es
00003A20 1F                         pop     ds
00003A21 C3                         retn
```

```
                                ; ----------------------------------------------------------------------------
                                ; SYSINIT2.ASM - MSDOS 6.0 - 1991
                                ; ----------------------------------------------------------------------------
                                ; 14/04/2019 - Retro DOS v4.0

                                ; Multiple configuration block support  Created 16-Mar-1992 by JeffPar
                                ;
                                ; Summary:
                                ;
                                ;   The procedure "organize" crunches the in-memory copy of config.sys
                                ;   into lines delimited by CR/LF (sometimes no CR, but *always* an LF)
                                ;   with the leading "keyword=" replaced by single character codes (eg, B
                                ;   for BUFFERS, D for DEVICE, Z for any unrecognized keyword); see comtab
                                ;   and/or config.inc for the full list.
                                ;
                                ;   [blockname] and INCLUDE are the major syntactical additions for multi-
                                ;   configuration support. blockname is either MENU, which contains one
                                ;   or more MENUITEM lines, an optional MENUDEFAULT (which includes optional
                                ;   time-out), or any user-defined keyword, such as NETWORK, CD-ROM, etc.
                                ;   INCLUDE allows the current block to name another block for inclusion
                                ;   during the processing phase of CONFIG.SYS. An INCLUDE is only honored
                                ;   once, precluding nasty infinite-loop scenarios. If blocks are present
                                ;   without a MENU block, then only lines inside COMMON blocks are processed.
                                ;
                                ; Example:
                                ;
                                ;   [menu]
                                ;   menuitem=misc,Miscellaneous
                                ;   menuitem=network,Network Configuration
                                ;   menudefault=network,15
                                ;
                                ;   [network]
                                ;   include misc
                                ;   device=foo
                                ;
                                ;   [misc]
                                ;   device=bar
                                ;   include alternate
                                ;
                                ;   [alternate]
                                ;   device=tar
                                ;
                                ;
                                ;   When the menu is displayed
                                ;
                                ;    1. Miscellaneous
                                ;    2. Network Configuration
                                ;
                                ;   #2 is highlighted as the default option, and will be automatically
                                ;   selected after 15 seconds. It will invoke the following lines in the
                                ;   following order:
                                ;
                                ;       DEVICE=BAR
                                ;       DEVICE=TAR
                                ;       DEVICE=FOO
                                ;

                                ;MULTI_CONFIG equ 1

                                ; the following depend on the positions of the various letters in switchlist

                                switchnum   equ 11111000b ; 0F8h  ; which switches require number

                                flagec35    equ 00000100b ; 4     ; electrically compatible 3.5 inch disk drive
                                flagdrive   equ 00001000b ; 8
                                flagcyln    equ 00010000b ; 16
                                flagseclim  equ 00100000b ; 32
                                flagheads   equ 01000000b ; 64
                                flagff              equ 10000000b ; 128

                                ;----------------------------------------------------------------------------
                                ; 19/04/2019 - Retro DOS v4.0

                                ; MSDOS 6.21 IO.SYS - SYSINIT:3E78h

                                ; 01/11/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 IO.SYS)
                                ; MSDOS 5.0 IO.SYS - SYSINIT:3054h

00003A22 00                     insert_blank:      db     0

                                ;----------------------------------------------------------------------------
                                ;
                                ; procedure : setparms
                                ;
                                ; the following set of routines is used to parse the drivparm = command in
                                ; the config.sys file to change the default drive parameters.
                                ;
                                ;----------------------------------------------------------------------------

                                setparms:
00003A23 1E                         push    ds
00003A24 50                         push    ax
00003A25 53                         push    bx
00003A26 51                         push    cx
00003A27 52                         push    dx

00003A28 0E                         push    cs
00003A29 1F                         pop     ds

00003A2A 31DB                       xor     bx,bx
00003A2C 8A1E[1C4F]                 mov     bl,[drive]
                                    ; 18/12/2022
00003A30 43                         inc     bx
                                    ;inc    bl                      ; get it correct for ioctl call
                                                                    ; (1=a,2=b...)
00003A31 BA[BE4D]                   mov     dx,deviceparameters
                                    ;mov    ah,IOCTL ; 44h
                                    ;mov    al,GENERIC_IOCTL ; 0Dh
                                    ; 04/07/2023
00003A34 B80D44                     mov     ax,(IOCTL<<8)|GENERIC_IOCTL
                                    ;mov    ch,RAWIO ; 8
                                    ;mov    cl,SET_DEVICE_PARAMETERS ; 40h
                                    ; 04/07/2023
00003A37 B94008                     mov     cx,(RAWIO<<8)|SET_DEVICE_PARAMETERS
00003A3A CD21                       int     21h

                                ; 27/07/2023 - Retro DOS v4.2 IO.SYS (optimization)
00003A3C 8A26[1D4F]                 mov     ah,[switches]
                                    ;mov    al,[deviceparameters+20]
00003A40 A0[D24D]                   mov     al,[deviceparameters+A_DEVICEPARAMETERS.DP_BPB+A_BPB.SECTORSPERTRACK]
00003A43 8A0E[1C4F]                 mov     cl,[drive]
                                ;
                                ;; 01/11/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
```

```
40024                                   ;   ;;mov  ax,Bios_Data          ; get Bios_Data segment
40025                                   ;   ;mov   ax,KERNEL_SEGMENT ; 70h
40026                                   ;   ; 21/10/2022
40027                                   ;   ;mov   ax,DOSBIODATASEG ; 0070h
40028                                   ;   ;mov   ds,ax                 ; set Bios_Data segment
40029                                   ;
40030                                   ;   ; 27/07/2023
40031                                   ;   ;;test  word [cs:switches],flagec35 ; 4
40032                                   ;   ;test  byte [cs:switches],flagec35
40033                                   ;   ;jz    short not_ec35
40034                                   ;
40035                                   ;   ; 27/07/2023
40036                                   ;   ; 01/11/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
40037                                   ;   ;;test  word [switches],flagec35 ; 4
40038                                   ;   ; 12/12/2022
40039                                   ;   ;test  byte [switches],flagec35 ; 4
40040                                   ;   ;jz    short eot_ok
40041                                   ;
40042                                       ;mov   cl,[cs:drive]         ; which drive was this for?
40043                                       ; 02/11/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
40044                                       ;mov   cl,[drive]
40045                                       ; 27/07/2023
40046                                       ;mov   ax,DOSBIODATASEG ; 0070h
40047                                       ;mov   ds,ax
40048
40049 00003A47 BA7000                      mov    dx,DOSBIODATASEG
40050 00003A4A 8EDA                       mov    ds,dx
40051
40052 00003A4C F6C404                     test   ah,flagec35    ; test byte [cs:switches],flagec35
40053 00003A4F 7408                       jz     short not_ec35
40054
40055                                     ;mov   al,1                  ; assume drive 0
40056                                     ;shl   al,cl                 ; set proper bit depending on drive
40057                                     ;;or   [531h],al ; (MSDOS 6.21 IO.SYS Offset SYINIT:3EACh)
40058                                     ;or    [ec35_flag],al        ; set the bit in the permanent flags
40059                                     ; 27/07/2023
40060 00003A51 B401                       mov    ah,1
40061 00003A53 D2E4                       shl    ah,cl
40062 00003A55 0826[A204]                 or     [ec35_flag],ah
40063
40064                                   ; 07/07/2023 - Retro DOS v4.2 IO.SYS (optimization)
40065                                   ;   MSDOS 6.21 IO.SYS - SYINIT:3EB0h
40066                                   ; 01/11/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
40067                                   not_ec35:
40068                                   ;   Now adjust the BIOS's EOT variable if our new drive has more
40069                                   ;   sectors per track than any old ones.
40070
40071                                       ; 27/07/2023
40072                                       ;;mov  al,[cs:deviceparameters+20]
40073                                       ;mov   al,[cs:deviceparameters+A_DEVICEPARAMETERS.DP_BPB+A_BPB.SECTORSPERTRACK]
40074
40075                                       ;cmp   al,[12Ch] ; (MSDOS 6.21 IO.SYS Offset SYINIT:3EB4h)
40076 00003A59 3A06[2C01]                   cmp    al,[eot]
40077 00003A5D 7603                         jbe    short eot_ok
40078 00003A5F A2[2C01]                     mov    [eot],al
40079                                   eot_ok:
40080 00003A62 5A                           pop    dx                    ; fix up all the registers
40081 00003A63 59                           pop    cx
40082 00003A64 5B                           pop    bx
40083 00003A65 58                           pop    ax
40084 00003A66 1F                           pop    ds ; 13/05/2019
40085 00003A67 C3                           retn
40086
40087                                   ;--------------------------------------------------------------------------
40088                                   ;
40089                                   ; procedure : diddleback
40090                                   ;
40091                                   ; replace default values for further drivparm commands
40092                                   ;
40093                                   ;--------------------------------------------------------------------------
40094
40095                                   diddleback:
40096 00003A68 1E                           push   ds
40097 00003A69 0E                           push   cs
40098 00003A6A 1F                           pop    ds
40099                                       ;mov   word [deviceparameters+4],80
40100 00003A6B C706[C24D]5000               mov    word [deviceparameters+A_DEVICEPARAMETERS.DP_CYLINDERS],80
40101                                       ;mov   byte [deviceparameters+1],2
40102 00003A71 C606[BF4D]02                 mov    byte [deviceparameters+A_DEVICEPARAMETERS.DP_DEVICETYPE],DEV_3INCH720KB ; 2
40103                                       ;mov   word [deviceparameters+2],0
40104 00003A76 C706[C04D]0000               mov    word [deviceparameters+A_DEVICEPARAMETERS.DP_DEVICEATTRIBUTES],0
40105 00003A7C C706[1D4F]0000               mov    word [switches],0         ; zero all switches
40106 00003A82 1F                           pop    ds
40107 00003A83 C3                           retn
40108
40109                                   ; 03/01/2023
40110                                   %if 0
40111
40112                                   ; 15/04/2019 - Retro DOS v4.0
40113
40114                                   ;--------------------------------------------------------------------------
40115                                   ;
40116                                   ; procedure : parseline
40117                                   ;
40118                                   ; entry point is parseline. al contains the first character in command line.
40119                                   ;
40120                                   ;--------------------------------------------------------------------------
40121
40122                                       ; 03/01/2023 - Retrodos v4.2 (Modified MSDOS 6.21 IO.SYS)
40123                                       ; (SYSINIT:3EDFh)
40124
40125                                       ; 01/11/2022 - Retrodos v4.0 (Modified MSDOS 5.0 IO.SYS)
40126                                       ; (SYSINIT:30ACh)
40127                                   parseline:
40128                                       ; 03/01/2023
40129                                       ; ds = cs ; *
40130
40131                                       ;push  ds ; *
40132
40133                                       ;push  cs ; *
40134                                       ;pop   ds ; *
40135
40136                                   nextswtch:
40137                                       cmp    al,cr                 ; carriage return?
40138                                       je     short done_line
40139                                       cmp    al,lf                 ; linefeed?
40140                                       je     short put_back        ; put it back and done
40141
40142                                   ; anything less or equal to a space is ignored.
40143
40144                                       cmp    al,' '                  ; space?
40145                                       jbe    short getnext         ; skip over space
40146                                       cmp    al,'/'
40147                                       je     short getparm
```

```
                                    stc                             ; mark error invalid-character-in-input
                                    ;jmp    short exitpl
                                    ; 03/01/2023
                                swterr:
                                    retn

                                getparm:
                                    call    check_switch
                                    mov     [switches],bx           ; save switches read so far
                                    jc      short swterr
                                getnext:
                                    call    getchr
                                    ;jc     short done_line
                                    ;jmp    short nextswtch
                                    ; 03/01/2023
                                    jnc     short nextswtch
                                ;swterr:
                                    ;jmp    short exitpl            ; exit if error

                                done_line:
                                    ; 12/12/2022
                                    test    byte [switches],flagdrive ; 8
                                    ;test   word [switches],flagdrive ; 8 ; see if drive specified
                                    jnz     short okay
                                    stc                             ; mark error no-drive-specified
                                    ;jmp    short exitpl
                                    ; 03/01/2023
                                    retn

                                okay:
                                    mov     ax,[switches]
                                    and     ax,0003h            ; get flag bits for changeline and non-rem
                                    mov     [deviceparameters+A_DEVICEPARAMETERS.DP_DEVICEATTRIBUTES],ax
                                    mov     word [deviceparameters+A_DEVICEPARAMETERS.DP_TRACKTABLEENTRIES],0
                                    ;clc                            ; everything is fine
                                    ; 01/11/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
                                    ; 12/12/2022
                                    ; cf=0
                                    ;clc
                                    ;call   setdeviceparameters
                                    ; 03/01/2023
                                    jmp     setdeviceparameters
                                ;exitpl:
                                    ; 03/01/2023
                                    ; ds = cs
                                    ;pop    ds ; *
                                    retn
                                put_back:
                                    inc     word [count]            ; one more char to scan
                                    dec     word [chrptr]           ; back up over linefeed
                                    jmp     short done_line

                                %endif

                                ;--------------------------------------------------------------------------
                                ;
                                ; procedure : check_switch
                                ;
                                ; processes a switch in the input. it ensures that the switch is valid, and
                                ; gets the number, if any required, following the switch. the switch and the
                                ; number *must* be separated by a colon. carry is set if there is any kind of
                                ; error.
                                ;
                                ;--------------------------------------------------------------------------

                                ; 09/09/2023

                                err_swtch:
00003A84 31CB                       xor     bx,cx                   ; remove this switch from the records
                                err_check:
00003A86 F9                         stc
                                err_chk:
                                done_swtch: ; 09/09/2023 (cf=0)
00003A87 C3                         retn

                                    ; 09/09/2023 - Retro DOS v4.2 IO.SYS (Optimization)

                                check_switch:
00003A88 E8240D                     call    getchr
                                    ;jc     short err_check
00003A8B 72FA                       jc      short err_chk
00003A8D 24DF                           and     al,0DFh                 ; convert it to upper case
00003A8F 3C41                       cmp     al,'A'
                                    ;jb     short err_check
00003A91 72F4                       jb      short err_chk ; 15/04/2019 - Retro DOS v4.0
00003A93 3C5A                       cmp     al,'Z'
00003A95 77EF                       ja      short err_check

00003A97 06                         push    es

00003A98 0E                         push    cs
00003A99 07                         pop     es

                                    ;mov    cl,[switchlist]             ; get number of valid switches
                                    ;mov    ch,0
                                    ;mov    di,1+switchlist             ; point to string of valid switches
                                    ; 09/09/2023
00003A9A BF[4250]                   mov     di,switchlist
00003A9D 8A0D                       mov     cl,[di]
00003A9F B500                       mov     ch,0
00003AA1 47                         inc     di      ; 1+switchlist

00003AA2 F2AE                       repne   scasb

00003AA4 07                         pop     es
00003AA5 75DF                       jnz     short err_check

00003AA7 B80100                     mov     ax,1
00003AAA D3E0                       shl     ax,cl                   ; set bit to indicate switch
00003AAC 8B1E[1D4F]                 mov     bx,[switches]           ; get switches so far
00003AB0 09C3                       or      bx,ax                   ; save this with other switches
00003AB2 89C1                       mov     cx,ax
                                    ; 12/12/2022
00003AB4 A8F8                       test    al,switchnum ; 0F8h
                                    ;test   ax,switchnum ; 0F8h     ; test against switches that require number to follow
00003AB6 74CF                       jz      short done_swtch

00003AB8 E8F40C                     call    getchr
00003ABB 72C7                       jc      short err_swtch

00003ABD 3C3A                       cmp     al,':'
00003ABF 75C3                       jne     short err_swtch

00003AC1 E8EB0C                     call    getchr
```

```
00003AC4 53                             push    bx                      ; preserve switches
                                        ; 02/11/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
                                        ;mov    byte [cs:sepchr],' '    ; allow space separators
                                        ; 12/12/2022
                                        ; ds = cs
00003AC5 C606[AE02]20                   mov     byte [sepchr],' '
00003ACA E8980D                         call    getnum
                                        ;mov    byte [cs:sepchr],0
                                        ; 12/12/2022
00003ACD C606[AE02]00                   mov     byte [sepchr],0
00003AD2 5B                             pop     bx                      ; restore switches

                                    ; because getnum does not consider carriage-return or line-feed as ok, we do
                                    ; not check for carry set here. if there is an error, it will be detected
                                    ; further on (hopefully).

                                        ; 09/09/2023
                                        ;call   process_num
                                        ;jmp    short process_num

                                    ;done_swtch:
                                    ;   ;clc
                                    ;   ; 02/11/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
                                    ;   ; 12/12/2022
                                    ;   ; cf=0
                                    ;   ;clc
                                    ;   retn

                                    ;--------------------------------------------------------------------------
                                    ;
                                    ; procedure : process_num
                                    ;
                                    ; this routine takes the switch just input, and the number following (if any),
                                    ; and sets the value in the appropriate variable. if the number input is zero
                                    ; then it does nothing - it assumes the default value that is present in the
                                    ; variable at the beginning. zero is ok for form factor and drive, however.
                                    ;
                                    ;--------------------------------------------------------------------------

                                        ; 02/11/2022 - Retrodos v4.0 (Modified MSDOS 5.0 IO.SYS)
                                        ; (SYSINIT:3156h)
                                    process_num:
00003AD3 850E[1D4F]                     test    [switches],cx           ; if this switch has been done before,
00003AD7 752B                           jnz     short done_ret          ; ignore this one.
                                        ; 12/12/2022
00003AD9 F6C108                         test    cl,flagdrive ; 8
                                        ;test   cx,flagdrive ; 8
00003ADC 7404                           jz      short try_f
00003ADE A2[1C4F]                       mov     byte [drive],al
                                        ; 02/11/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
                                        ;jmp    short done_ret
                                        ; 12/12/2022
                                        ; cf=0
00003AE1 C3                             retn    ; 13/05/2019
                                    try_f:
                                        ; 12/12/2022
00003AE2 F6C180                         test    cl,flagff ; 80h
                                        ;test   cx,flagff ; 80h
00003AE5 7404                           jz      short try_t

                                    ; ensure that we do not get bogus form factors that are not supported

                                        ;mov    [deviceparameters+1],al
00003AE7 A2[BF4D]                       mov     [deviceparameters+A_DEVICEPARAMETERS.DP_DEVICETYPE],al
                                        ; 02/11/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
                                        ;jmp    short done_ret
                                        ; 12/12/2022
                                        ; cf=0
00003AEA C3                             retn    ; 13/05/2019
                                    try_t:
00003AEB 09C0                           or      ax,ax
00003AED 7415                           jz      short done_ret          ; if number entered was 0, assume default value
                                        ; 12/12/2022
00003AEF F6C110                         test    cl,flagcyln ; 10h
                                        ;test   cx,flagcyln ; 10h
00003AF2 7404                           jz      short try_s

                                        ;mov    [deviceparameters+4],ax
00003AF4 A3[C24D]                       mov     [deviceparameters+A_DEVICEPARAMETERS.DP_CYLINDERS],ax
                                        ; 02/11/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
                                        ;jmp    short done_ret
                                        ; 12/12/2022
                                        ; cf=0
00003AF7 C3                             retn    ; 13/05/2019
                                    try_s:
                                        ; 12/12/2022
00003AF8 F6C120                         test    cl,flagseclim ; 20h
                                        ;test   cx,flagseclim ; 20h
00003AFB 7404                           jz      short try_h
00003AFD A3[1A4F]                       mov     [slim],ax
                                        ; 02/11/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
                                        ;jmp    short done_ret
                                        ; 12/12/2022
                                        ; cf=0
00003B00 C3                             retn    ; 13/05/2019

                                    ; must be for number of heads

                                    try_h:
00003B01 A3[184F]                       mov     [hlim],ax
                                    done_ret:
                                        ;clc
                                        ; 02/11/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
                                        ; 12/12/2022
                                        ; cf=0 (test instruction resets cf)
                                        ;clc
00003B04 C3                             retn


                                    ; 16/04/2024 - Retro DOS v5.0
                                    ; 03/01/2023 - Retro DOS v4.2
                                    %if 1

                                    ; 15/04/2019 - Retro DOS v4.0

                                    ;--------------------------------------------------------------------------
                                    ;
                                    ; procedure : parseline
                                    ;
                                    ; entry point is parseline. al contains the first character in command line.
                                    ;
                                    ;--------------------------------------------------------------------------

                                        ; 16/04/2024 - Retrodos v5.0 (Modified PCDOS 7.1 IBMBIO.COM)
```

```
                                    ; (PCDOS 7.1 IBMBIO.COM - SYSINIT:4151h)
                                    ;
                                    ; 03/01/2023 - Retrodos v4.2 (Modified MSDOS 6.21 IO.SYS)
                                    ; (SYSINIT:3EDFh)
                                    ;
                                    ; 01/11/2022 - Retrodos v4.0 (Modified MSDOS 5.0 IO.SYS)
                                    ; (SYSINIT:30ACh)
                                    ;
                                parseline:
                                    ; 03/01/2023
                                    ; ds = cs ; *
                                    ;
                                    ;push  ds ; *
                                    ;
                                    ;push  cs ; *
                                    ;pop   ds ; *
                                    ;
                                nextswtch:
00003B05 3C0D                       cmp     al,cr                   ; carriage return?
00003B07 741C                       je      short done_line
00003B09 3C0A                       cmp     al,lf                   ; linefeed?
00003B0B 7421                       je      short put_back          ; put it back and done
                                    ;
                                ; anything less or equal to a space is ignored.
                                    ;
00003B0D 3C20                       cmp     al,' '                    ; space?
00003B0F 760F                       jbe     short getnext           ; skip over space
00003B11 3C2F                       cmp     al,'/'
00003B13 7402                       je      short getparm
00003B15 F9                         stc                             ; mark error invalid-character-in-input
                                    ;jmp    short exitpl
                                    ; 03/01/2023
                                swterr:
00003B16 C3                         retn
                                    ;
                                getparm:
00003B17 E86EFF                     call    check_switch
00003B1A 891E[1D4F]                 mov     [switches],bx           ; save switches read so far
00003B1E 72F6                       jc      short swterr
                                getnext:
00003B20 E88C0C                     call    getchr
                                    ;jc     short done_line
                                    ;jmp    short nextswtch
                                    ; 03/01/2023
00003B23 73E0                       jnc     short nextswtch
                                ;swterr:
                                    ;jmp    short exitpl            ; exit if error
                                    ;
                                done_line:
                                    ; 12/12/2022
00003B25 F606[1D4F]08               test    byte [switches],flagdrive ; 8
                                    ;test   word [switches],flagdrive ; 8 ; see if drive specified
00003B2A 750C                       jnz     short okay
00003B2C F9                         stc                             ; mark error no-drive-specified
                                    ;jmp    short exitpl
                                    ; 03/01/2023
00003B2D C3                         retn
                                    ;
                                ;exitpl:
                                    ; 03/01/2023
                                    ; ds = cs
                                    ;;pop   ds ; *
                                    ;retn
                                    ;
                                put_back:
00003B2E FF06[5603]                 inc     word [count]            ; one more char to scan
00003B32 FF0E[5A03]                 dec     word [chrptr]           ; back up over linefeed
00003B36 EBED                       jmp     short done_line
                                    ;
                                okay:
00003B38 A1[1D4F]                   mov     ax,[switches]
00003B3B 83E003                     and     ax,0003h            ; get flag bits for changeline and non-rem
00003B3E A3[C04D]                   mov     [deviceparameters+A_DEVICEPARAMETERS.DP_DEVICEATTRIBUTES],ax
                                    ; 16/04/2024
                                    ;mov    word [deviceparameters+A_DEVICEPARAMETERS.DP_TRACKTABLEENTRIES],0
                                    ;;;
00003B41 C706[1A4E]0000             mov     word [deviceparameters+92],0 ; PCDOS 7.1 IBMBIO.COM
                                    ;;;
                                    ;clc                            ; everything is fine
                                    ; 01/11/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
                                    ; 12/12/2022
                                    ; cf=0
                                    ;clc
                                    ;call  setdeviceparameters
                                    ; 03/01/2023
                                    ;jmp    short setdeviceparameters
                                    ;
                                %endif
                                    ;
                                ;   M047 -- Begin modifications (too numerous to mark specifically)
                                    ;
                                ;--------------------------------------------------------------------------
                                ;
                                ; procedure : setdeviceparameters
                                ;
                                ; setdeviceparameters sets up the recommended bpb in each bds in the
                                ; system based on the form factor. it is assumed that the bpbs for the
                                ; various form factors are present in the bpbtable. for hard files,
                                ; the recommended bpb is the same as the bpb on the drive.
                                ; no attempt is made to preserve registers since we are going to jump to
                                ; sysinit straight after this routine.
                                ;
                                ;   if we return carry, the DRIVPARM will be aborted, but presently
                                ;     we always return no carry
                                ;
                                ;   note:  there is a routine by the same name in msdioctl.asm
                                ;
                                ;--------------------------------------------------------------------------
                                    ;
                                ; 15/04/2019 - Retro DOS v4.0
                                    ;
                                    ; 02/11/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 IO.SYS)
                                    ;
                                    ; 03/01/2023 - Retro DOS v4.2 (Modified MSDOS 6.21 IO.SYS)
                                    ; (SYSINIT:3FC4h)
                                    ;
                                    ; 09/09/2023 - Retro DOS v4.2 IO.SYS (Optimization)
                                    ; (PCDOS 7.1 IBMBIO.COM - SYSINIT:4236h)
                                    ;
                                setdeviceparameters:
                                    ; 03/01/2023
                                    ; ds = cs
                                    ;
00003B47 06                         push    es
```

```
40520
40521 00003B48 0E                         push    cs
40522 00003B49 07                         pop     es
40523
40524 00003B4A 31DB                       xor     bx,bx
40525 00003B4C 8A1E[BF4D]                 mov     bl,[deviceparameters+A_DEVICEPARAMETERS.DP_DEVICETYPE]
40526 00003B50 80FB00                     cmp     bl,DEV_5INCH ; 0
40527 00003B53 7506                       jne     short got_80
40528
40529 00003B55 C706[C24D]2800             mov     word [deviceparameters+A_DEVICEPARAMETERS.DP_CYLINDERS],40
40530                                                                   ; 48 tpi=40 cyl
40531                                   got_80:
40532 00003B5B D1E3                       shl     bx,1                  ; get index into bpb table
40533 00003B5D 8BB7[2E50]                 mov     si,[bpbtable+bx]      ; get address of bpb
40534
40535                                     ;mov    di,deviceparameters+7
40536                                     ; 02/11/2022
40537 00003B61 BF[C54D]                   mov     di,deviceparameters+A_DEVICEPARAMETERS.DP_BPB ; es:di -> bpb
40538 00003B64 B93B00                     mov     cx,A_BPB.size ; 31
40539                                     ; 09/09/2023
40540                                     ;mov    cx,59 ; PCDOS 7.1 IBMBIO.COM A_BPB.size
40541 00003B67 FC                         cld
40542                                     ;repe   movsb
40543                                     ; 02/11/2022
40544 00003B68 F3A4                       rep     movsb
40545
40546 00003B6A 07                         pop     es
40547
40548                                     ; 12/12/2022
40549 00003B6B F606[1D4F]20               test    byte [switches],flagseclim ; 20h
40550                                     ;test   word [switches],flagseclim ; 20h
40551 00003B70 7406                       jz      short see_heads
40552
40553 00003B72 A1[1A4F]                   mov     ax,[slim]
40554                                     ;mov    [deviceparameters+20],ax
40555 00003B75 A3[D24D]                   mov     [deviceparameters+A_DEVICEPARAMETERS.DP_BPB+A_BPB.SECTORSPERTRACK],ax
40556
40557                                   see_heads:
40558                                     ; 12/12/2022
40559 00003B78 F606[1D4F]40               test    byte [switches],flagheads ; 40h
40560                                     ;test   word [switches],flagheads ; 40h
40561 00003B7D 7406                       jz      short heads_not_altered
40562
40563 00003B7F A1[184F]                   mov     ax,[hlim]
40564                                     ;mov    [deviceparameters+22],ax
40565 00003B82 A3[D44D]                   mov     [deviceparameters+A_DEVICEPARAMETERS.DP_BPB+A_BPB.HEADS],ax
40566
40567                                   heads_not_altered:
40568
40569                                   ; set up correct media descriptor byte and sectors/cluster
40570                                   ;   sectors/cluster is always 2 except for any one sided disk or 1.44M
40571
40572                                     ;mov    byte [deviceparameters+9],2
40573                                     ; 02/11/2022
40574                                     ;mov    byte [deviceparameters+A_DEVICEPARAMETERS.DP_BPB+A_BPB.SECTORSPERCLUSTER],2
40575                                     ; 03/01/2023
40576 00003B85 B80200                     mov     ax,2
40577 00003B88 A2[C74D]                   mov     [deviceparameters+A_DEVICEPARAMETERS.DP_BPB+A_BPB.SECTORSPERCLUSTER],al ; 2
40578
40579 00003B8B B3F0                       mov     bl,0F0h               ; get default mediabyte
40580
40581                                   ;   preload the mediadescriptor from the bpb into bh for convenient access
40582
40583                                     ;mov    bh,[deviceparameters+17]
40584                                     ; 02/11/2022
40585 00003B8D 8A3E[CF4D]                 mov     bh,[deviceparameters+A_DEVICEPARAMETERS.DP_BPB+A_BPB.MEDIADESCRIPTOR]
40586
40587                                     ; 03/01/2023
40588                                     ; ax = 2
40589 00003B91 3906[D44D]                 cmp     [deviceparameters+A_DEVICEPARAMETERS.DP_BPB+A_BPB.HEADS],ax ; >2 heads?
40590                                     ;cmp    word [deviceparameters+A_DEVICEPARAMETERS.DP_BPB+A_BPB.HEADS],2 ; >2 heads?
40591 00003B95 773C                       ja      short got_correct_mediad ; just use default if heads>2
40592
40593 00003B97 7524                       jne     short only_one_head   ; one head, do one head stuff
40594
40595                                   ;   two head drives will use the mediadescriptor from the bpb
40596
40597 00003B99 88FB                       mov     bl,bh                 ; get mediadescriptor from bpb
40598
40599                                   ;   two sided drives have two special cases to look for. One is
40600                                   ;      a 320K diskette (40 tracks, 8 secs per track). It uses
40601                                   ;      a mediaid of 0fch. The other is 1.44M, which uses only
40602                                   ;      one sector/cluster.
40603
40604                                   ;   any drive with 18secs/trk, 2 heads, 80 tracks, will be assumed
40605                                   ;      to be a 1.44M and use only 1 sector per cluster. Any other
40606                                   ;      type of 2 headed drive is all set.
40607
40608 00003B9B 833E[D24D]12                cmp     word [deviceparameters+A_DEVICEPARAMETERS.DP_BPB+A_BPB.SECTORSPERTRACK],18
40609 00003BA0 7509                       jne     short not_144m
40610 00003BA2 833E[C24D]50               cmp     word [deviceparameters+A_DEVICEPARAMETERS.DP_CYLINDERS],80
40611 00003BA7 7502                       jne     short not_144m
40612
40613                                   ;   We've got cyl=80, heads=2, secpertrack=18. Set cluster size to 1.
40614
40615 00003BA9 EB24                       jmp     short got_one_secperclus_drive
40616
40617                                   ;   check for 320K
40618
40619                                   not_144m:
40620 00003BAB 833E[C24D]28               cmp     word [deviceparameters+A_DEVICEPARAMETERS.DP_CYLINDERS],40
40621 00003BB0 7521                       jne     short got_correct_mediad
40622 00003BB2 833E[D24D]08               cmp     word [deviceparameters+A_DEVICEPARAMETERS.DP_BPB+A_BPB.SECTORSPERTRACK],8
40623 00003BB7 751A                       jne     short got_correct_mediad
40624
40625 00003BB9 B3FC                       mov     bl,0FCh
40626 00003BBB EB16                       jmp     short got_correct_mediad
40627
40628                                   only_one_head:
40629
40630                                   ;   if we don't have a 360K drive, then just go use 0f0h as media descr.
40631
40632 00003BBD 803E[BF4D]00               cmp     byte [deviceparameters+A_DEVICEPARAMETERS.DP_DEVICETYPE],DEV_5INCH ; 0
40633 00003BC2 740B                       je      short got_one_secperclus_drive
40634
40635                                   ;   single sided 360K drive uses either 0fch or 0feh, depending on
40636                                   ;     whether sectorspertrack is 8 or 9. For our purposes, anything
40637                                   ;     besides 8 will be considered 0fch
40638
40639 00003BC4 B3FC                       mov     bl,0FCh               ; single sided 9 sector media id
40640 00003BC6 833E[D24D]08               cmp     word [deviceparameters+A_DEVICEPARAMETERS.DP_BPB+A_BPB.SECTORSPERTRACK],8
40641                                     ; 12/12/2022
40642 00003BCB 7502                       jne     short got_one_secperclus_drive ; okay if anything besides 8
40643
```

```
                            
00003BCD B3FE                       mov     bl,0FEh             ; 160K mediaid
                            
                            ;   we've either got a one sided drive, or a 1.44M drive
                            ;     either case we'll use 1 sector per cluster instead of 2
                            
                            got_one_secperclus_drive:
                                ; 03/01/2023
                                ; ax = 2
00003BCF 48                         dec     ax  ; ax = 1
00003BD0 A2[C74D]                   mov     [deviceparameters+A_DEVICEPARAMETERS.DP_BPB+A_BPB.SECTORSPERCLUSTER],al ; 1
                                    ;mov    byte [deviceparameters+A_DEVICEPARAMETERS.DP_BPB+A_BPB.SECTORSPERCLUSTER],1
                            
                            got_correct_mediad:
00003BD3 881E[CF4D]                 mov     [deviceparameters+A_DEVICEPARAMETERS.DP_BPB+A_BPB.MEDIADESCRIPTOR],bl
                            
                            ;    Calculate the correct number of Total Sectors on medium
                            
00003BD7 A1[C24D]                   mov     ax,[deviceparameters+A_DEVICEPARAMETERS.DP_CYLINDERS]
00003BDA F726[D44D]                 mul     word [deviceparameters+A_DEVICEPARAMETERS.DP_BPB+A_BPB.HEADS]
00003BDE F726[D24D]                 mul     word [deviceparameters+A_DEVICEPARAMETERS.DP_BPB+A_BPB.SECTORSPERTRACK]
00003BE2 A3[CD4D]                   mov     word [deviceparameters+A_DEVICEPARAMETERS.DP_BPB+A_BPB.TOTALSECTORS],ax
00003BE5 F8                         clc                         ; we currently return no errors
                            
00003BE6 C3                         retn
                            
                            ;   M047 -- end rewritten routine
                            
                            ;--------------------------------------------------------------------------
                            ;
                            ; procedure : organize
                            ;
                            ;--------------------------------------------------------------------------
                            
                            ; 09/09/2023 - Retrodos v4.2 IO.SYS (Optimization)
                            %if 1
                            end_commd_line:
00003BE7 AA                         stosb                       ; store line feed char in buffer for the linecount.
                                    ;mov    byte [cs:com_level],0 ; reset the command level.
                                    ; 03/01/2023
                                    ; ds = cs
                                    ;mov    byte [com_level],0
                                    ;jmp    short org1
                                    ; 09/09/2023
00003BE8 EB0E                       jmp     short org0
                            nochar1:
00003BEA F9                         stc
00003BEB C3                         retn
                            %endif
                                    ; 02/11/2022 - Retrodos v4.0 (Modified MSDOS 5.0 IO.SYS)
                                    ; (SYSINIT:3234h)
                            
                                    ; 03/01/2023 - Retrodos v4.2 (Modified MSDOS 6.21 IO.SYS)
                                    ; (SYSINIT:4067h)
                            
                                    ; 09/09/2023 - Retrodos v4.2 IO.SYS (Optimization)
                                    ; (PCDOS 7.1 IBMBIO.COM - SYSINIT:42D9h)
                            
                            organize:
                                    ; 03/01/2023
                                    ; ds = cs
00003BEC 8B0E[5603]                 mov     cx,[count]
                                    ;mov    cx,[cs:count]
00003BF0 E3F8                       jcxz    nochar1
                            
                            ;ifndef     MULTI_CONFIG
                            ;
                            ;;   In MULTI_CONFIG, we map to upper case on a line-by-line basis,
                            ;;   because we the case of values in SET commands preserved
                            ;
                            ;   call    mapcase
                            ;endif
                                    ; 02/11/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
                                    ; 03/01/2023 - Retro DOS v4.2
                                    ;call   mapcase
                            
00003BF2 31F6                       xor     si,si
00003BF4 89F7                       mov     di,si
00003BF6 31C0                       xor     ax,ax
                                    ; 02/11/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
                                    ;;mov   byte [cs:com_level],0
                                    ; 12/12/2022
                                    ;mov    [cs:com_level],al ; 0
                                    ; 03/01/2023
                                    ; ds = cs
                                    ; 09/09/2023
                                    ;mov    [com_level],al ; 0
                            org0:
00003BF8 C606[5003]00               mov     byte [com_level],0 ; 09/09/2023
                            org1:
00003BFD E8EF01                     call    skip_comment
00003C00 74E5                       jz      short end_commd_line  ; found a comment string and skipped.
00003C02 E8D001                     call    get2                ; not a comment string. then get a char.
00003C05 3C0A                       cmp     al,lf ; 0Ah
00003C07 74DE                       je      short end_commd_line  ; starts with a blank line.
00003C09 3C20                       cmp     al,' ' ; 20h
00003C0B 76F0                       jbe     short org1          ; skip leading control characters
                                    ; 09/09/2023
                                    ;jmp    short findit
                            
                            ; 09/09/2023
                            %if 0
                            end_commd_line:
                                stosb                           ; store line feed char in buffer for the linecount.
                                ;mov    byte [cs:com_level],0 ; reset the command level.
                                ; 03/01/2023
                                ; ds = cs
                                mov     byte [com_level],0
                                jmp     short org1
                            
                            nochar1:
                                stc
                                retn
                            %endif
                            
                            findit:
00003C0D 51                         push    cx
00003C0E 56                         push    si
00003C0F 57                         push    di
00003C10 89F5                       mov     bp,si
00003C12 4D                         dec     bp
00003C13 BE[D24C]                       mov     si,comtab       ; prepare to search command table
00003C16 B500                       mov     ch,0
                            findcom:
00003C18 89EF                       mov     di,bp
```

```
40768 00003C1A 8A0C                     mov     cl,[si]
40769 00003C1C 46                       inc     si
40770 00003C1D E345                     jcxz    nocom
40771
40772                                   ; 02/11/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
40773
40774                                 ; 03/01/2023 - Retrodos v4.2 (Modified MSDOS 6.21 IO.SYS)
40775
40776                                 ;ifdef      MULTI_CONFIG
40777
40778                                 ;   Simplify future parsing by collapsing ";" onto "REM", and at the same
40779                                 ;   time skip the upcoming delimiter test (since ";" need not be followed by
40780                                 ;   anything in particular)
40781
40782 00003C1F 26803D3B                   cmp     byte [es:di],CONFIG_SEMICOLON  ; ';'
40783 00003C23 7430                     je      short semicolon
40784                                 loopcom:
40785                                   ;mov    al,[es:di]
40786                                   ;inc    di
40787                                   ;and    al,~20h ; 0DFh        ; force upper case
40788                                   ;inc    si                      ; compare to byte @es:di
40789                                   ;cmp    al,[si-1]
40790                                   ; 28/07/2023 - Retro DOS v4.2 IO.SYS (optimization)
40791 00003C25 268A25                   mov     ah,[es:di]
40792 00003C28 47                       inc     di
40793 00003C29 80E4DF                   and     ah,~20h ; 0DFh
40794 00003C2C AC                       lodsb                   ; mov al,[si]
40795                                                           ; inc si
40796                                   ;cmp    al,ah
40797                                   ;loope  loopcom
40798                                   ; 28/07/2023
40799 00003C2D 30C4                     xor     ah,al           ; result: ah = 0 (*) if ah = al
40800 00003C2F E1F4                     loopz   loopcom
40801                                 ;else
40802                                 ;   repe    cmpsb
40803                                 ;endif
40804                                   ; 02/11/2022
40805                                   ; 03/01/2023 - Retro DOS v4.2
40806                                   ;repe   cmpsb
40807
40808                                   ; 28/07/2023
40809                                   ;lahf
40810 00003C31 01CE                       add     si,cx                 ; bump to next position without affecting flags
40811                                   ;sahf
40812 00003C33 AC                         lodsb                         ; get indicator letter
40813                                   ;jnz    short findcom
40814                                     ; 28/07/2023
40815 00003C34 08E4                     or      ah,ah                 ; (*)
40816 00003C36 75E0                     jnz     short findcom
40817
40818 00003C38 26803D0D                 cmp     byte [es:di],cr               ; the next char might be cr,lf
40819 00003C3C 7421                     je      short gotcom0         ; such as in "rem",cr,lf case.
40820 00003C3E 26803D0A                 cmp     byte [es:di],lf
40821 00003C42 741B                     je      short gotcom0
40822
40823                                   ; 02/11/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
40824
40825                                 ; 03/01/2023 - Retrodos v4.2 (Modified MSDOS 6.21 IO.SYS)
40826
40827                                 ;ifdef      MULTI_CONFIG
40828
40829                                 ; Skip the delimiter test for the BEGIN identifier (it doesn't have one).
40830
40831 00003C44 3C5B                     cmp     al,CONFIG_BEGIN  ; '['
40832 00003C46 7417                     je      short gotcom0
40833                                 ;endif
40834 00003C48 50                       push    ax
40835 00003C49 268A05                   mov     al,[es:di]            ; now the next char. should be a delim.
40836
40837                                 ;ifdef      MULTI_CONFIG
40838
40839                                 ;   If keyword is *immediately* followed by a question mark (?), then
40840                                 ;   set the high bit of the ASCII command code (CONFIG_OPTION_QUERY) that is
40841                                 ;   stored in the CONFIG.SYS memory image.
40842
40843 00003C4C 3C3F                     cmp     al,'?'                  ; explicit interactive command?
40844 00003C4E 7509                     jne     short no_query        ; no
40845 00003C50 58                       pop     ax                      ; yes, so retrieve the original code
40846                                   ;or     al,80h ; 03/01/2023
40847 00003C51 0C80                     or      al,CONFIG_OPTION_QUERY  ; and set the QUERY bit
40848 00003C53 EB0A                     jmp     short gotcom0           ;
40849                                 semicolon:
40850 00003C55 B030                     mov     al,CONFIG_REM ; '0'
40851 00003C57 EB06                     jmp     short gotcom0
40852                                 no_query:
40853                                 ;endif  ;MULTI_CONFIG
40854
40855                                   ; 02/11/2022
40856                                   ; 03/01/2023 - Retro DOS v4.2
40857                                   ;push   ax
40858                                   ;mov    al,[es:di]            ; now the next char. should be a delim.
40859
40860 00003C59 E82E0B                     call    delim
40861                                 no_delim:
40862 00003C5C 58                       pop     ax
40863 00003C5D 75B9                     jnz     short findcom
40864                                 gotcom0:
40865 00003C5F 5F                       pop     di
40866 00003C60 5E                       pop     si
40867 00003C61 59                       pop     cx
40868 00003C62 EB10                     jmp     short gotcom
40869                                 nocom:
40870 00003C64 5F                       pop     di
40871 00003C65 5E                       pop     si
40872 00003C66 59                       pop     cx
40873 00003C67 B05A                         mov     al,CONFIG_UNKNOWN  ; 'Z'
40874 00003C69 AA                       stosb                         ; save indicator char.
40875                                 _skipline:
40876 00003C6A E86801                   call    get2
40877 00003C6D 3C0A                     cmp     al,lf ; 0Ah           ; skip this bad command line
40878 00003C6F 75F9                       jne     short _skipline
40879                                   ;jmp    short end_commd_line  ; handle next command line
40880                                   ; 09/09/2023
40881 00003C71 E973FF                   jmp     end_commd_line
40882                                 gotcom:
40883 00003C74 AA                         stosb                               ; save indicator char in buffer
40884
40885                                   ; 02/11/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
40886
40887                                 ; 03/01/2023 - Retrodos v4.2 (Modified MSDOS 6.21 IO.SYS)
40888
40889                                 ;ifdef      MULTI_CONFIG
40890
40891                                 ;   Don't pollute "cmd_indicator" with the CONFIG_OPTION_QUERY bit though;
```

```
                                ;   it screws up the direct comparisons below.
                                
00003C75 247F                       and     al,~CONFIG_OPTION_QUERY ; 7Fh
                                ;endif
                                    ;mov    [cs:cmd_indicator],al ; save it for the future use.
                                    ; 03/01/2023
                                    ; ds = cs
00003C77 A2[5403]                   mov     [cmd_indicator],al    ; save it for the future use.
                                
                                ;ifdef      MULTI_CONFIG
                                
                                ;   There is no whitespace/delimiter between the "begin block" character
                                ;   ([) and the name of block (eg, [menu]), therefore skip this delimiter
                                ;   skipping code
                                
00003C7A 3C5B                       cmp     al,CONFIG_BEGIN
00003C7C 7455                       je      short org31
00003C7E 3C4F                       cmp     al,CONFIG_SUBMENU ; 'O'
00003C80 740F                       je      short no_mapcase
00003C82 3C45                       cmp     al,CONFIG_MENUITEM ; 'E'
00003C84 740B                       je      short no_mapcase
00003C86 3C41                       cmp     al,CONFIG_MENUDEFAULT ; 'A'
00003C88 7407                       je      short no_mapcase
00003C8A 3C4A                       cmp     al,CONFIG_INCLUDE ; 'J'
00003C8C 7403                       je      short no_mapcase
00003C8E E8350B                     call    mapcase               ; map case of rest of line to UPPER
                                no_mapcase:
                                ;endif
                                    ;; 02/11/2022
                                    ;;mov   [cs:cmd_indicator],al ; save it for the future use.
                                    ;; 03/01/2023
                                    ;; ds = cs
                                    ;mov    [cmd_indicator],al
                                org2:
00003C91 E84101                     call    get2                    ; skip the command name until delimiter
                                        cmp     al,lf           ; 0Ah
00003C94 3C0A                       je      short org21
00003C96 740F                       cmp     al,cr       ; 0Dh
00003C98 3C0D                       je      short org21
00003C9A 740B                       ; 02/11/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
                                    ; 03/01/2023 - Retro DOS v4.2
00003C9C 3C2F                       cmp     al, '/'             ; T-RICHJ: Added to allow DEVHIGH/L:...
00003C9E 7407                       je      short org21         ; T-RICHJ: to be parsed properly.
                                
00003CA0 E8E70A                     call    delim
                                        jnz short org2
00003CA5 EB02                       jmp     short org3
                                org21:                                  ;if cr or lf then
00003CA7 4E                         dec     si                  ; undo si, cx register
00003CA8 41                         inc     cx                  ;  and continue
                                org3:
                                    ;cmp    byte [cs:cmd_indicator],CONFIG_COMMENT ; 'Y'
                                    ;je     short get_cmt_token
                                    ; 02/11/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
                                    ; 03/01/2023 - Retro DOS v4.2
                                    ;cmp    byte [cs:cmd_indicator],CONFIG_DEVICE ; 'D'
                                    ;je     short org_file
                                    ;cmp    byte [cs:cmd_indicator],CONFIG_INSTALL ; 'I'
                                    ;je     short org_file
                                    ;cmp    byte [cs:cmd_indicator],CONFIG_INSTALLHIGH ; 'W'
                                    ;je     short org_file
                                    ; 02/11/2022
                                    ; 03/01/2023 - Retro DOS v4.2
                                    ;;cmp   byte [cs:cmd_indicator],CONFIG_DEVICE ; 'D'
                                    ;;je    short org_file
                                    ;cmp    byte [cs:cmd_indicator],CONFIG_SHELL ; 'S'
                                    ;je     short org_file
                                        ;cmp        byte [cs:cmd_indicator],CONFIG_SWITCHES ; '1'
                                    ;je     short org_switch
                                
                                    ; 03/01/2023
                                    ; ds = cs
                                
00003CA9 803E[5403]59               cmp     byte [cmd_indicator],CONFIG_COMMENT ; 'Y'
00003CAE 745D                       je      short get_cmt_token
                                    ; 02/11/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
                                    ; 03/01/2023 - Retro DOS v4.2
00003CB0 803E[5403]44               cmp     byte [cmd_indicator],CONFIG_DEVICE ; 'D'
00003CB5 7430                       je      short org_file
00003CB7 803E[5403]49               cmp     byte [cmd_indicator],CONFIG_INSTALL ; 'I'
00003CBC 7429                       je      short org_file
00003CBE 803E[5403]57               cmp     byte [cmd_indicator],CONFIG_INSTALLHIGH ; 'W'
00003CC3 7422                       je      short org_file
                                    ; 02/11/2022
                                    ; 03/01/2023 - Retro DOS v4.2
                                    ;cmp    byte [cmd_indicator],CONFIG_DEVICE ; 'D'
                                    ;je     short org_file
00003CC5 803E[5403]53                   cmp byte [cmd_indicator],CONFIG_SHELL ; 'S'
00003CCA 741B                       je      short org_file
00003CCC 803E[5403]31                   cmp byte [cmd_indicator],CONFIG_SWITCHES ; '1'
00003CD1 7403                       je      short org_switch
                                
                                org31:
00003CD3 E99500                     jmp     org4
                                
                                org_switch:
00003CD6 E81601                     call    skip_comment
00003CD9 7472                       jz      short end_commd_line_brdg
                                
00003CDB E8F700                     call    get2
00003CDE E8B10A                     call    org_delim
00003CE1 74F3                       jz      short org_switch
                                
00003CE3 AA                         stosb
00003CE4 E99300                     jmp     org5
                                
                                org_file:                   ; get the filename and put 0 at end
00003CE7 E80501                     call    skip_comment
00003CEA 7464                       jz      short org_put_zero
                                
00003CEC E8E600                     call    get2            ; not a comment
00003CEF E8980A                     call    delim
00003CF2 74F3                       jz      short org_file ; skip the possible delimiters
                                
00003CF4 AA                         stosb                   ; copy the first non delim char found in buffer
                                
                                org_copy_file:
00003CF5 E8F700                     call    skip_comment    ; comment char in the filename?
00003CF8 7456                       jz      short org_put_zero ; then stop copying filename at that point
                                
00003CFA E8D800                     call    get2
00003CFD 3C2F                       cmp     al,'/'          ; a switch char? (device=filename/xxx)
00003CFF 7457                       je      short end_file_slash ; this will be the special case.
                                
```

```
41016 00003D01 AA                       stosb                   ; save the char. in buffer
41017 00003D02 E8850A                   call    delim
41018 00003D05 7459                     jz      short end_copy_file
41019
41020 00003D07 3C20                     cmp     al, ' '
41021 00003D09 77EA                     ja      short org_copy_file ; keep copying
41022 00003D0B EB53                     jmp     short end_copy_file ; otherwise, assume end of the filename.
41023
41024                                 get_cmt_token:                  ; get the token. just max. 2 char.
41025 00003D0D E8C500                   call    get2
41026 00003D10 3C20                     cmp     al,' '          ; skip white spaces or "=" char.
41027 00003D12 74F9                     je      short get_cmt_token ; (we are allowing the other special
41028 00003D14 3C09                     cmp     al,tab ; 9      ;  characters can used for comment id.
41029 00003D16 74F5                     je      short get_cmt_token ;  character.)
41030 00003D18 3C3D                     cmp     al,'='          ; = is special in this case.
41031 00003D1A 74F1                     je      short get_cmt_token
41032 00003D1C 3C0D                     cmp     al,cr
41033 00003D1E 7426                     je      short get_cmt_end ; cannot accept the carriage return
41034 00003D20 3C0A                     cmp     al,lf
41035 00003D22 7422                     je      short get_cmt_end
41036
41037                                   ; 03/01/2023
41038                                   ; ds = cs
41039                                   ;mov    [cs:cmmt1],al  ; store it
41040                                   ;mov    byte [cs:cmmt],1 ; 1 char. so far.
41041 00003D24 A2[5203]                 mov     [cmmt1],al      ; store it
41042 00003D27 C606[5103]01             mov     byte [cmmt],1  ; 1 char. so far.
41043 00003D2C E8A600                   call    get2
41044 00003D2F 3C20                     cmp     al,' ' ; 20h
41045 00003D31 7413                     je      short get_cmt_end
41046 00003D33 3C09                     cmp     al,tab ; 9
41047 00003D35 740F                     je      short get_cmt_end
41048 00003D37 3C0D                     cmp     al,cr  ; 0Dh
41049 00003D39 740B                     je      short get_cmt_end
41050 00003D3B 3C0A                     cmp     al,lf  ; 0Ah
41051 00003D3D 740E                     je      short end_commd_line_brdg
41052
41053                                   ;mov    [cs:cmmt2],al
41054                                   ;inc    byte [cs:cmmt]
41055                                   ; 03/01/2023
41056 00003D3F A2[5303]                 mov     [cmmt2],al
41057 00003D42 FE06[5103]               inc     byte [cmmt]
41058
41059                                 get_cmt_end:
41060 00003D46 E88C00                   call    get2
41061 00003D49 3C0A                     cmp     al,lf
41062 00003D4B 75F9                     jne     short get_cmt_end     ; skip it.
41063                                 end_commd_line_brdg:
41064 00003D4D E997FE                   jmp     end_commd_line        ; else jmp to end_commd_line
41065
41066                                 org_put_zero:                           ; make the filename in front of
41067 00003D50 26C60500                 mov     byte [es:di],0        ;  the comment string to be an asciiz.
41068 00003D54 47                       inc     di
41069 00003D55 E98FFE                   jmp     end_commd_line        ;  (maybe null if device=/*)
41070
41071                                 end_file_slash:                 ; al = "/" option char.
41072 00003D58 26C60500                 mov     byte [es:di],0        ; make a filename an asciiz
41073 00003D5C 47                       inc     di                    ; and
41074 00003D5D AA                       stosb                         ; store "/" after that.
41075 00003D5E EB1A                     jmp     short org5            ; continue with the rest of the line
41076
41077                                 end_copy_file:
41078 00003D60 26C645FF00               mov     byte [es:di-1],0      ; make it an asciiz and handle the next char.
41079 00003D65 3C0A                     cmp     al,lf
41080 00003D67 74E4                     je      short end_commd_line_brdg
41081 00003D69 EB0F                     jmp     short org5
41082
41083                                 org4:                                   ; org4 skips all delimiters after the command name except for '/'
41084 00003D6B E88100                   call    skip_comment
41085 00003D6E 74DD                     jz      short end_commd_line_brdg
41086
41087 00003D70 E86200                   call    get2
41088 00003D73 E81C0A                   call    org_delim             ; skip delimiters except '/' (mrw 4/88)
41089 00003D76 74F3                     jz      short org4
41090 00003D78 EB08                     jmp     short org51
41091
41092                                 org5:                                   ; rest of the line
41093 00003D7A E87200                   call    skip_comment          ; comment?
41094 00003D7D 74CE                     jz      short end_commd_line_brdg
41095 00003D7F E85300                   call    get2                  ; not a comment.
41096
41097                                 org51:
41098 00003D82 AA                       stosb                         ; copy the character
41099 00003D83 3C22                     cmp     al,'"'  ; 22h         ; a quote ?
41100 00003D85 743A                     je      short at_quote
41101 00003D87 3C20                     cmp     al,' '  ; 20h
41102 00003D89 77EF                     ja      short org5
41103
41104                                   ; 09/09/2023
41105                                   ; (Note: PCDOS 7.1 IBMBIO.COM does not contain M051 modification)
41106
41107                                                                   ; M051 - Start
41108                                   ; 03/01/2023
41109                                   ; ds = cs
41110 00003D8B 803E[5403]55                 cmp byte [cmd_indicator],CONFIG_DEVICEHIGH
41111                                   ;cmp    byte [cs:cmd_indicator],CONFIG_DEVICEHIGH ; Q: is this devicehigh
41112 00003D90 7514                     jne     short not_dh          ; N:
41113 00003D92 3C0A                     cmp     al,lf                 ; Q: is this line feed
41114 00003D94 7416                     je      short org_dhlf        ; Y: stuff a blank before the lf
41115 00003D96 3C0D                     cmp     al,cr                 ; Q: is this a cr
41116 00003D98 75E0                     jne     short org5            ; N:
41117 00003D9A 26C645FF20               mov     byte [es:di-1],' '    ; overwrite cr with blank
41118 00003D9F AA                       stosb                         ; put cr after blank
41119 00003DA0 FE06[223A]               inc     byte [insert_blank]
41120                                   ;inc    byte [cs:insert_blank]; indicate that blank has been
41121                                                                 ; inserted
41122 00003DA4 EBD4                     jmp     short org5
41123                                 not_dh:                                 ; M051 - End
41124
41125 00003DA6 3C0A                     cmp     al,lf                 ; line feed?
41126 00003DA8 740F                     je      short org1_brdg               ; handles the next command line.
41127 00003DAA EBCE                     jmp     short org5            ; handles next char in this line.
41128
41129                                 org_dhlf:                               ; M051 - Start
41130                                   ; 03/01/2023
41131                                   ; ds = cs
41132 00003DAC 803E[223A]01             cmp     byte [insert_blank],1
41133                                   ;cmp    byte [cs:insert_blank],1 ; Q:has a blank already been inserted
41134 00003DB1 7406                     je      short org1_brdg               ; Y:
41135 00003DB3 26C645FF20               mov     byte [es:di-1],' '    ; overwrite lf with blank
41136 00003DB8 AA                       stosb                         ; put lf after blank
41137                                                                 ; M051 - End
41138                                 org1_brdg:
41139 00003DB9 C606[223A]00             mov     byte [insert_blank],0
```

```
                                    ;mov   byte [cs:insert_blank],0 ; M051: clear blank indicator for
                                                                    ; M051: devicehigh
00003DBE E93CFE                     jmp    org1

                                at_quote:
00003DC1 803E[5003]00               cmp    byte [com_level],0
                                    ;cmp   byte [cs:com_level],0
00003DC6 7407                       je     short up_level
                                    ;mov   byte [cs:com_level],0 ; reset it.
00003DC8 C606[5003]00               mov    byte [com_level],0
00003DCD EBAB                       jmp    short org5

                                up_level:
                                    ;inc   byte [cs:com_level]   ; set it.
00003DCF FE06[5003]                 inc    byte [com_level]
00003DD3 EBA5                       jmp    short org5

                                ;--------------------------------------------------------------------------------
                                ;
                                ; procedure : get2
                                ;
                                ;--------------------------------------------------------------------------------

                                    ; 02/11/2022 - Retrodos v4.0 (Modified MSDOS 5.0 IO.SYS)
                                    ; (SYSINIT:33FAh)

                                    ; 03/01/2023 - Retrodos v4.2 (Modified MSDOS 6.21 IO.SYS)
                                    ; (SYSINIT:4270h)
                                get2:
00003DD5 E304                       jcxz   noget
                                    ;
                                    ; 02/11/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
                                    ;;lods byte ptr es:[si]
                                    ; 12/12/2022
00003DD7 26                         es
00003DD8 AC                         lodsb
                                    ;mov   al, [es:si]
                                    ;inc   si
                                    ;
00003DD9 49                         dec    cx
00003DDA C3                         retn
                                noget:
00003DDB 59                         pop    cx
                                    ; 03/01/2023
                                    ; ds = cs
                                    ;mov   [cs:count],di ; 13/05/2019
                                    ;mov   [cs:org_count],di
00003DDC 893E[5603]                 mov    [count],di
00003DE0 893E[5803]                 mov    [org_count],di
00003DE4 31F6                       xor    si,si
                                    ;mov   [cs:chrptr],si
00003DE6 8936[5A03]                 mov    [chrptr],si

                                    ; 02/11/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)

                                ; 03/01/2023 - Retrodos v4.2 (Modified MSDOS 6.21 IO.SYS)

                                ;ifndef MULTI_CONFIG
                                ;   retn
                                ;else

                                ;   This was the rather kludgy way out of procedure "organize", but instead
                                ;   of returning to doconf, we now want to check config.sys BEGIN/END blocks
                                ;   and the new boot menu stuff...

00003DEA 89F9                       mov    cx,di
00003DEC E9E300                     jmp    menu_check

                                ;endif
                                    ; 02/11/2022
                                    ; 03/01/2023 - Retro DOS v4.2
                                    ;retn

                                ;--------------------------------------------------------------------------------
                                ;
                                ; procedure : skip_comment
                                ;
                                ;skip the commented string until lf, if current es:si-> a comment string.
                                ;in) es:si-> string
                                ;    cx -> length.
                                ;out) zero flag not set if not found a comment string.
                                ;     zero flag set if found a comment string and skipped it. al will contain
                                ;     the line feed character at this moment when return.
                                ;     ax register destroyed.
                                ;     if found, si, cx register adjusted accordingly.
                                ;
                                ;--------------------------------------------------------------------------------

                                    ; 03/01/2023 - Retrodos v4.2 (Modified MSDOS 6.21 IO.SYS)
                                    ; (SYSINIT:428Dh)

                                skip_comment:
00003DEF E3EA                       jcxz   noget          ; get out of the organize routine.

                                    ; 03/01/2023
                                    ; ds = cs

00003DF1 803E[5003]00               cmp    byte [com_level],0
                                    ;cmp   byte [cs:com_level],0 ; only check it if parameter level is 0.
00003DF6 752C                       jne    short no_commt  ; (not inside quotations)

00003DF8 803E[5103]01               cmp    byte [cmmt],1
                                    ;cmp   byte [cs:cmmt],1
00003DFD 7225                       jb     short no_commt

00003DFF 268A04                     mov    al,[es:si]

00003E02 3806[5203]                 cmp    [cmmt1],al
                                    ;cmp   [cs:cmmt1],al
00003E06 751C                       jne    short no_commt

00003E08 803E[5103]02               cmp    byte [cmmt],2
                                    ;cmp   byte [cs:cmmt],2
00003E0D 750A                       jne    short skip_cmmt

00003E0F 268A4401                   mov    al,[es:si+1]

00003E13 3806[5303]                 cmp    [cmmt2],al
                                    ;cmp   [cs:cmmt2],al
00003E17 750B                       jne    short no_commt
                                skip_cmmt:
00003E19 E3C0                       jcxz   noget          ; get out of organize routine.
00003E1B 268A04                     mov    al,[es:si]
00003E1E 46                         inc    si
```

```
41264 00003E1F 49                       dec     cx
41265 00003E20 3C0A                     cmp     al,lf           ; line feed?
41266 00003E22 75F5                     jne     short skip_cmmt
41267                                 no_commt:
41268 00003E24 C3                       retn
41269
41270                                 ; 01/01/2023 - Retro DOS v4.2 (Modified MSDOS 6.21 IO.SYS)
41271                                 ; (SYSINIT:42C8h)
41272
41273                                 ; 02/11/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
41274                                 ;%if 0
41275
41276                                 ;ifdef      MULTI_CONFIG
41277
41278                                 ;--------------------------------------------------------------------------
41279                                 ;
41280                                 ;   kbd_read: wait for keystroke
41281                                 ;
41282                                 ;   INPUT
41283                                 ;       DS == CS == sysinitseg
41284                                 ;
41285                                 ;   OUTPUT
41286                                 ;       Carry SET to clean boot, CLEAR otherwise
41287                                 ;
41288                                 ;   OTHER REGS USED
41289                                 ;       All
41290                                 ;
41291                                 ;   HISTORY
41292                                 ;       Created 16-Nov-1992 by JeffPar
41293                                 ;
41294                                 ;--------------------------------------------------------------------------
41295
41296                                 kbd_read:
41297 00003E25 F606[814C]02                     test    byte [bDisableUI],2
41298 00003E2A 7520                             jnz     short kbd_nodelay
41299
41300 00003E2C 1E                               push    ds              ; the bios timer tick count is incremented
41301 00003E2D 29C0                             sub     ax,ax           ; 18.2 times per second;
41302 00003E2F 8ED8                             mov     ds,ax           ; watch the timer tick count for 37 transitions
41303                                       ;mov   dx,[046Ch]     ; get initial value
41304                                 kbd_loop:
41305 00003E31 B401                             mov     ah,1            ;
41306 00003E33 CD16                             int     16h             ; peek the keyboard
41307 00003E35 7514                             jnz short kbd_loopdone ; something's there, get out
41308 00003E37 B402                             mov     ah,2            ; peek the shift states
41309 00003E39 CD16                             int     16h             ;
41310 00003E3B A803                             test    al,03h          ; either right or left shift key bits set?
41311 00003E3D 750C                             jnz short kbd_loopdone ; yes
41312 00003E3F A16C04                           mov     ax,[046Ch] ;
41313                                       ;sub   ax,dx           ; get difference
41314                                       ; 15/04/2019 - Retro DOS v4.0
41315 00003E42 2E2B06[8C03]                   sub     ax,[cs:_timer_lw_] ; MSDOS 6.21 IO.SYS - SYSINIT:42E5h
41316
41317 00003E47 3C25                           cmp     al,37           ; reached limit?  ; (2 seconds)
41318 00003E49 72E6                             jb  short kbd_loop ; not yet
41319                                 kbd_loopdone:
41320 00003E4B 1F                               pop     ds              ; delay complete!
41321                                 kbd_nodelay:
41322 00003E4C 29DB                             sub     bx,bx           ; assume clean boot
41323 00003E4E B402                             mov     ah,2            ; peek the shift states
41324 00003E50 CD16                             int     16h             ;
41325 00003E52 A803                             test    al,03h          ; either right or left shift key bits set?
41326 00003E54 7407                             jz      short kbd_notshift ; no
41327 00003E56 43                               inc     bx              ; yes
41328 00003E57 43                               inc     bx
41329                                       ; MSDOS 6.21 IO.SYS - SYSINIT:4301h
41330 00003E58 800E[854C]04                   or      byte [bQueryOpt],4
41331                                 kbd_notshift:                   ;
41332 00003E5D B401                             mov     ah,1            ; peek the keyboard
41333 00003E5F CD16                             int     16h             ;
41334 00003E61 743E                             jz  short kbd_test ; no key present
41335 00003E63 08C0                             or      al,al           ; is it a function key?
41336 00003E65 753A                             jnz short kbd_test ; no
41337
41338                                       ; MSDOS 6.21 IO.SYS - SYSINIT:430Bh
41339 00003E67 80FC62                           cmp     ah,62h          ; CTRL F5
41340 00003E6A 7405                             je  short kbd_cfg_bypass
41341
41342 00003E6C 80FC3F                           cmp     ah,3Fh          ; F5 function key?
41343 00003E6F 750D                             jne short kbd_notf5       ; no
41344                                 kbd_cfg_bypass:
41345 00003E71 BA[FC51]                         mov     dx,_$CleanMsg
41346 00003E74 E8DD0B                           call    print
41347                                       ; MSDOS 6.21 IO.SYS - SYSINIT:431Bh
41348 00003E77 800E[854C]04                   or      byte [bQueryOpt],4
41349 00003E7C EB16                             jmp     short kbd_eat   ; yes, clean boot selected
41350                                 kbd_notf5:
41351                                       ; MSDOS 6.21 IO.SYS - SYSINIT:4322h
41352 00003E7E 80FC65                           cmp     ah,65h          ; CTRL F8
41353 00003E81 7405                             je  short kbd_cfg_confirm
41354
41355 00003E83 80FC42                           cmp     ah,42h          ; F8 function key?
41356 00003E86 7523                             jne short kbd_exit ; no
41357                                 kbd_cfg_confirm:
41358 00003E88 BA[3A52]                         mov     dx,_$InterMsg
41359 00003E8B E8C60B                           call    print           ;
41360 00003E8E B301                             mov     bl,1            ; yes, interactive-boot option enabled
41361 00003E90 881E[854C]                       mov     [bQueryOpt],bl  ; change default setting
41362                                 kbd_eat:                        ;
41363 00003E94 B400                             mov     ah,0            ;
41364 00003E96 CD16                             int     16h             ; eat the key we assumed was a signal
41365 00003E98 C606[8B4C]FF                     mov byte [secElapsed],-1
41366 00003E9D 09DB                             or      bx,bx           ;
41367 00003E9F 7405                             jz  short kbd_clean       ;
41368                                 kbd_test:                       ;
41369 00003EA1 80FB02                           cmp     bl,2            ;
41370 00003EA4 7205                             jb  short kbd_exit ;
41371                                 kbd_clean:                      ;
41372 00003EA6 E86E08                           call    disable_autoexec; yes, tell COMMAND to skip autoexec.bat
41373 00003EA9 F9                               stc                     ; set carry to indicate abort
41374 00003EAA C3                               retn                    ;
41375                                 kbd_exit:                       ;
41376 00003EAB F8                               clc                     ; clear carry to indicate success
41377 00003EAC C3                               retn                    ;
41378
41379                                 ;--------------------------------------------------------------------------
41380                                 ;
41381                                 ;   set_numlock: set numlock LED
41382                                 ;
41383                                 ;   INPUT
41384                                 ;       ES:SI -> numlock setting (ie, "ON" or "OFF")
41385                                 ;
41386                                 ;   OUTPUT
41387                                 ;       None
```

```
41388                                   ;
41389                                   ;   OTHER REGS USED
41390                                   ;       None
41391                                   ;
41392                                   ;   HISTORY
41393                                   ;       Created 16-Nov-1992 by JeffPar
41394                                   ;
41395                                   ;-----------------------------------------------------------------------------
41396
41397                                       ; 04/01/2023 - Retro DOS v4.2
41398
41399                                   set_numlock:
41400                                           ; 04/01/2023
41401                                       ;push   ax
41402 00003EAD 1E                             push    ds
41403 00003EAE 29C0                           sub     ax,ax
41404 00003EB0 8ED8                           mov     ds,ax
41405 00003EB2 268B04                         mov     ax,[es:si]      ; get 1st 2 bytes of value (ON or OF)
41406 00003EB5 2E3B06[C451]                   cmp     ax,[cs:OnOff+2]   ; should we turn it off?
41407 00003EBA 7507                           jne short not_off  ; no
41408 00003EBC 80261704DF                     and     byte [0417h],~20h ; 0DFh
41409 00003EC1 EB0D                           jmp     short set_done
41410                                   not_off:
41411 00003EC3 2E3B06[C251]                   cmp     ax,[cs:OnOff]     ; should we turn it on?
41412 00003EC8 F9                             stc
41413 00003EC9 7505                           jne short set_done ; no
41414 00003ECB 800E170420                     or      byte [0417h],20h
41415                                   set_done:
41416 00003ED0 1F                             pop     ds
41417                                       ; 04/01/2023
41418                                       ;pop    ax
41419 00003ED1 C3                             retn
41420
41421                                   ; 16/04/2019 - Retro DOS v4.0
41422
41423                                   ;-----------------------------------------------------------------------------
41424                                   ;
41425                                   ;   menu_check:  check for presence of menu (and other) configuration blocks
41426                                   ;
41427                                   ;   INPUT
41428                                   ;       CX == "organized" config.sys memory image length
41429                                   ;    ES:SI -> "organized" config.sys memory image
41430                                   ;       DS == CS == sysinitseg
41431                                   ;
41432                                   ;   OUTPUT
41433                                   ;       Same as above; the idea is that menu_check simply transforms
41434                                   ;       a block-structured config.sys image into a conventional image,
41435                                   ;       based on the user's block selection and any other boot-time options
41436                                   ;       the user may have employed...
41437                                   ;
41438                                   ;   OTHER REGS USED
41439                                   ;       All
41440                                   ;
41441                                   ;   NOTES
41442                                   ;       [count] and [org_count] are set to the new config.sys image length
41443                                   ;
41444                                   ;   HISTORY
41445                                   ;       Created 16-Mar-1992 by JeffPar
41446                                   ;
41447                                   ;-----------------------------------------------------------------------------
41448
41449                                       ; 04/01/2023 - Retrodos v4.2 (Modified MSDOS 6.21 IO.SYS)
41450                                       ; (SYSINIT:4378h)
41451
41452                                   menu_check:
41453
41454                                   ;   Search for SWITCHES, determine if /N or /F are present; if so, then
41455                                   ;   disable clean/interactive boot options
41456
41457 00003ED2 51                            push    cx              ;
41458 00003ED3 56                            push    si              ;
41459 00003ED4 29DB                          sub     bx,bx           ; remains ZERO until first block
41460                                   swchk_loop:                   ;
41461 00003ED6 E83507                        call    get_char        ; get first char of current line
41462 00003ED9 724C                          jc  short swchk_end       ; hit eof
41463 00003EDB 3C5B                          cmp     al,CONFIG_BEGIN ; '['
41464 00003EDD 7503                          jne short swchk_next1 ;
41465 00003EDF 43                            inc     bx              ; remember that we've seen a block
41466 00003EE0 EB40                          jmp     short swchk_nextline
41467                                   swchk_next1:                  ;
41468 00003EE2 3C4E                          cmp     al,CONFIG_NUMLOCK
41469 00003EE4 750E                          jne short swchk_next2 ;
41470 00003EE6 09DB                          or      bx,bx           ; only do NUMLOCK commands that exist
41471 00003EE8 7538                          jnz short swchk_nextline ; before the first block
41472 00003EEA E8C0FF                        call    set_numlock     ; REM it out so we don't act on it later, too
41473 00003EED 26C644FF30                    mov     byte [es:si-1],CONFIG_REM
41474 00003EF2 EB2E                          jmp     short swchk_nextline
41475                                   swchk_next2:                  ;
41476 00003EF4 3C31                          cmp     al,CONFIG_SWITCHES
41477 00003EF6 752A                          jne short swchk_nextline ; this line ain't it
41478                                   swchk_scan:                   ;
41479 00003EF8 E81307                        call    get_char        ; look for /N or /F
41480                                   swchk_scan1:                  ;
41481 00003EFB 3C0A                          cmp     al,LF           ; end of line?
41482 00003EFD 7423                          je  short swchk_nextline ; yes
41483 00003EFF 3C2F                          cmp     al,'/'          ; switch-char?
41484 00003F01 75F5                          jne short swchk_scan ; no
41485 00003F03 E80807                        call    get_char        ;
41486 00003F06 24DF                          and     al,~20h ; 0DFh    ; convert to upper case
41487 00003F08 3A06[6323]                    cmp     al,[swit_n+1]     ; 'N'
41488 00003F0C 7507                          jne short swchk_scan2 ; no
41489 00003F0E 800E[814C]01                  or      byte [bDisableUI],1
41490 00003F13 EBE3                          jmp short swchk_scan ; continue looking for switches of interest
41491                                   swchk_scan2:                  ;
41492 00003F15 3A06[6F23]                    cmp     al,[swit_f+1]     ; 'F'
41493 00003F19 75E0                          jne short swchk_scan1 ; no
41494 00003F1B 800E[814C]02                  or      byte [bDisableUI],2
41495 00003F20 EBD6                          jmp     short swchk_scan ; continue looking for switches of interest
41496                                   swchk_nextline:               ;
41497 00003F22 E8C306                        call    skip_opt_line   ;
41498 00003F25 EBAF                          jmp     short swchk_loop ;
41499                                   swchk_end:                    ;
41500 00003F27 5E                            pop     si              ;
41501 00003F28 59                            pop     cx              ;
41502
41503                                   ;   Do the keyboard tests for clean/interactive boot now, but only if
41504                                   ;   the DisableUI flag is still clear
41505
41506 00003F29 F606[814C]01                  test    byte [bDisableUI],1
41507 00003F2E 7508                          jnz short menu_search
41508                                   ;
41509                                   ;   Wait for 2 seconds first, UNLESS the /F bit was set in bDisableUI, or
41510                                   ;   there is anything at all in the keyboard buffer
41511                                   ;
```

```
41512 00003F30 E8F2FE                          call    kbd_read
41513 00003F33 7303                            jnc short menu_search
41514 00003F35 E9EE01                          jmp menu_abort
41515
41516                                  ;   Search for MENU block; it is allowed to be anywhere in config.sys
41517
41518                                  menu_search:
41519 00003F38 29DB                            sub     bx,bx           ; if no MENU, default to zero for no_selection
41520 00003F3A BF[C64C]                        mov     di,szMenu ;
41521 00003F3D E80304                          call    find_block      ; find the MENU block
41522 00003F40 7337                            jnc short menu_found ;
41523 00003F42 C606[BE4C]00                    mov     byte [szBoot],0
41524 00003F47 E90C02                          jmp no_selection ; not found
41525
41526                                  ;   Process the requested menu color(s)
41527
41528                                  menu_color:
41529 00003F4A 51                          push    cx              ;
41530 00003F4B 52                          push    dx              ;
41531                                      ;;mov   dx,0007h        ; default color setting
41532                                      ; 10/09/2023
41533                                      ;mov    dl,7 ; !*!
41534 00003F4C E89E06                      call    get_number      ; get first number
41535 00003F4F 80E30F                      and     bl,0Fh  ; !**! ; first # is foreground color (for low nibble)
41536 00003F52 88DD                        mov     ch,bl           ; save it in CH
41537                                      ; 01/08/2023 - Retro DOS v4.2 IO.SYS (optimization) by Erdogan Tan
41538                                      ; (high nibble of dl is 0)
41539                                      ;and    dl,0F0h ; !*!   ; (low nibble of dl would be zero)
41540                                      ;or     dl,bl           ; (low nibble of dl is 7) ! 14/08/2023
41541 00003F54 88DA                        mov     dl,bl   ; 14/08/2023
41542 00003F56 E83108                      call    delim           ; did we hit a delimiter
41543 00003F59 750E                        jne     short check_color ; no, all done
41544 00003F5B E88F06                      call    get_number      ; get next number
41545 00003F5E 80E30F                      and     bl,0Fh          ; second # is background color (for high nibble)
41546 00003F61 88DE                        mov     dh,bl           ; save it in DH
41547                                      ; 10/09/2023
41548                                      ;and    dl,0Fh  ; !**!  ;
41549 00003F63 B104                        mov     cl,4            ;
41550 00003F65 D2E3                        shl     bl,cl           ;
41551 00003F67 08DA                        or      dl,bl           ;
41552                                  check_color:
41553 00003F69 38F5                        cmp     ch,dh           ; are foreground/background the same?
41554 00003F6B 7503                        jne     short set_color       ; no
41555 00003F6D 80F208                      xor     dl,08h          ; yes, so modify the fgnd intensity
41556                                  set_color:
41557 00003F70 8816[7C4C]                  mov     [bMenuColor],dl ;
41558 00003F74 5A                          pop     dx              ;
41559 00003F75 59                          pop     cx              ;
41560 00003F76 E9A900                      jmp     menu_nextitem
41561
41562                                  ;   Back to our regularly scheduled program (the COLOR and other goop
41563                                  ;   above is there simply to alleviate short jump problems)
41564
41565                                  menu_found:
41566 00003F79 C606[864C]01                mov     byte [bDefBlock],1
41567                                      ;mov       word [offDefBlock],0
41568 00003F7E C606[8A4C]FF                mov     byte [secTimeOut],-1
41569 00003F83 8026[854C]FD                and     byte [bQueryOpt],~2 ; 0FDh
41570                                      ; 10/09/2023
41571 00003F88 29D2                        sub     dx,dx
41572 00003F8A 8916[884C]                  mov     [offDefBlock],dx ; 0
41573
41574 00003F8E E85706                          call    skip_opt_line   ; skip to next line
41575                                      ; 10/09/2023
41576                                      ;sub    dx,dx           ; initialize total block count (0 => none yet)
41577
41578                                  ;   Process the menu block now
41579
41580                                  menu_process:
41581 00003F91 E87A06                      call    get_char        ; get first char of current line
41582 00003F94 722E                        jc  short to_menu_getdefault ; could happen if menu block at end (rare)
41583 00003F96 247F                        and     al,~CONFIG_OPTION_QUERY ; 7Fh
41584 00003F98 3C5B                        cmp     al,CONFIG_BEGIN ; BEGIN implies END
41585 00003F9A 7428                        je  short to_menu_getdefault
41586 00003F9C 3C4F                        cmp     al,CONFIG_SUBMENU
41587 00003F9E 744D                        je  short menu_item       ; go process sub-menu
41588 00003FA0 3C45                        cmp     al,CONFIG_MENUITEM
41589 00003FA2 7449                        je  short menu_item       ; go process menu item
41590 00003FA4 3C41                        cmp     al,CONFIG_MENUDEFAULT
41591 00003FA6 741E                        je  short menu_default ; go process menu default
41592 00003FA8 3C52                        cmp     al,CONFIG_MENUCOLOR
41593 00003FAA 749E                        je  short menu_color ; go process menu color
41594 00003FAC 3C4E                        cmp     al,CONFIG_NUMLOCK
41595 00003FAE 740F                        je  short menu_numlock ;
41596 00003FB0 3C30                        cmp     al,CONFIG_REM   ; allow remarks in menu block
41597 00003FB2 746E                        je  short menu_nextitem ;
41598 00003FB4 E8C307                      call    any_delim       ; allow blank lines and such
41599 00003FB7 7469                        je  short menu_nextitem ;
41600 00003FB9 F9                          stc                     ;
41601 00003FBA E82607                      call    print_error     ; non-MENU command!
41602 00003FBD EB63                        jmp     short menu_nextitem
41603                                  menu_numlock:
41604 00003FBF E8EBFE                      call    set_numlock
41605 00003FC2 EB5E                        jmp     short menu_nextitem
41606                                  to_menu_getdefault:
41607 00003FC4 EB62                        jmp     short menu_getdefault
41608
41609                                  ;   Save the offset of the default block name, we'll need it later
41610
41611                                  menu_default:
41612 00003FC6 8936[884C]                  mov     [offDefBlock],si; save address of default block name
41613 00003FCA 803E[8B4C]00                cmp     byte [secElapsed],0
41614 00003FCF 751A                        jne short timeout_skip ; secElapsed is only zero for the FIRST menu,
41615 00003FD1 E8EA05                      call    skip_token      ; and for subsequent menus IF nothing was typed;
41616 00003FD4 724C                        jc  short menu_nextitem ; secElapsed becomes -1 forever as soon as
41617 00003FD6 E8FB05                      call    skip_delim      ; something is typed
41618 00003FD9 7247                        jc  short menu_nextitem ;
41619 00003FDB 89DE                        mov     si,bx           ;
41620 00003FDD E80D06                      call    get_number      ; get number (of seconds for timeout)
41621 00003FE0 80FB5A                      cmp     bl,90           ; limit it to a reasonable number
41622                                      ;jb     short timeout_ok ; (besides, 99 is the largest # my simple
41623 00003FE3 7602                        jna short timeout_ok ; 01/08/2023
41624 00003FE5 B35A                    mov     bl,90           ;  display function can handle)
41625                                  timeout_ok:                     ;
41626 00003FE7 881E[8A4C]                  mov     [secTimeOut],bl ;
41627                                  timeout_skip:
41628 00003FEB EB35                        jmp     short menu_nextitem
41629
41630                                  ;   Verify that this is a valid menu item by searching for the named block
41631
41632                                  menu_item:
41633                                      ;cmp    dl,9    ; 04/01/2023
41634 00003FED 80FA09                      cmp     dl,MAX_MULTI_CONFIG ; have we reached the max # of items yet?
41635 00003FF0 7330                        jae short menu_nextitem ;
```

```
41636 00003FF2 89F7                             mov     di,si           ; DS:DI -> block name to search for
41637 00003FF4 E83303                           call    srch_block      ;
41638 00003FF7 7406                             je  short menu_itemfound ;
41639 00003FF9 F9                               stc                     ;
41640 00003FFA E8E606                           call    print_error     ; print error and pause
41641 00003FFD EB23                             jmp     short menu_nextitem ; if not found, ignore this menu item
41642
41643                                   ;   srch_block, having succeeded, returns DI -> past the token that it
41644                                   ;   just matched, which in this case should be a descriptive string; ES:SI
41645                                   ;   and CX are unmodified
41646
41647                                   menu_itemfound:
41648 00003FFF 42                               inc     dx              ; otherwise, increment total block count
41649 00004000 89D3                             mov     bx,dx           ; and use it to index the arrays of offsets
41650 00004002 8887[8C4C]                       mov [abBlockType+bx],al
41651 00004006 01DB                             add     bx,bx           ; of recorded block names and descriptions
41652
41653                                   ;   There should be a description immediately following the block name on
41654                                   ;   MENUITEM line; failing that, we'll just use the block name as the
41655                                   ;   description...
41656
41657 00004008 89B7[964C]                           mov     [aoffBlockName+bx],si
41658 0000400C 89B7[AA4C]                       mov     [aoffBlockDesc+bx],si
41659 00004010 89DF                             mov     di,bx           ; skip_delim modifies BX, so stash it in DI
41660 00004012 E8A905                           call    skip_token      ;
41661 00004015 720B                             jc  short menu_nextitem ; hit eol/eof
41662 00004017 E8BA05                           call    skip_delim      ;
41663 0000401A 7206                             jc  short menu_nextitem ; hit eol/eof
41664 0000401C 87DF                             xchg    bx,di           ;
41665 0000401E 89BF[AA4C]                       mov     [aoffBlockDesc+bx],di
41666
41667                                   menu_nextitem:
41668 00004022 E8C305                           call    skip_opt_line   ;
41669 00004025 E969FF                           jmp     menu_process    ; go back for more lines
41670
41671                                   ;   Display menu items now, after determining which one is default
41672
41673                                   menu_getdefault:
41674 00004028 08D2                             or      dl,dl           ; where there any valid blocks at all?
41675 0000402A 7505                             jnz short menu_valid ; yes
41676 0000402C 29DB                             sub     bx,bx           ; no, so force autoselect of 0
41677 0000402E E9ED00                           jmp     menu_autoselect ; (meaning: process common blocks only)
41678                                   menu_valid:
41679 00004031 29DB                             sub     bx,bx           ;
41680 00004033 8816[874C]                       mov     [bMaxBlock],dl  ; first, record how many blocks we found
41681 00004037 8B3E[884C]                       mov     di,[offDefBlock];
41682 0000403B 09FF                             or      di,di           ; does a default block exist?
41683 0000403D 741C                             jz  short menu_nodefault ; no
41684 0000403F 43                               inc     bx              ; yes, walk name table, looking for default
41685                                   menu_chkdefault:
41686 00004040 53                               push    bx              ;
41687 00004041 01DB                             add     bx,bx           ;
41688 00004043 8BB7[964C]                       mov     si,[aoffBlockName+bx]
41689 00004047 B98000                           mov     cx,128          ; arbitrary maximum length of a name
41690 0000404A 1E                               push    ds              ;
41691 0000404B 06                               push    es              ;
41692 0000404C 1F                               pop     ds              ;
41693 0000404D E81A03                           call    comp_names      ; is this block the same as the default?
41694 00004050 1F                               pop     ds              ;
41695 00004051 5B                               pop     bx              ;
41696 00004052 7409                             je  short menu_setdefault ; yes
41697 00004054 43                               inc     bx              ;
41698 00004055 3A1E[874C]                       cmp     bl,[bMaxBlock]  ; all done searching?
41699 00004059 76E5                             jbe short menu_chkdefault ; not yet
41700                                   menu_nodefault:
41701 0000405B B301                             mov     bl,1            ; if no default, force default to #1
41702                                   menu_setdefault:
41703 0000405D 881E[864C]                       mov     [bDefBlock],bl  ; yes, this will be the initial current block
41704
41705                                   ;   If the timeout was explicitly set to 0 (or technically, anything that
41706                                   ;   failed to resolve to a number, like "NONE" or "EAT POTATOES"), then we're
41707                                   ;   supposed to skip menu display and run with the specified default block;
41708                                   ;   however, if the user hit Enter prior to boot, thereby requesting fully
41709                                   ;   INTERACTIVE boot, then we shall display the menu block anyway (though still
41710                                   ;   with no timeout)
41711
41712 00004061 803E[8A4C]00                        cmp     byte [secTimeOut],0 ; is timeout zero? (ie, assume default)
41713 00004066 750A                             jne short menu_display ; no
41714 00004068 F606[854C]01                     test    byte [bQueryOpt],1 ; yes, but was INTERACTIVE requested?
41715 0000406D 7503                             jnz short menu_display ; yes, so *don't* assume default after all
41716 0000406F E9C700                           jmp     not_topmenu       ;
41717
41718                                   ;   Reset the mode, so that we know screen is clean and cursor is home
41719
41720                                   menu_display:
41721 00004072 B40F                             mov     ah,0Fh          ; get current video mode
41722 00004074 CD10                             int     10h             ;
41723 00004076 B400                             mov     ah,00h          ; just re-select that mode
41724 00004078 CD10                             int     10h             ;
41725 0000407A 06                               push    es              ;
41726 0000407B B84000                           mov     ax,40h          ; reach down into the ROM BIOS data area
41727 0000407E 8EC0                             mov     es,ax           ; and save the current (default) video page
41728 00004080 26A14E00                         mov     ax,[es:004Eh]   ; start address and page #, in case the
41729 00004084 A3[834C]                         mov     [wCRTStart],ax  ; undocumented QUIET option was enabled
41730 00004087 26A06200                         mov     al,[es:0062h]   ;
41731 0000408B A2[824C]                         mov     [bCRTPage],al   ;
41732 0000408E A1[7D4C]                         mov     ax,[bMenuPage]    ; select new page for menu
41733 00004091 CD10                             int     10h             ;
41734 00004093 B80006                           mov     ax,0600h        ; clear entire screen
41735 00004096 8A3E[7C4C]                       mov     bh,[bMenuColor] ; using this color
41736 0000409A 29C9                             sub     cx,cx           ; upper left row/col
41737                                           ;mov       dl,[es:CRT_Cols]
41738 0000409C 268A164A00                       mov dl,[es:4Ah]
41739 000040A1 FECA                         dec     dl              ;
41740                                           ;mov       dh,[es:CRT_Rows];
41741 000040A3 268A368400                       mov dh,[es:84h]
41742 000040A8 08F6                         or      dh,dh           ; # of rows valid?
41743 000040AA 7504                             jnz short menu_clear ; hopefully
41744 000040AC 8A36[804C]                       mov     dh,[bLastRow]   ; no, use a default
41745                                   menu_clear:
41746 000040B0 CD10                             int     10h             ; clear the screen using the req. attribute
41747 000040B2 07                               pop     es              ;
41748 000040B3 8836[804C]                       mov     [bLastRow],dh   ; save DH
41749 000040B7 BA[7752]                         mov     dx,_$MenuHeader
41750 000040BA E89709                           call    print           ; cursor now on row 3 (numbered from 0)
41751
41752 000040BD F606[814C]01                     test    byte [bDisableUI],1
41753 000040C2 751F                             jnz     short menu_nostatus
41754 000040C4 8A3E[7D4C]                       mov     bh,[bMenuPage]  ;
41755 000040C8 8A36[804C]                       mov     dh,[bLastRow]   ; restore DH
41756 000040CC B200                             mov     dl,0            ; print the status line on row DH, col 0,
41757 000040CE B402                             mov     ah,02h          ; now that we can trash the cursor position
41758 000040D0 CD10                             int     10h             ;
41759 000040D2 BA[C352]                         mov     dx,_$StatusLine
```

```
41760 000040D5 E87C09                   call    print           ;
41761 000040D8 B403                     mov     ah,3            ; get cursor position
41762 000040DA CD10                     int     10h             ;
41763 000040DC 80EA02                   sub     dl,2            ;
41764 000040DF 8816[7F4C]               mov     [bLastCol],dl   ; save column where status char will go
41765
41766                                menu_nostatus:
41767 000040E3 BB0100                   mov     bx,1            ; now prepare to display all the menu items
41768                                menu_disploop:
41769 000040E6 E8B002                   call    print_item ; print item #BL
41770 000040E9 43                       inc     bx              ; why "inc bx"? because it's a 1-byte opcode
41771 000040EA 3A1E[874C]               cmp     bl,[bMaxBlock]  ; all done?
41772 000040EE 76F6                     jbe short menu_disploop ; not yet
41773
41774                                ;   Set cursor position to just below the menu items
41775
41776 000040F0 B200                      mov     dl,0            ; select column
41777 000040F2 88DE                     mov     dh,bl           ;
41778 000040F4 80C604                   add     dh,4            ; select row below menu
41779 000040F7 8A3E[7D4C]               mov     bh,[bMenuPage]  ;
41780 000040FB B402                     mov     ah,02h          ; set cursor position beneath the block list
41781 000040FD CD10                     int     10h             ;
41782
41783 000040FF BA[B052]                 mov     dx,_$MenuPrmpt
41784 00004102 E84F09                   call    print           ;
41785 00004105 E82903                   call    select_item     ; make a selection, return # in BX
41786 00004108 BA[7050]                 mov     dx,crlfm
41787 0000410B E84609                   call    print           ;
41788 0000410E FF36[814C]               push    word [bDisableUI]
41789 00004112 800E[814C]01             or      byte [bDisableUI],1
41790 00004117 E86704                   call    show_status     ; clear the status line now
41791 0000411A 8F06[814C]               pop     word [bDisableUI]
41792
41793                                ;   Now begins the "re-organization" process...
41794
41795                                menu_autoselect:
41796 0000411E 83FBFF                   cmp     bx,-1 ; 0FFFFh    ; clean boot requested?
41797 00004121 7508                     jne short normal_boot ; no
41798 00004123 E8F105                   call    disable_autoexec; basically, add a /D to the command.com line
41799                                menu_abort:
41800 00004126 29C9                     sub     cx,cx           ; then immediately exit with 0 config.sys image
41801 00004128 E9E400                   jmp menu_exit      ;
41802
41803                                normal_boot:
41804 0000412B 83FBFE                   cmp     bx,-2 ; 0FFFEh    ; back to top-level menu?
41805 0000412E 7509                     jne short not_topmenu ; no
41806 00004130 8B0E[5603]               mov     cx,[count]      ; yes, start all over
41807 00004134 29F6                     sub     si,si           ;
41808 00004136 E9FFFD                   jmp     menu_search
41809
41810                                not_topmenu:
41811 00004139 80BF[8C4C]4F             cmp     byte [abBlockType+bx],CONFIG_SUBMENU
41812 0000413E 7510                     jne short not_submenu
41813 00004140 01DB                     add     bx,bx           ;
41814 00004142 8BBF[964C]               mov     di,[aoffBlockName+bx]
41815 00004146 E8E101                   call    srch_block      ; THIS CANNOT FAIL!
41816 00004149 89FE                     mov     si,di           ;
41817 0000414B 89D9                     mov     cx,bx           ; ES:SI and CX are ready for another round
41818 0000414D E929FE                   jmp     menu_found
41819
41820                                not_submenu:
41821 00004150 01DB                     add     bx,bx           ; get BX -> name of selected block
41822 00004152 8B9F[964C]               mov     bx,[aoffBlockName+bx]
41823
41824                                ;   BX should now either be ZERO (meaning no block has been selected) or
41825                                ;   the offset relative to ES of the block name to be processed (along with
41826                                ;   all the "common" lines of course)
41827
41828                                no_selection:
41829 00004156 891E[884C]               mov     [offDefBlock],bx; save selection
41830 0000415A 8B0E[5603]               mov     cx,[count]      ; reset ES:SI and CX for reprocessing
41831 0000415E 29F6                     sub     si,si           ;
41832 00004160 1E                       push    ds              ;
41833 00004161 8E1E[6219]               mov     ds,[config_wrkseg]; this is where we'll store new config.sys image
41834 00004165 29FF                     sub     di,di           ;
41835
41836                                ;   ES:SI-> config.sys, DS:DI -> new config.sys workspace
41837                                ;
41838                                ;   Work our way through the config.sys image again, this time copying
41839                                ;   all lines that are (A) "common" lines outside any block or (B) lines
41840                                ;   within the requested block. Lines inside INCLUDEd blocks are transparently
41841                                ;   copied by copy_block in a recursive fashion; the amount of recursion is
41842                                ;   limited by the fact INCLUDE statements are REMed by copy_block as they are
41843                                ;   processed and by the number of unique INCLUDE stmts in config.sys...
41844                                ;
41845                                ;   BUGBUG 20-Mar-1992 JeffPar: If we can figure out the lower bound of the
41846                                ;   stack we're running on, then we should check it inside copy_block
41847
41848                                copyblock_loop:
41849 00004167 53                        push    bx              ; save selected block name
41850 00004168 E82F01                   call    copy_block      ; process (named or common) block
41851 0000416B 5B                       pop     bx              ;
41852 0000416C 7232                     jc  short move_config ; hit eof
41853
41854                                ;   copy_block can only return for two reasons: it hit eof or a new block
41855
41856                                copyblock_begin:
41857
41858                                ; 10/09/2023
41859                                %if 0
41860                                        push    ax              ;
41861                                        push    cx              ;
41862                                        push    si              ;
41863                                        push    di              ; always do "common" blocks
41864                                        mov     di,szCommon
41865                                        push    ds              ;
41866                                        push    cs              ;
41867                                        pop     ds              ;
41868                                        call    comp_names      ;
41869                                        pop     ds              ;
41870                                        pop     di              ;
41871                                        pop     si              ;
41872                                        pop     cx              ;
41873                                        pop     ax              ;
41874                                        je  short copyblock_check
41875                                %endif
41876                                    ; 10/09/2023
41877 0000416E 57                        push    di
41878 0000416F BF[CB4C]                 mov     di,szCommon    ; always do "common" blocks
41879 00004172 E81602                   call    comp_names_x   ; (comp_names_safe)
41880 00004175 5F                       pop     di
41881 00004176 740F                     je      short copyblock_check
41882
41883 00004178 09DB                         or      bx,bx           ; is there a block name to check?
```

```
41884 0000417A 7414                     jz  short copyblock_skip ; no
41885 0000417C 57                       push    di              ;
41886 0000417D 89DF                     mov     di,bx           ; check block against given block name
41887 0000417F 1E                       push    ds              ;
41888 00004180 06                       push    es              ;
41889 00004181 1F                       pop     ds              ;
41890 00004182 E8E501                   call    comp_names      ; is this the block we really want to do?
41891 00004185 1F                       pop     ds              ;
41892 00004186 5F                       pop     di              ;
41893                           copyblock_check:
41894 00004187 7217                     jc  short move_config ; hit eof
41895 00004189 7505                     jne short copyblock_skip ;
41896 0000418B E85A04                   call    skip_opt_line   ;
41897 0000418E EBD7                     jmp short copyblock_loop
41898
41899                           copyblock_skip:                 ;
41900 00004190 E85504                   call    skip_opt_line   ; this ain't the block we wanted, so skip it
41901 00004193 E87804                   call    get_char        ;
41902 00004196 7208                     jc  short move_config ; hit eof
41903 00004198 247F                     and     al,~CONFIG_OPTION_QUERY ; 7Fh
41904 0000419A 3C5B                     cmp     al,CONFIG_BEGIN ;
41905 0000419C 74D0                     je  short copyblock_begin
41906 0000419E EBF0                     jmp     short copyblock_skip ; anything else is just skipped
41907                           ;
41908                           ;   To create as little risk to the rest of SysInit as little as possible,
41909                           ;   and to free the workspace at "config_wrkseg" for creating an environment,
41910                           ;   copy the new config.sys image to "confbot"
41911                           ;
41912                           move_config:
41913 000041A0 89F9                     mov     cx,di           ; now copy workspace at DS:DI to "confbot"
41914 000041A2 51                       push    cx              ;
41915                           ;
41916                           ;   But first, copy the CONFIG=<configuration><0> string to the workspace,
41917                           ;   since the configuration name only currently exists in the "confbot" area
41918                           ;
41919                               ;mov    cx,7
41920 000041A3 B90700                  mov     cx,szMenu-szBoot-1
41921 000041A6 BE[BE4C]                 mov     si,szBoot ; first copy the CONFIG= part
41922 000041A9 47                       inc     di              ; skip a byte, in case absolutely nothing
41923                                                           ; was copied to the workspace, because we always
41924                                                           ; zero the first byte of the workspace (below)
41925                           copy_boot:
41926                               ;lods    byte ptr cs:[si];
41927 000041AA 2E                       cs
41928 000041AB AC                      lodsb
41929 000041AC 8805                    mov     [di],al         ;
41930 000041AE 47                       inc     di              ;
41931 000041AF E2F9                     loop    copy_boot       ;
41932
41933 000041B1 06                       push    es              ; then copy the configuration name
41934                                   ;mov       cx,128-7      ; put an upper limit on the name, to be safe
41935                               ; 04/01/2023
41936 000041B2 B179                    mov     cl,128-7
41937 000041B4 2E8B36[884C]            mov     si,[cs:offDefBlock]; ES:SI -> default block name
41938 000041B9 09F6                     or      si,si           ; valid?
41939 000041BB 7505                     jnz short l1        ; yes
41940 000041BD 0E                       push    cs              ;
41941 000041BE 07                       pop     es              ;
41942 000041BF BE[CB4C]                 mov     si,szCommon
41943 000041C2 268A04                  l1:  mov     al,[es:si]      ;
41944 000041C5 E8B205                   call    any_delim       ;
41945 000041C8 7406                     je  short l2        ;
41946 000041CA 8805                     mov     [di],al         ;
41947 000041CC 46                       inc     si              ;
41948 000041CD 47                       inc     di              ;
41949 000041CE E2F2                     loop    l1              ;
41950 000041D0 C6050A                  l2:  mov     byte [di],lf     ; terminate the configuration string
41951 000041D3 07                       pop     es              ;
41952
41953                           ;   Now we can copy "config_wrkseg" (DS) to "confbot" (ES)
41954
41955 000041D4 29FF                     sub     di,di           ;
41956 000041D6 2E893E[6019]             mov     [cs:config_envlen],di
41957 000041DB 29F6                     sub     si,si           ;
41958 000041DD 59                       pop     cx              ; recover the size of "config_wrkseg"
41959
41960 000041DE 51                       push    cx              ;
41961 000041DF F3A4                     rep     movsb           ; moved!
41962 000041E1 59                       pop     cx              ;
41963 000041E2 8CD8                     mov     ax,ds           ;
41964 000041E4 1F                       pop     ds              ;
41965
41966                           ;   Now that the config_wrkseg is available once again, we shall
41967                           ;   use it to create an environment. The first thing to go in will be
41968                           ;   the "CONFIG=configuration" thing. It is also important to zero
41969                           ;   the first byte of the workspace, so that copy_envvar knows the buffer
41970                           ;   is empty.
41971
41972 000041E5 06                       push    es              ;
41973 000041E6 8EC0                     mov     es,ax           ;
41974 000041E8 46                       inc     si              ; ES:SI -> "CONFIG=configuration"
41975 000041E9 26C606000000             mov     byte [es:0],0     ;empty the environment block
41976 000041EF E82600                   call    copy_envvar     ; copy envvar at ES:SI to "config_wrkseg"
41977 000041F2 07                       pop     es
41978
41979                           ;   Before returning, restore the default video page setting but do NOT
41980                           ;   do it using INT 10h's Set Active Page function, because if the menu was
41981                           ;   displayed on a different page, then it's because we don't want to see
41982                           ;   all the device driver/TSR goop (which goes to the default page)
41983
41984                           menu_done:
41985 000041F3 803E[7D4C]00             cmp     byte [bMenuPage],0
41986 000041F8 7415                     je  short menu_exit       ;
41987 000041FA 06                       push    es              ;
41988 000041FB B84000                   mov     ax,40h          ;
41989 000041FE 8EC0                     mov     es,ax           ;
41990 00004200 A1[834C]                 mov     ax,[wCRTStart]  ;
41991 00004203 26A34E00                 mov     [es:004Eh],ax   ;
41992 00004207 A0[824C]                 mov     al,[bCRTPage]   ;
41993 0000420A 26A26200                 mov     [es:0062h],al   ;
41994 0000420E 07                       pop     es              ;
41995                           menu_exit:
41996 0000420F 890E[5603]               mov     [count],cx      ; set new counts
41997 00004213 890E[5803]               mov     [org_count],cx  ;
41998                                   ; 10/09/2023 (*) - Erdogan Tan
41999                               ; MSDOS 6.21 IO.SYS - SYSINIT:46D3h
42000                               ; PCDOS 7.1 IBMBIO.COM - SYSINIT:491Ah
42001                               ;sub    si,si           ; always return ES:SI pointing to config.sys
42002 00004217 C3                      retn
42003
42004                           ; (*) NOTE: MSDOS 6.0 source code (SYSINIT2.ASM) contains 'sub si,si' at this
42005                           ;   position (then 'retn' just after it)
42006                           ;   but MSDOS 6.21 and PCDOS 7.1 SYSINITs contain only 'retn' here.
42007
```

```
                                ;--------------------------------------------------------------------------
                                ;
                                ;   copy_envvar:  copy the envvar at ES:SI to "config_wrkseg"
                                ;
                                ;   INPUT
                                ;    ES:SI -> environment variable (in the form "var=string<cr/lf>")
                                ;
                                ;   OUTPUT
                                ;       config_envlen (ie, where to put next envvar) updated appropriately
                                ;       carry set if error (eg, missing =); clear otherwise
                                ;
                                ;   OTHER REGS USED
                                ;       None
                                ;
                                ;   NOTES
                                ;       None
                                ;
                                ;   HISTORY
                                ;       Created 29-Mar-1992 by JeffPar
                                ;
                                ;--------------------------------------------------------------------------

                                    ; 04/01/2023 - Retrodos v4.2 (Modified MSDOS 6.21 IO.SYS)
                                    ; (SYSINIT:46D4h)

                                copy_envvar:
00004218 51                             push    cx              ;
00004219 56                             push    si              ;
0000421A 1E                             push    ds              ;
0000421B 06                             push    es              ;
0000421C 06                             push    es              ;
0000421D 8E06[6219]                     mov     es,[config_wrkseg] ; ES:DI to point to next available byte
00004221 1F                             pop     ds                 ; DS:SI to point to envvar

                                ;   Have to calculate the length of the variable name (and if we hit
                                ;   the end of the line before we hit '=', then it's curtains for this
                                ;   config.sys line)
                                ;
                                ;   The check for NULL is important because copy_envvar is also used to copy
                                ;   the initial CONFIG= setting, which will have been zapped by a NULL if no
                                ;   menu block existed (in order to prevent the creation of an environment)

00004222 29C9                           sub     cx,cx           ;
                                copy_varlen:                    ;
00004224 AC                             lodsb                   ;
00004225 08C0                           or      al,al           ; NULL?
                                        ;stc       ; 10/09/2023 (x)
00004227 746B                           jz  short copy_envexit ; yes, abort
00004229 3C0D                           cmp     al,cr           ;
                                        ;stc       ; 10/09/2023 (x)
0000422B 7467                           je  short copy_envexit
0000422D 3C0A                           cmp     al,lf           ;
                                        ;stc       ; 10/09/2023 (x)
0000422F 7463                           je  short copy_envexit
00004231 41                             inc     cx              ;
00004232 3C3D                           cmp     al,'='          ;
00004234 75EE                           jne short copy_varlen
00004236 B000                           mov     al,0            ;
00004238 8A24                           mov     ah,[si]         ; save char after '='
0000423A 29CE                           sub     si,cx           ; back up to given varname
0000423C 49                             dec     cx              ; CX == # of bytes in varname
0000423D 29FF                           sub     di,di           ; start looking for DS:SI at ES:0
                                copy_varsrch:
0000423F 263805                         cmp     byte [es:di],al
00004242 7425                           je  short copy_envprep ; search failed, just copy var
00004244 89FB                           mov     bx,di           ; ES:BX -> start of this varname
00004246 51                             push    cx              ;
00004247 56                             push    si              ;
00004248 F3A6                           repe    cmpsb           ;
0000424A 5E                             pop     si              ;
0000424B 59                             pop     cx              ;
0000424C 7531                           jne short copy_varnext ; no match, skip to next varname
0000424E 26803D3D                       cmp     byte [es:di],'='
00004252 752B                           jne     short copy_varnext ; no match, there's more characters

                                ;   Previous occurrence of variable has been found; determine the
                                ;   entire length and then destroy it

00004254 B9FFFF                         mov     cx,-1           ;
00004257 F2AE                           repne   scasb           ; guaranteed to get null (since we put it there)
00004259 56                             push    si              ;
0000425A 89FE                           mov     si,di           ;
0000425C 89DF                           mov     di,bx           ;
0000425E 2E8B0E[6019]                   mov     cx,[cs:config_envlen]
00004263 29F1                           sub     cx,si           ; destroy variable now
                                    ;rep movs byte ptr es:[di],byte ptr es:[si]
                                    ;;db 0F3h,26h,0A4h ; MSDOS 6.21 IO.SYS - SYSINIT:4724h

00004265 F3                             rep     ; 0F3h
00004266 26                             es      ; 26h
00004267 A4                             movsb   ; 0A4h

00004268 5E                             pop     si
                                copy_envprep:
00004269 80FC0D                         cmp     ah,cr           ; if there is nothing after the '='
0000426C 741D                           je  short copy_envdel ; then just exit with variable deleted
0000426E 80FC0A                         cmp     ah,lf           ;
00004271 7418                           je  short copy_envdel
                                        ;jmp       short copy_envloop
                                    ; 04/01/2023
                                copy_envloop:                   ;
00004273 AC                             lodsb                   ;
00004274 3C0D                           cmp     al,cr           ;
00004276 7410                           je      short copy_envdone
00004278 3C0A                           cmp     al,lf           ;
0000427A 740C                           je      short copy_envdone
0000427C AA                             stosb                   ;
0000427D EBF4                           jmp     short copy_envloop

                                copy_varnext:                   ;
0000427F 51                             push    cx              ;
00004280 B9FFFF                         mov     cx,-1           ;
00004283 F2AE                           repne   scasb           ;
00004285 59                             pop     cx              ;
00004286 EBB7                           jmp short copy_varsrch

                                    ; 04/01/2023
                                ;copy_envloop:                  ;
                                ;   lodsb                       ;
                                ;   cmp     al,cr           ;
                                ;   je      short copy_envdone
                                ;   cmp     al,lf           ;
                                ;   je      short copy_envdone
                                ;   stosb                       ;
```

```
42132                                   ;   jmp     short copy_envloop
42133
42134                                   copy_envdone:                   ;
42135 00004288 28C0                             sub     al,al           ; do SUB to clear carry as well
42136 0000428A AA                               stosb                   ; always null-terminate these puppies
42137                                   copy_envdel:                    ;
42138 0000428B 268805                           mov     [es:di],al      ; and stick another null to terminate the env.
42139 0000428E 2E893E[6019]                     mov     [cs:config_envlen],di
42140                                       ; 10/09/2023 (x) - Erdogan Tan
42141 00004293 F9                           stc ; in order to clear carry flag via cmc (compact code trick!)
42142                                   copy_envexit:                   ;
42143 00004294 F5                       cmc ; (x) ; reverse carry flag status (je -> cf=1)
42144 00004295 07                           pop     es              ;
42145 00004296 1F                           pop     ds              ;
42146 00004297 5E                           pop     si              ;
42147 00004298 59                           pop     cx              ;
42148
42149                                   copy_done:  ; 18/12/2022
42150 00004299 C3                               retn
42151
42152                                   ;---------------------------------------------------------------------------
42153                                   ;
42154                                   ;   copy_block:  copy the current block to the new config.sys workspace
42155                                   ;
42156                                   ;   INPUT
42157                                   ;       CX == remaining bytes in "organized" config.sys memory image
42158                                   ;    ES:SI -> remaining bytes in "organized" config.sys memory image
42159                                   ;    DS:DI -> new config.sys workspace (equal in size to the original
42160                                   ;             config.sys image) where the current block is to be copied
42161                                   ;
42162                                   ;   OUTPUT
42163                                   ;       Same as above
42164                                   ;       AL also equals the last character read from the organized image
42165                                   ;
42166                                   ;   OTHER REGS USED
42167                                   ;       All
42168                                   ;
42169                                   ;   NOTES
42170                                   ;       None
42171                                   ;
42172                                   ;   HISTORY
42173                                   ;       Created 16-Mar-1992 by JeffPar
42174                                   ;
42175                                   ;---------------------------------------------------------------------------
42176
42177                                       ; 04/01/2023 - Retrodos v4.2 (Modified MSDOS 6.21 IO.SYS)
42178                                       ; (SYSINIT:4759h)
42179
42180                                   copy_block:
42181 0000429A E87103                           call    get_char        ; check for include
42182 0000429D 72FA                             jc  short copy_done     ;
42183 0000429F 247F                         and     al,~CONFIG_OPTION_QUERY ; 7Fh
42184 000042A1 3C5B                         cmp     al,CONFIG_BEGIN ; another BEGIN implies END as well
42185 000042A3 74F4                             je  short copy_done ;
42186
42187 000042A5 3C4A                             cmp     al,CONFIG_INCLUDE ; 'J'
42188 000042A7 88E0                             mov     al,ah           ; AL == the original line code
42189 000042A9 753A                             jne short copy_line     ; not an "include" line
42190
42191                                   ;   We have hit an "INCLUDE" line; first, REM out the line so that we
42192                                   ;   never try to include the block again (no infinite include loops please),
42193                                   ;   then search for the named block and call copy_block again.
42194
42195 000042AB 26C644FF30                       mov     byte [es:si-1],CONFIG_REM ; '0'
42196 000042B0 57                               push    di              ;
42197
42198 000042B1 BF[C64C]                         mov     di,szMenu
42199 000042B4 E8D400                           call    comp_names_safe ; don't allow INCLUDE MENU
42200 000042B7 7426                             je  short copy_skip     ;
42201
42202 000042B9 BF[CB4C]                         mov     di,szCommon
42203 000042BC E8CC00                           call    comp_names_safe ; don't allow INCLUDE COMMON
42204 000042BF 741E                             je  short copy_skip     ;
42205
42206 000042C1 89F7                             mov     di,si           ; try to find the block
42207 000042C3 E86400                           call    srch_block      ;
42208 000042C6 89FA                             mov     dx,di           ;
42209                                       ; 10/09/2023
42210                                       ;pop    di              ;
42211 000042C8 7514                             jne short copy_error ; no such block
42212 000042CA 5F                           pop     di  ; 10/09/2023
42213 000042CB 51                               push    cx              ;
42214 000042CC 89D9                             mov     cx,bx           ;
42215 000042CE 56                               push    si              ;
42216 000042CF 4A                               dec     dx              ;
42217 000042D0 89D6                             mov     si,dx           ;
42218 000042D2 E80E03                           call    skip_line       ; skip the rest of the "block name" line
42219 000042D5 E8C2FF                           call    copy_block      ; and copy in the rest of that block
42220 000042D8 5E                               pop     si              ;
42221 000042D9 59                               pop     cx              ;
42222 000042DA 28C0                             sub     al,al           ; force skip_opt_line to skip...
42223 000042DC EB2B                             jmp     short copy_nextline
42224
42225                                   copy_error:
42226                                       ; 10/09/2023
42227 000042DE F8                           clc
42228                                   copy_skip:
42229 000042DF 5F                               pop     di
42230                                   ;copy_error:
42231                                       ; 10/09/2023 (cf=0)
42232                                       ;clc                    ;
42233 000042E0 E80004                           call    print_error     ; note that carry is clear, no pause
42234 000042E3 EB24                             jmp     short copy_nextline
42235
42236                                   ;   Copy the line at ES:SI to the current location at DS:DI
42237
42238                                   copy_line:
42239 000042E5 8805                             mov     [di],al         ;
42240 000042E7 47                               inc     di              ;
42241 000042E8 3C20                             cmp     al,' '          ; is this is a "real" line with a "real" code?
42242 000042EA 721D                             jb  short copy_nextline ; no
42243 000042EC 2E803E[6519]00                   cmp     byte [cs:config_multi],0
42244 000042F2 7409                             je  short copy_loop     ; not a multi-config config.sys, don't embed #s
42245 000042F4 E81700                           call    get_linenum     ; BX == line # of line @ES:SI
42246 000042F7 891D                             mov     [di],bx         ; stash it immediately following the line code
42247 000042F9 47                               inc     di              ;
42248 000042FA 47                               inc     di              ;
42249 000042FB EB08                             jmp     short copy_next ;
42250                                   copy_loop:                      ;
42251 000042FD E80E03                           call    get_char        ;
42252 00004300 7297                             jc  short copy_done ; end of file
42253 00004302 8805                             mov     [di],al         ;
42254 00004304 47                               inc     di              ;
42255                                   copy_next:
```

```
00004305 3C0A                           cmp     al,lf ; 0Ah       ; done with line?
00004307 75F4                           jne short copy_loop       ; nope

                                copy_nextline:
00004309 E8DC02                         call    skip_opt_line   ;
0000430C EB8C                           jmp     short copy_block

                                    ; 18/12/2022
                                ;copy_done:
                                        ;retn

                                ;--------------------------------------------------------------------------
                                ;
                                ;   get_linenum:  return line # (in BX) of current line (@ES:SI)
                                ;
                                ;   INPUT
                                ;    ES:SI -> some line in the config.sys memory image
                                ;
                                ;   OUTPUT
                                ;       BX == line # (relative to 1)
                                ;
                                ;   OTHER REGS USED
                                ;       DX
                                ;
                                ;   NOTES
                                ;       None
                                ;
                                ;   HISTORY
                                ;       Created 16-Mar-1992 by JeffPar
                                ;
                                ;--------------------------------------------------------------------------

                                get_linenum:
0000430E 50                             push    ax              ;
0000430F 29DB                           sub     bx,bx           ; BX == line # (to be returned)
00004311 51                             push    cx              ;
00004312 89F2                           mov     dx,si           ; DX == the offset we're looking for
00004314 56                             push    si              ;
00004315 2E8B0E[5603]                   mov     cx,[cs:count]   ;
0000431A 29F6                           sub     si,si           ; prepare to scan entire file
                                get_linenum_loop:               ;
0000431C E8C402                         call    skip_line       ;
0000431F 7205                           jc  short get_linenum_done
00004321 43                             inc     bx              ;
00004322 39D6                           cmp     si,dx           ; have we exceeded the desired offset yet?
00004324 72F6                           jb  short get_linenum_loop ; no
                                get_linenum_done:               ;
00004326 5E                             pop     si              ;
00004327 59                             pop     cx              ;
00004328 58                             pop     ax              ;
00004329 C3                             retn

                                ;--------------------------------------------------------------------------
                                ;
                                ;   srch_block:  searches entire config.sys for block name @ES:DI
                                ;
                                ;   INPUT
                                ;       ES -> config.sys image
                                ;    ES:DI -> block name to find
                                ;
                                ;   OUTPUT
                                ;       ZF flag set, if found
                                ;    ES:DI -> just past the name in the block heading, if found
                                ;       BX == # bytes remaining from that point, if found
                                ;
                                ;   OTHER REGS USED
                                ;       None
                                ;
                                ;   NOTES
                                ;       This differs from "find_block" in that it searches the ENTIRE
                                ;       config.sys image, not merely the remaining portion, and that it
                                ;       takes a pointer to block name that is *elsewhere* in the image
                                ;       (ie, ES) as opposed to some string constant in our own segment (DS).
                                ;
                                ;   HISTORY
                                ;       Created 16-Mar-1992 by JeffPar
                                ;
                                ;--------------------------------------------------------------------------

                                srch_block:           ; returns BX -> named block in CONFIG.SYS
0000432A 50                             push    ax              ;
0000432B 51                             push    cx              ;
0000432C 2E8B0E[5603]                   mov     cx,[cs:count]   ;
00004331 56                             push    si              ;
00004332 29F6                           sub     si,si           ;
00004334 1E                             push    ds              ;
00004335 06                             push    es              ;
00004336 1F                             pop     ds              ;
00004337 E80900                         call    find_block      ;
0000433A 89F7                           mov     di,si           ;
0000433C 89CB                           mov     bx,cx           ;
0000433E 1F                             pop     ds              ;
0000433F 5E                             pop     si              ;
00004340 59                             pop     cx              ;
00004341 58                             pop     ax              ;
                                find_exit: ; 16/04/2019
00004342 C3                             retn                     ;

                                ;--------------------------------------------------------------------------
                                ;
                                ;   find_block:  searches rest of config.sys for block name @DS:DI
                                ;
                                ;   INPUT
                                ;    DS:DI -> block name to find
                                ;    ES:SI -> remainder of config.sys image
                                ;       CX == remaining size of config.sys image
                                ;
                                ;   OUTPUT
                                ;       ZF flag set, if found (also, CF set if EOF)
                                ;    ES:SI -> where the search stopped (at end of block name or EOF)
                                ;       CX == # bytes remaining from that point
                                ;
                                ;   OTHER REGS USED
                                ;       AX
                                ;
                                ;   NOTES
                                ;       This differs from "srch_block" in that it searches only the
                                ;       remaining portion of the config.sys image and leaves SI and CX
                                ;       pointing to where the search left off, and that it takes a pointer
                                ;       to search string in our own segment (DS:DI instead of ES:DI).
                                ;
                                ;   HISTORY
                                ;       Created 16-Mar-1992 by JeffPar
                                ;
```

```
                                ;--------------------------------------------------------------------------

                                find_block:
00004343 E8C802                         call    get_char        ; get line code
00004346 72FA                           jc  short find_exit      ; end of file
00004348 247F                           and     al,~CONFIG_OPTION_QUERY
0000434A 3C5B                           cmp     al,CONFIG_BEGIN ; beginning of a block?
0000434C 740C                           je  short check_line ; no
0000434E 3C4A                           cmp     al,CONFIG_INCLUDE
00004350 7513                           jne short next_line       ;
00004352 2E800E[6519]01                 or  byte [cs:config_multi],1
00004358 EB0B                           jmp     short next_line ;
                                check_line:
0000435A 2E800E[6519]01                 or      byte [cs:config_multi],1
00004360 E80700                         call    comp_names      ; compare block names
00004363 76DD                           jbe short find_exit       ; end of file, or names matched
                                next_line:
00004365 E88002                         call    skip_opt_line   ; no, so skip to next line
00004368 EBD9                           jmp short find_block  ;
                                ;find_exit:
                                ;   retn

                                ;--------------------------------------------------------------------------
                                ;
                                ;   comp_names:  compares keyword @DS:DI to position in config.sys @ES:SI
                                ;
                                ;   INPUT
                                ;    DS:DI -> keyword to compare
                                ;    ES:SI -> position in config.sys
                                ;       CX == remaining bytes in config.sys
                                ;
                                ;   OUTPUT
                                ;       ZF flag set, if match (also, CF set if EOF)
                                ;    ES:SI -> where the comparison stopped (at end of block name or EOF)
                                ;       CX == # bytes remaining from that point
                                ;
                                ;   OTHER REGS USED
                                ;       AX
                                ;
                                ;   NOTES
                                ;       None
                                ;
                                ;   HISTORY
                                ;       Created 16-Mar-1992 by JeffPar
                                ;
                                ;--------------------------------------------------------------------------

                                comp_names:
0000436A 57                             push    di              ;
                                comp_loop:                      ;
0000436B E8A002                         call    get_char        ;
0000436E 7210                           jc  short comp_exit       ;
00004370 E80704                         call    any_delim       ; is next character a delimiter?
00004373 8A25                           mov     ah,[di]         ; (get next character we're supposed to match)
00004375 740B                           je  short comp_almost ; yes, it *could* be a match
00004377 47                             inc     di              ;
00004378 25DFDF                         and     ax,~2020h ; 0DFDFh
                                                                ; BUGBUG -- assumes both names are alphanumeric -JTP
0000437B 38E0                           cmp     al,ah           ; match?
0000437D 74EC                           je  short comp_loop ; yes, keep looking at the characters
0000437F F8                             clc                     ; prevent erroneous eof indication: clear carry
                                comp_exit:                      ;
00004380 5F                             pop     di              ;
00004381 C3                             retn                      ;
                                comp_almost:                    ;
00004382 86C4                           xchg    al,ah           ; we don't know for sure if it's a match
00004384 E8F303                         call    any_delim       ; until we verify that the second string has
00004387 86C4                           xchg    al,ah           ; been exhausted also...
00004389 EBF5                           jmp     short comp_exit ; if we are, this call to any_delim will tell...

                                ;--------------------------------------------------------------------------

                                    ; 10/09/2023 - Retrodos v4.2 IO.SYS (Optimization)
                                comp_names_x:
                                    ;
                                comp_names_safe:
0000438B 50                             push    ax
0000438C 51                             push    cx
0000438D 56                             push       si
0000438E 1E                             push    ds
0000438F 0E                             push    cs
00004390 1F                             pop     ds
00004391 E8D6FF                         call    comp_names
00004394 1F                             pop     ds
00004395 5E                         pop     si
00004396 59                             pop     cx
00004397 58                             pop     ax
00004398 C3                             retn

                                ;--------------------------------------------------------------------------
                                ;
                                ;   print_item:  display menu item #BL
                                ;
                                ;   INPUT
                                ;       BL == menu item # to display
                                ;
                                ;   OUTPUT
                                ;       Menu item displayed, with appropriate highlighting if BL == bDefBlock
                                ;
                                ;   OTHER REGS USED
                                ;       None
                                ;
                                ;   NOTES
                                ;       This function saves/restores the current cursor position, so you
                                ;       needn't worry about it.
                                ;
                                ;   HISTORY
                                ;       Created 16-Mar-1992 by JeffPar
                                ;
                                ;--------------------------------------------------------------------------

                                    ; 04/01/2023 - Retro DOS v4.2 (Modified MSDOS 6.21 IO.SYS)
                                    ; (SYSINIT:485Ah)

                                print_item:         ; prints menu item #BL (1 to N)
00004399 50                             push    ax              ;
0000439A 53                             push    bx              ;
0000439B 51                             push    cx              ;
0000439C 52                             push    dx              ;
0000439D 56                             push    si              ;
0000439E B403                           mov     ah,03h          ; get cursor position
000043A0 8A3E[7D4C]                     mov     bh,[bMenuPage]  ; always page zero
000043A4 CD10                           int     10h             ; DH/DL = row/column
000043A6 52                             push    dx              ; save it
```

```
42504 000043A7 B402                             mov     ah,02h          ; set cursor position
42505 000043A9 88DE                             mov     dh,bl           ;
42506 000043AB 80C603                           add     dh,3            ;
42507 000043AE B205                             mov     dl,5            ;
42508 000043B0 CD10                             int     10h             ; set cursor position for correct row/col
42509 000043B2 88D8                             mov     al,bl           ;
42510 000043B4 0430                             add     al,'0'          ; convert menu item # to ASCII digit
42511 000043B6 8A26[7C4C]                       mov     ah,[bMenuColor] ; normal attribute
42512 000043BA 3A1E[864C]                       cmp     bl,[bDefBlock]  ; are we printing the current block?
42513 000043BE 7510                             jne short print_other ; no
42514 000043C0 80CC70                           or      ah,70h          ; yes, set bgnd color to white
42515 000043C3 88E5                             mov     ch,ah           ;
42516 000043C5 B104                             mov     cl,4            ;
42517 000043C7 D2C5                             rol     ch,cl           ;
42518 000043C9 38E5                             cmp     ch,ah           ; are fgnd/bgnd the same?
42519 000043CB 7503                             jne short print_other ; no
42520 000043CD 80F408                           xor     ah,08h          ; yes, so modify the fgnd intensity
42521                                   print_other:                    ;
42522 000043D0 B700                             mov     bh,0            ;
42523 000043D2 01DB                             add     bx,bx           ;
42524 000043D4 8BBF[AA4C]                       mov     di,[aoffBlockDesc+bx]
42525 000043D8 88E3                             mov     bl,ah           ; put the attribute in the correct register now
42526 000043DA 8A3E[7D4C]                       mov     bh,[bMenuPage]  ; get correct video page #
42527 000043DE B409                             mov     ah,09h          ; write char/attr
42528 000043E0 B90100                           mov     cx,1            ;
42529 000043E3 CD10                             int     10h             ;
42530 000043E5 FEC2                             inc     dl              ; increment column
42531 000043E7 B402                             mov     ah,02h          ;
42532 000043E9 CD10                             int     10h             ;
42533                                           ;mov       ax,0900h+'.'    ;
42534 000043EB B82E09                           mov ax,092Eh
42535 000043EE CD10                         int     10h             ; display '.'
42536 000043F0 FEC2                             inc     dl              ; increment column
42537 000043F2 B402                             mov     ah,02h          ;
42538 000043F4 CD10                             int     10h             ;
42539                                           ;mov       ax,0900h+' '    ;
42540 000043F6 B82009                           mov ax,0920h
42541 000043F9 CD10                         int     10h             ; display ' '
42542 000043FB FEC2                             inc     dl              ; increment column
42543 000043FD B402                             mov     ah,02h          ;
42544 000043FF CD10                             int     10h             ;
42545 00004401 06                               push    es              ;
42546                                   print_loop:                     ;
42547 00004402 268A05                           mov     al,[es:di]; get a character of the description
42548 00004405 47                               inc     di              ;
42549 00004406 3C09                             cmp     al,TAB ; 9; substitute spaces for tabs
42550 00004408 7502                             jne short print_nontab ;
42551 0000440A B020                             mov     al,' '          ;
42552                                   print_nontab:                   ;
42553 0000440C 3C20                             cmp     al,' '          ;
42554 0000440E 7215                             jb  short print_done ; stop at the 1st character < space
42555 00004410 3C24                             cmp     al,'$'          ;
42556 00004412 7411                             je  short print_done ; also stop on $
42557 00004414 B409                             mov     ah,09h          ; display function #
42558 00004416 CD10                             int     10h             ;
42559 00004418 FEC2                             inc     dl              ; increment column
42560 0000441A 80FA4E                           cmp     dl,78           ; far enough?
42561 0000441D 7306                             jae short print_done ; yes
42562 0000441F B402                             mov     ah,02h          ;
42563 00004421 CD10                             int     10h             ;
42564 00004423 EBDD                             jmp short  print_loop
42565                                   print_done:                     ;
42566 00004425 07                               pop     es              ;
42567 00004426 5A                               pop     dx              ;
42568 00004427 B402                             mov     ah,02h          ;
42569 00004429 CD10                             int     10h             ; restore previous row/col
42570 0000442B 5E                               pop     si              ;
42571 0000442C 5A                               pop     dx              ;
42572 0000442D 59                               pop     cx              ;
42573 0000442E 5B                               pop     bx              ;
42574 0000442F 58                               pop     ax              ;
42575 00004430 C3                               retn                      ;
42576
42577                                   ;---------------------------------------------------------------------------
42578                                   ;
42579                                   ;   select_item:  wait for user to select menu item, with time-out
42580                                   ;
42581                                   ;   INPUT
42582                                   ;       None
42583                                   ;
42584                                   ;   OUTPUT
42585                                   ;       BX == menu item # (1-N), or -1 for clean boot
42586                                   ;       Selected menu item highlighted
42587                                   ;       Cursor positioned beneath menu, ready for tty-style output now
42588                                   ;
42589                                   ;   OTHER REGS USED
42590                                   ;       None
42591                                   ;
42592                                   ;   NOTES
42593                                   ;       None
42594                                   ;
42595                                   ;   HISTORY
42596                                   ;       Created 16-Mar-1992 by JeffPar
42597                                   ;
42598                                   ;---------------------------------------------------------------------------
42599
42600                                   select_item:                ; returns digit value in BX (trashes AX/CX/DX)
42601 00004431 8A1E[864C]                       mov     bl,[bDefBlock]  ; BL will be the default block #
42602 00004435 88D8                             mov     al,bl           ;
42603 00004437 E83701                           call    disp_num        ;
42604 0000443A E84401                           call    show_status     ; display current interactive status
42605 0000443D 803E[8A4C]FF                     cmp     byte [secTimeOut],-1
42606 00004442 7452                             je  short input_key      ; no time-out, just go to input
42607 00004444 B42C                             mov     ah,GET_TIME ; 2Ch
42608 00004446 CD21                             int     21h             ;
42609 00004448 88F7                             mov     bh,dh           ; BH = initial # of seconds
42610                                   check_time:
42611 0000444A A0[8A4C]                         mov     al,[secTimeOut] ;
42612 0000444D 2A06[8B4C]                       sub     al,[secElapsed] ;
42613 00004451 730D                             jae short show_time      ;
42614 00004453 800E[854C]02                     or      byte [bQueryOpt],2  ; disable all further prompting
42615 00004458 C606[8B4C]00                     mov     byte [secElapsed],0
42616 0000445D E9F600                           jmp select_done    ; time's up!
42617                                   show_time:
42618 00004460 53                               push    bx              ;
42619 00004461 88C3                             mov     bl,al           ; save # in BL
42620 00004463 8A3E[7D4C]                       mov     bh,[bMenuPage]  ;
42621 00004467 B403                             mov     ah,03h          ; get cursor position
42622 00004469 CD10                             int     10h             ;
42623 0000446B 52                               push    dx              ;
42624 0000446C 80C208                       add     dl,8            ; move cursor to the right
42625 0000446F B402                             mov     ah,02h          ; set cursor position
42626 00004471 CD10                             int     10h             ;
42627 00004473 BA[2753]                         mov     dx,_$TimeOut
```

```
42628 00004476 E8DB05                   call    print           ; print the "Time remaining: " prompt
42629 00004479 88D8                     mov     al,bl           ; recover # from BL
42630 0000447B 98                       cbw                     ; this works because AL is always <= 90
42631 0000447C B10A                     mov     cl,10           ;
42632 0000447E F6F1                     div     cl              ; AL = tens digit, AH = ones digit
42633 00004480 88E1                     mov     cl,ah           ;
42634 00004482 0430                     add     al,'0'          ;
42635 00004484 B40E                     mov     ah,0Eh          ;
42636 00004486 CD10                     int     10h             ; write TTY tens digit
42637 00004488 88C8                     mov     al,cl           ;
42638 0000448A 0430                     add     al,'0'          ;
42639 0000448C B40E                     mov     ah,0Eh          ;
42640 0000448E CD10                     int     10h             ; write TTY ones digit
42641 00004490 5A                       pop     dx
42642 00004491 B402                     mov     ah,02h          ; set cursor position back to where it was
42643 00004493 CD10                     int     10h             ;
42644 00004495 5B                       pop     bx              ;
42645                           input_key:
42646 00004496 B406                     mov     ah,RAW_CON_IO ; 6
42647 00004498 B2FF                     mov     dl,0FFh         ; input request
42648 0000449A CD21                     int     21h             ;
42649 0000449C 751F                     jnz short got_key  ;
42650 0000449E 803E[8A4C]FF             cmp     byte [secTimeOut],-1; is there a time-out?
42651 000044A3 74F1                     je  short input_key       ; no, just go back to input
42652 000044A5 B42C                     mov     ah,GET_TIME     ;
42653 000044A7 CD21                     int     21h             ; DH = seconds
42654 000044A9 88F4                     mov     ah,dh           ;
42655 000044AB 28FE                     sub     dh,bh           ; should generally be zero or one
42656 000044AD 88E7                     mov     bh,ah           ;
42657 000044AF 7302                     jnc short got_time ;
42658 000044B1 B601                     mov     dh,1            ; it wrapped back to zero, so assume one
42659                           got_time:
42660 000044B3 08F6                     or      dh,dh           ; any change?
42661 000044B5 74DF                     jz  short input_key       ; no
42662 000044B7 0036[8B4C]               add     [secElapsed],dh ;
42663 000044BB EB8D                     jmp short check_time ;
42664                           got_key:
42665 000044BD 50                       push    ax              ;
42666 000044BE B8FFFF                   mov     ax,-1           ; zap both secTimeOut and secElapsed
42667 000044C1 8706[8A4C]               xchg    [secTimeOut],ax
42668 000044C5 3CFF                     cmp     al,-1           ; was time-out already disabled?
42669 000044C7 740E                     je  short timeout_disabled ; yes
42670 000044C9 53                       push    bx              ; let's disable # seconds display
42671 000044CA B8200A                   mov     ax,0A20h        ; write multiple spaces
42672 000044CD 8B1E[7C4C]               mov     bx,[bMenuColor]
42673 000044D1 B95000                   mov     cx,80           ; 80 of them, to be safe
42674 000044D4 CD10                     int     10h             ; to completely obliterate # seconds display
42675 000044D6 5B                       pop     bx                ;
42676
42677                           timeout_disabled:
42678 000044D7 58                       pop     ax              ;
42679 000044D8 08C0                     or      al,al           ; extended key pressed?
42680 000044DA 755A                     jnz short normal_key ; no
42681 000044DC CD21                     int     21h             ; get the next part of the key then
42682 000044DE 74B6                     jz  short input_key       ; hmmm, what happened to the second part?
42683
42684 000044E0 3C48                     cmp     al,48h          ; up arrow?
42685 000044E2 7510                     jne short not_up   ; no
42686 000044E4 80FB01                   cmp     bl,1            ; are we as up as up can get?
42687 000044E7 76AD                     jbe short input_key       ; yes, ignore it
42688 000044E9 FE0E[864C]               dec     byte [bDefBlock] ;
42689 000044ED E8A9FE                   call    print_item      ; re-print the current item
42690 000044F0 FECB                     dec     bl              ; and then print the new current item
42691 000044F2 EB12                     jmp     short print1
42692                           not_up:
42693 000044F4 3C50                     cmp     al,50h          ; down arrow?
42694 000044F6 7518                     jne short not_down ; no
42695 000044F8 3A1E[874C]               cmp     bl,[bMaxBlock]  ; are we as down as down can get?
42696 000044FC 7310                     jae short to_input_key ; yes, ignore it
42697 000044FE FE06[864C]               inc     byte [bDefBlock] ;
42698 00004502 E894FE                   call    print_item      ; re-print the current item
42699 00004505 43                       inc     bx              ; and then print the new current item
42700                           print1:
42701 00004506 88D8                     mov     al,bl           ;
42702                           print2:
42703 00004508 E88EFE                   call    print_item      ;
42704 0000450B E86300                   call    disp_num        ;
42705                           to_input_key:
42706 0000450E EB86                     jmp short input_key ; 10/09/2023
42707                           not_down:
42708 00004510 F606[814C]01             test    byte [bDisableUI],1
42709 00004515 75F7                     jnz short to_input_key ; don't allow F8 or F5
42710 00004517 3C42                     cmp     al,42h          ; F8 function key?
42711 00004519 750B                     jne short not_f8   ; no
42712 0000451B 8036[854C]01             xor     byte [bQueryOpt],1
42713 00004520 E85E00                   call    show_status     ;
42714 00004523 E970FF                   jmp     input_key ;
42715                           not_f8:
42716 00004526 3C3F                     cmp     al,3Fh          ; F5 function key?
42717 00004528 75E4                     jne short to_input_key ; no
42718                                   ; 02/08/2023 - Retro DOS v4.2 IO.SYS (optimization)
42719                                   ; MSDOS 6.21 IO.SYS - SYSINIT:49EBh
42720                                   ; (PCDOS 7.1 IBMBIO.COM - SYSINIT:4C32h)
42721 0000452A 800E[854C]04             or      byte [bQueryOpt],4 ; no more queries
42722 0000452F BBFFFF                   mov     bx,-1           ; special return code (-1) indicating clean boot
42723 00004532 B020                     mov     al,' '          ; don't want to display anything really;
42724 00004534 EB26                     jmp     short disp_input ; just want to display the cr/lf sequence...
42725
42726                           normal_key:
42727 00004536 3C0D                     cmp     al,0Dh          ; Enter?
42728 00004538 741C                     je  short select_done ; yes
42729 0000453A 3C08                     cmp     al,08h          ; backspace?
42730 0000453C 7504                     jne short not_backspace ; no
42731 0000453E BBFEFF                   mov     bx,-2 ; 0FFFEh    ; yes, special return code
42732 00004541 C3                       retn                      ;
42733                           not_backspace:
42734 00004542 2C30                     sub     al,'0'          ; is greater than '0'?
42735 00004544 76C8                     jbe short to_input_key ; no
42736 00004546 3A06[874C]               cmp     al,[bMaxBlock]  ; is less than or equal to the maximum digit?
42737 0000454A 77C2                     ja  short to_input_key ; no
42738 0000454C A2[864C]                 mov [bDefBlock],al  ;
42739 0000454F E847FE                   call    print_item      ; redisplay the current selection
42740 00004552 88C3                     mov     bl,al           ; set new selection
42741 00004554 EBB2                     jmp short print2
42742
42743                           select_done:
42744 00004556 B700                     mov     bh,0            ; return a full 16-bit value (for indexing)
42745 00004558 88D8                     mov     al,bl           ;
42746 0000455A 0430                     add     al,'0'          ; convert it into a digit, then display it
42747
42748                                   ; fall into disp_input
42749
42750                           ; 16/04/2019 - Retro DOS v4.0
42751
```

```
                                ;--------------------------------------------------------------------------
                                ;
                                ;   disp_input:  display a single character + cr/lf
                                ;
                                ;   INPUT
                                ;       AL == character to display
                                ;
                                ;   OUTPUT
                                ;       None
                                ;
                                ;   OTHER REGS USED
                                ;       None
                                ;
                                ;   NOTES
                                ;       This function is used not only for the menu input selection but
                                ;       also for the interactive line prompting (the y/n/a thing).
                                ;
                                ;   HISTORY
                                ;       Created 16-Mar-1992 by JeffPar
                                ;
                                ;--------------------------------------------------------------------------

                                disp_input:
0000455C 50                         push    ax
                                    ;cmp    al,' '
                                    ;jae    short disp_ok
                                    ;mov    al,' '
                                    ; 10/09/2023 - Retro DOS v4.2 IO:SYS (Optimization)
0000455D B220                       mov     dl,' ' ; 20h
0000455F 38D0                       cmp     al,dl
00004561 7602                       jna     short disp_input_ok
                                disp_ok:
00004563 88C2                       mov     dl,al
                                disp_input_ok:
00004565 B402                       mov     ah,STD_CON_OUTPUT ; 2
00004567 CD21                       int     21h
00004569 BA[7050]                   mov     dx,crlfm
0000456C E8E504                     call    print
0000456F 58                         pop     ax
00004570 C3                         retn

                                ;--------------------------------------------------------------------------

                                disp_num:
00004571 53                             push    bx
00004572 0430                           add     al,'0'
00004574 B40A                           mov     ah,0Ah
00004576 8B1E[7C4C]                     mov     bx,[bMenuColor]
0000457A B90100                         mov     cx,1
0000457D CD10                           int     10h
0000457F 5B                             pop     bx
00004580 C3                             retn

                                ;--------------------------------------------------------------------------
                                ;
                                ;   show_status:  display current interactive mode setting (on/off/none)
                                ;
                                ;   INPUT
                                ;       None
                                ;
                                ;   OUTPUT
                                ;       None
                                ;
                                ;   OTHER REGS USED
                                ;       None
                                ;
                                ;   NOTES
                                ;       None
                                ;
                                ;   HISTORY
                                ;       Created 16-Mar-1992 by JeffPar
                                ;
                                ;--------------------------------------------------------------------------

                                show_status:
00004581 53                             push    bx              ; BL = video page #
00004582 8B1E[7C4C]                     mov     bx,[bMenuColor]
00004586 B403                           mov     ah,03h          ; get cursor position
00004588 CD10                           int     10h             ;
0000458A 52                             push    dx              ; save it
0000458B B402                           mov     ah,02h          ; set cursor position
0000458D 8B16[7F4C]                     mov     dx,[bLastCol]   ; set correct row/col
00004591 F606[814C]01                   test    byte [bDisableUI],1
00004596 740C                           jz  short show_onoff ; just show on/off
00004598 B200                           mov     dl,0            ;
0000459A CD10                           int     10h             ;
0000459C B8200A                         mov     ax,0A20h        ; write multiple spaces
0000459F B95000                         mov     cx,80           ; 80 of them, to be exact
                                    ; 10/09/2023
                                    ;int    10h             ; to obliterate the status line
000045A2 EB11                           jmp     short show_done ;
                                show_onoff:
000045A4 CD10                           int     10h
                                            ; - VIDEO - WRITE CHARACTERS ONLY AT CURSOR POSITION
                                            ; AL = character, BH = display page - alpha mode
                                            ; BL = color of character (graphics mode, PCjr only)
                                            ; CX = number of times to write character

000045A6 A0[2353]                       mov     al,[_$NO]  ; assume OFF
000045A9 803E[854C]01                   cmp     byte [bQueryOpt],1 ; is interactive mode on?
000045AE 7503                           jne short show_noton ; no
000045B0 A0[1F53]                       mov     al,[_$YES]; yes
                                show_noton:                     ;
000045B3 B40E                           mov     ah,0Eh          ; write TTY
                                show_done:  ; 10/09/2023
000045B5 CD10                           int     10h             ;
                                ;show_done:                     ;
000045B7 5A                             pop     dx              ;
000045B8 B402                           mov     ah,02h          ;
000045BA CD10                           int     10h             ; restore original cursor position
000045BC 5B                             pop     bx              ;
000045BD C3                             retn                      ;

                                ; 16/04/2019 - Retro DOS v4.0

                                ;--------------------------------------------------------------------------
                                ;
                                ;   skip_token: advances ES:SI/CX past the current token
                                ;
                                ;   INPUT
                                ;    ES:SI -> position in config.sys
                                ;       CX == remaining bytes in config.sys
                                ;
```

```
                                ;   OUTPUT
                                ;       CF set if EOL/EOF hit
                                ;       AL == 1st char of delimiter
                                ;    ES:SI -> just past the delimiter
                                ;       CX == # bytes remaining from that point
                                ;
                                ;   OTHER REGS USED
                                ;       AX
                                ;
                                ;   NOTES
                                ;       None
                                ;
                                ;   HISTORY
                                ;       Created 16-Mar-1992 by JeffPar
                                ;
                                ;-----------------------------------------------------------------------------

                                skip_token:
000045BE E84D00                         call    get_char
000045C1 7210                           jc  short skip_token_done
000045C3 E8B401                         call    any_delim
000045C6 75F6                           jne short skip_token
                                skip_check_eol:
000045C8 3C0D                           cmp     al,cr ; 0Dh
000045CA 7406                           je  short skip_token_eol
000045CC 3C0A                           cmp     al,lf ; 0Ah
000045CE 7402                           je  short skip_token_eol
000045D0 F8                             clc
                                        ;jmp       short skip_token_done
000045D1 C3                         retn
                                skip_token_eol:
000045D2 F9                             stc
                                skip_token_done:
000045D3 C3                             retn

                                ;-----------------------------------------------------------------------------
                                ;
                                ;   skip_delim: advances ES:SI/CX past the current delimiter
                                ;
                                ;   INPUT
                                ;    ES:SI -> position in config.sys
                                ;       CX == remaining bytes in config.sys
                                ;
                                ;   OUTPUT
                                ;       CF set if EOF hit
                                ;       AL == 1st char of token
                                ;    ES:SI -> just past the token
                                ;       CX == # bytes remaining from that point
                                ;    ES:BX -> new token (since ES:SI is already pointing 1 byte past token)
                                ;
                                ;   OTHER REGS USED
                                ;       AX
                                ;
                                ;   NOTES
                                ;       None
                                ;
                                ;   HISTORY
                                ;       Created 16-Mar-1992 by JeffPar
                                ;
                                ;-----------------------------------------------------------------------------

                                skip_delim: ; returns carry set if eol/eof
000045D4 E83700                         call    get_char        ;
000045D7 8D5CFF                         lea     bx,[si-1]       ; also returns BX -> next token
000045DA 72F7                           jc  short skip_token_done ;
000045DC E8AB01                         call    delim           ;
000045DF 74F3                           je  short skip_delim ;
000045E1 EBE5                           jmp short skip_check_eol  ; 13/05/2019

                                ;-----------------------------------------------------------------------------
                                ;
                                ;   skip_opt_line: same as skip_line provided AL != LF
                                ;
                                ;   INPUT
                                ;       AL == last character read
                                ;    ES:SI -> position in config.sys
                                ;       CX == remaining bytes in config.sys
                                ;
                                ;   OUTPUT
                                ;       CF set if EOF hit
                                ;       AL == 1st char of new line
                                ;    ES:SI -> just past 1st char of new line
                                ;       CX == # bytes remaining from that point
                                ;
                                ;   OTHER REGS USED
                                ;       AX
                                ;
                                ;   NOTES
                                ;       In other words, the purpose here is to skip to the next line,
                                ;       unless ES:SI is already sitting at the front of the next line (which
                                ;       it would be if the last character fetched -- AL -- was a linefeed)
                                ;
                                ;   HISTORY
                                ;       Created 16-Mar-1992 by JeffPar
                                ;
                                ;-----------------------------------------------------------------------------

                                ; 03/08/2023 - Retro DOS v4.2 IO.SYS (optimization)
                                ;skip_opt_line:
                                ;   cmp     al,lf ; 0Ah
                                ;   je      short skip_line_done

                                    ; fall into skip_line

                                ;-----------------------------------------------------------------------------
                                ;
                                ;   skip_line: skip to the next line
                                ;
                                ;   INPUT
                                ;    ES:SI -> position in config.sys
                                ;       CX == remaining bytes in config.sys
                                ;
                                ;   OUTPUT
                                ;       CF set if EOF hit
                                ;    ES:SI -> just past 1st char of new line
                                ;       CX == # bytes remaining from that point
                                ;
                                ;   OTHER REGS USED
                                ;       AX
                                ;
                                ;   NOTES
                                ;       None
                                ;
                                ;   HISTORY
```

```
                                ;       Created 16-Mar-1992 by JeffPar
                                ;
                                ;--------------------------------------------------------------------------

                                skip_line:
000045E3 E82800                         call    get_char
000045E6 7204                           jc  short skip_line_done
                                skip_opt_line:      ; 03/08/2023 - Retro DOS v4.2 IO.SYS (optimization)
000045E8 3C0A                           cmp     al,lf ; 0Ah
000045EA 75F7                           jne short skip_line
                                skip_line_done:
                                num_done:   ; 18/12/2022
000045EC C3                             retn

                                ;--------------------------------------------------------------------------
                                ;
                                ;   get_number: return binary equivalent of numeric string
                                ;
                                ;   INPUT
                                ;    ES:SI -> position in config.sys
                                ;       CX == remaining bytes in config.sys
                                ;
                                ;   OUTPUT
                                ;       AL == non-digit encountered
                                ;       BX == binary #
                                ;    ES:SI -> just past 1st non-digit
                                ;       CX == # bytes remaining from that point
                                ;
                                ;   OTHER REGS USED
                                ;       AX
                                ;
                                ;   NOTES
                                ;       None
                                ;
                                ;   HISTORY
                                ;       Created 16-Mar-1992 by JeffPar
                                ;
                                ;--------------------------------------------------------------------------
                                ; 13/05/2019

                                get_number:
000045ED 29DB                           sub     bx,bx           ; BX = result
                                num_loop:
000045EF E81C00                         call    get_char        ;
000045F2 72F8                           jc  short num_done ;
000045F4 3C30                           cmp     al,'0'          ; convert to value
000045F6 72F4                           jb  short num_done ; no more number
000045F8 3C39                           cmp     al,'9'          ;
000045FA 77F0                           ja  short num_done ;
000045FC 50                             push    ax              ;
000045FD B80A00                         mov     ax,10           ;
00004600 52                             push    dx              ;
00004601 F7E3                           mul     bx              ;
00004603 5A                             pop     dx              ;
00004604 89C3                           mov     bx,ax           ;
00004606 58                             pop     ax              ;
00004607 2C30                           sub     al,'0'          ;
00004609 98                             cbw                     ;
0000460A 01C3                           add     bx,ax           ;
0000460C EBE1                           jmp short num_loop ;

                                    ; 18/12/2022
                                ;num_done:
                                        ;retn

                                ;--------------------------------------------------------------------------
                                ;
                                ;   get_char:  return next character, advance ES:SI, and decrement CX
                                ;
                                ;   INPUT
                                ;    ES:SI -> position in config.sys
                                ;       CX == remaining bytes in config.sys
                                ;
                                ;   OUTPUT
                                ;       AL == next character
                                ;    ES:SI -> just past next character
                                ;       CX == # bytes remaining from that point
                                ;
                                ;   OTHER REGS USED
                                ;       AX
                                ;
                                ;   NOTES
                                ;       None
                                ;
                                ;   HISTORY
                                ;       Created 16-Mar-1992 by JeffPar
                                ;
                                ;--------------------------------------------------------------------------

                                get_char:
0000460E 83E901                         sub     cx,1            ; use SUB to set carry,zero
00004611 7205                           jb  short get_fail ; out of data
                                        ;lods       byte ptr es:[si] ;
00004613 26                         es
00004614 AC                         lodsb
00004615 88C4                           mov     ah,al           ;
00004617 C3                             retn                      ;
                                get_fail:                       ; restore CX to zero
00004618 B90000                         mov     cx,0            ; leave carry set, zero not set
                                nearby_ret:
0000461B C3                             retn

                                ;--------------------------------------------------------------------------
                                ;
                                ;   query_user: ask user whether to execute current config.sys command
                                ;
                                ;   INPUT
                                ;       AL == current command code
                                ;    ES:SI -> current command line in config.sys
                                ;    config_cmd == current command code, but with QUERY bit intact
                                ;                  (00h used to generate "Process AUTOEXEC.BAT" prompt)
                                ;
                                ;   OUTPUT
                                ;       CF set if command should be ignored (it is also REM'ed out)
                                ;
                                ;   OTHER REGS USED
                                ;       BX, CX, DX, DI
                                ;
                                ;   NOTES
                                ;       None
                                ;
                                ;   HISTORY
                                ;       Created 16-Mar-1992 by JeffPar
```

```
43124                                   ;
43125                                   ;--------------------------------------------------------------------------
43126
43127                                       ; 31/12/2022 - Retro UNIX 386 v4.2 (Modified MSDOS 6.21 IO.SYS)
43128                                       ; (SYSINIT:4AE5h)
43129
43130                                       ; 12/12/2022
43131                                   query_user:
43132 0000461C F606[854C]04                       test    byte [bQueryOpt],4       ; answer no to everything?
43133                                       ; 01/01/2023
43134 00004621 7403                         jz      short qu_1              ;
43135 00004623 E9B900                       jmp     skip_all
43136                                       ; 12/12/2022
43137                                       ;;jmp   short skip_all          ;
43138                                       ;jnz    short skip_all
43139                                   qu_1:
43140 00004626 F606[854C]02                 test    byte [bQueryOpt],2      ; answer yes to everything?
43141 0000462B 75EE                             jnz short nearby_ret        ; yes (and return carry clear!)
43142 0000462D 50                               push    ax                      ;
43143 0000462E A0[6419]                         mov     al,[config_cmd]         ;
43144 00004631 F606[854C]01                     test    byte [bQueryOpt],1       ; query every command?
43145 00004636 7506                             jnz short query_all              ; yes
43146 00004638 A880                             test    al,CONFIG_OPTION_QUERY  ;
43147                                       ; 01/01/2023
43148 0000463A 7502                         jnz     short query_all              ;
43149                                       ; 12/12/2022
43150                                       ;;jmp   short do_cmd            ;
43151                                       ;jz     short do_cmd ; cf=0
43152
43153                                       ; 01/01/2023
43154 0000463C 58                           pop     ax
43155 0000463D C3                           retn
43156
43157                                   query_all:
43158
43159                                   ;   Search for the command code (AL) in "comtab", and then print
43160                                   ;   out the corresponding keyword, followed by the rest of the actual
43161                                   ;   line pointed to by ES:SI
43162
43163 0000463E 56                               push    si                      ; save pointer to rest of CONFIG.SYS line
43164 0000463F BA[8253]                         mov     dx,_$AutoPrmpt           ;
43165 00004642 247F                             and     al,~CONFIG_OPTION_QUERY ; 7Fh
43166 00004644 7450                             jz  short generic_prompt  ; config_cmd must have been 0
43167
43168 00004646 88C6                             mov     dh,al                   ; save config_cmd in DH
43169 00004648 29DB                             sub     bx,bx                   ;
43170 0000464A BF[D24C]                         mov     di,comtab           ;
43171                                   find_match:                             ;
43172 0000464D 8A1D                             mov     bl,[di]                 ; get size of current keyword
43173 0000464F 08DB                             or      bl,bl                   ;
43174 00004651 7425                             jz  short line_print      ; end of table
43175 00004653 47                               inc     di                      ;
43176 00004654 3A01                             cmp     al,[di+bx]              ; match?
43177 00004656 7405                             je  short cmd_match              ; yes
43178 00004658 8D7901                           lea     di,[di+bx+1]            ; otherwise, skip this command code
43179                                       ; 13/05/2019
43180 0000465B EBF0                             jmp short find_match      ; loop
43181                                   cmd_match:                              ;
43182 0000465D 8A4DFF                           mov     cl,[di-1]               ;
43183 00004660 B500                             mov ch,0                         ;
43184 00004662 B402                             mov     ah,STD_CON_OUTPUT ; 2
43185                                   cmd_print:                              ;
43186 00004664 8A05                             mov     al,[di]                 ;
43187 00004666 47                               inc     di                      ;
43188 00004667 88C2                             mov     dl,al                   ;
43189 00004669 CD21                             int     21h                     ;
43190 0000466B E2F7                             loop    cmd_print               ;
43191 0000466D B23D                             mov     dl,'='                  ;
43192 0000466F 80FE56                           cmp     dh,CONFIG_SET  ; 'V'    ; for SET commands, don't display a '='
43193 00004672 7502                             jne short cmd_notset      ;
43194 00004674 B220                             mov     dl,' '                  ;
43195                                   cmd_notset:
43196 00004676 CD21                             int     21h                     ; '=' looks funny on SET commands
43197                                   line_print:                             ;
43198                                       ;lods   byte ptr es:[si]        ;
43199 00004678 26                               es
43200 00004679 AC                           lodsb
43201 0000467A 08C0                         or       al,al                   ;
43202 0000467C 7502                             jnz short non_null        ;
43203 0000467E B020                             mov     al,' '                  ;
43204                                   non_null:                               ;
43205 00004680 3C20                             cmp     al,' '                  ; control code?
43206 00004682 720F                             jb  short prompt_user     ; yes, assume end of line
43207 00004684 7505                             jne short non_space              ;
43208                                           ; 10/09/2023
43209 00004686 263804                       cmp     [es:si],al ; 20h
43210                                       ;cmp    byte [es:si],' '        ;
43211 00004689 7208                             jb  short prompt_user     ;
43212                                   non_space:                              ;
43213 0000468B 88C2                             mov     dl,al                   ;
43214 0000468D B402                             mov     ah,STD_CON_OUTPUT ; 2    ;
43215 0000468F CD21                             int     21h                     ;
43216 00004691 EBE5                             jmp short line_print      ;
43217
43218                                   prompt_user:                            ;
43219 00004693 BA[1353]                         mov     dx,_$InterPrmpt          ;
43220
43221                                   generic_prompt:
43222 00004696 E8BB03                           call    print                   ;
43223                                   input_loop:                             ;
43224 00004699 B400                             mov     ah,0                    ; read a key
43225 0000469B CD16                             int     16h                     ;
43226 0000469D 08C0                             or      al,al                   ; is it a function key?
43227 0000469F 750F                             jnz short not_func        ; no
43228 000046A1 80FC3F                           cmp     ah,3Fh                  ; F5 function key?
43229 000046A4 75F3                             jne short input_loop      ; no
43230 000046A6 A0[2353]                         mov     al,[_$NO]                ;
43231 000046A9 800E[854C]04                     or      byte [bQueryOpt],4       ; no more queries
43232 000046AE EB21                             jmp     short legal_char        ;
43233                                   not_func:
43234 000046B0 24DF                             and     al,~20h ; 0DFh           ; converting to upper case
43235 000046B2 3A06[2353]                       cmp     al,[_$NO]         ; verify character is legal
43236 000046B6 7419                             je  short legal_char      ;
43237 000046B8 3A06[1F53]                       cmp     al,[_$YES]              ;
43238 000046BC 7413                             je  short legal_char      ;
43239 000046BE 803E[6419]00                     cmp     byte [config_cmd],0      ;
43240 000046C3 74D4                             je  short input_loop      ; don't allow Esc on this query
43241 000046C5 3C1B                             cmp     al,1Bh                  ; Esc?
43242 000046C7 75D0                             jne short input_loop      ;
43243 000046C9 800E[854C]02                     or      byte [bQueryOpt],2       ; no more interactive boot prompts
43244 000046CE A0[1F53]                         mov     al,[_$YES]
43245                                   legal_char:                             ;
43246 000046D1 E888FE                           call    disp_input              ;
43247 000046D4 5E                               pop     si                      ; restore pointer to rest of CONFIG.SYS line
```

```
43248
43249 000046D5 3A06[2353]                      cmp     al,[_$NO]         ; process line?
43250 000046D9 7403                            je  short skip_cmd        ; no
43251                                      ; 12/12/2022
43252 000046DB F8                          clc
43253                                  do_cmd:
43254 000046DC 58                          pop     ax                    ;
43255                                      ; 12/12/2022
43256                                      ; cf=0
43257                                      ;clc                          ; just do the command
43258 000046DD C3                          retn
43259
43260                                  skip_cmd:
43261 000046DE 58                          pop     ax                    ;
43262                                  skip_all:
43263 000046DF B430                        mov     ah,CONFIG_REM ; '0'   ; fake out the rest of sysinit's processing
43264 000046E1 F9                          stc
43265 000046E2 C3                          retn
43266
43267                                  ;--------------------------------------------------------------------------
43268                                  ;
43269                                  ;   print_error: displays multi-config error conditions
43270                                  ;
43271                                  ;   INPUT
43272                                  ;    Carry set to pause, clear to not
43273                                  ;    ES:SI -> current command line in config.sys
43274                                  ;
43275                                  ;   OUTPUT
43276                                  ;       None
43277                                  ;
43278                                  ;   OTHER REGS USED
43279                                  ;       None
43280                                  ;
43281                                  ;   NOTES
43282                                  ;       None
43283                                  ;
43284                                  ;   HISTORY
43285                                  ;       Created 16-Mar-1992 by JeffPar
43286                                  ;
43287                                  ;--------------------------------------------------------------------------
43288
43289                                  print_error:
43290 000046E3 50                              push    ax
43291 000046E4 53                              push    bx
43292 000046E5 51                              push    cx
43293 000046E6 52                              push    dx
43294 000046E7 1E                              push    ds
43295 000046E8 0E                              push    cs
43296 000046E9 1F                              pop     ds
43297 000046EA 9C                              pushf
43298 000046EB E820FC                          call    get_linenum
43299 000046EE 891E[AF02]                      mov     [linecount],bx
43300 000046F2 E8C5E7                          call    error_line
43301 000046F5 9D                              popf
43302 000046F6 7319                            jnc short pe_ret
43303 000046F8 BA[DD51]                        mov     dx,_$PauseMsg
43304 000046FB E85603                          call    print
43305 000046FE B8070C                          mov     ax,0C07h          ; flush input buffer, then wait for key
43306 00004701 CD21                            int     21h                       ; wait for a key
43307 00004703 08C0                            or      al,al                     ; extended key?
43308 00004705 7504                            jnz short pe_1            ; no
43309 00004707 B407                            mov     ah,07h                    ; yes
43310 00004709 CD21                            int     21h                       ; eat it too
43311                                  pe_1:
43312 0000470B BA[7050]                    mov     dx,crlfm
43313 0000470E E84303                          call    print
43314                                  pe_ret:
43315 00004711 1F                          pop     ds
43316 00004712 5A                              pop     dx
43317 00004713 59                              pop     cx
43318 00004714 5B                              pop     bx
43319 00004715 58                              pop     ax
43320 00004716 C3                          retn
43321
43322                                  ;--------------------------------------------------------------------------
43323
43324                                  ;   This function is very simple: it merely prepends a "/D" to the
43325                                  ;   command-line for the shell; this (undocumented) switch disables
43326                                  ;   AUTOEXEC.BAT processing and the date/time prompt that is usually
43327                                  ;   displayed when there's no AUTOEXEC.BAT.
43328
43329                                  disable_autoexec:
43330                                      ; MSDOS 6.21 IO.SYS -  SYSINIT:4BE2h
43331                                      ; 17/04/2019 - Retro DOS v4.0
43332
43333 00004717 F606[854C]04                  test    byte [bQueryOpt],4
43334 0000471C 7443                        jz      short disable_exit
43335 0000471E F606[7B4C]01                test    byte [dae_flag],1
43336 00004723 753C                        jnz     short disable_exit
43337 00004725 800E[7B4C]01                or      byte [dae_flag],1
43338                                          ;or byte [bQueryOpt],2 ; MSDOS 6.0
43339 0000472A 810E[854C]0201              or      word [bQueryOpt],102h ; [bDefBlock] = 1
43340 00004730 BA4420                      mov     dx,'D ' ; 2044h
43341                                  dae_1:
43342                                      ; 03/08/2023 - Retro DOS v4.2 IO.SYS (optimization)
43343 00004733 A0[BA4B]                        mov al,[def_swchr]
43344                                      ;mov    al,[command_line-1]     ; get default switchchar
43345 00004736 08C0                            or      al,al                   ; anything there?
43346 00004738 7427                            jz  short disable_exit    ; no, disable_autoexec already called
43347 0000473A 8A1E[BB4B]                      mov     bl,[command_line]       ;
43348 0000473E B700                            mov     bh,0                    ; BX == command-line length
43349 00004740 89D9                            mov     cx,bx                   ;
43350 00004742 80C303                          add     bl,3                    ;
43351 00004745 80FB7E                          cmp     bl,126                  ;
43352 00004748 7717                            ja  short disable_exit    ;
43353 0000474A 881E[BB4B]                      mov     [command_line],bl       ; update length
43354 0000474E 81C3[BC4B]                      add     bx,command_line+1 ; make sure we move the NULL too
43355 00004752 41                              inc     cx                      ; (just for consistency sake)
43356                                  disable_loop:                          ;
43357 00004753 8A67FD                          mov     ah,[bx-3]               ;
43358 00004756 8827                            mov     [bx],ah                 ;
43359 00004758 4B                              dec     bx                      ;
43360 00004759 E2F8                            loop    disable_loop            ;
43361 0000475B 8847FE                          mov     [bx-2],al               ;
43362                                      ;mov    word [bx-1],'D ' ; 2044h ; /D is stuffed into place now
43363 0000475E 8957FF                      mov     [bx-1],dx  ; MSDOS 6.21 IO.SYS - SYSINIT:4C29h
43364                                          ;mov       byte [command_line-1],0 ;
43365                                  disable_exit:                          ;
43366 00004761 C3                              retn
43367
43368                                  CheckQueryOpt:     ; MSDOS 6.21 IO.YSYS - SYSINIT:4C2Dh
43369 00004762 803E[854C]01                cmp     byte [bQueryOpt],1
43370 00004767 75F8                        jnz     short disable_exit
43371 00004769 F606[7B4C]02                test    byte [dae_flag],2
```

```
43372 0000476E 75F1                     jnz     short disable_exit
43373 00004770 800E[7B4C]02             or      byte [dae_flag],2
43374                                   ;mov    dx,' Y' ; (MASM syntax) ; 2059h
43375                                   ; 10/09/2023 (BugFix)
43376 00004775 BA5920                   mov     dx,'Y ' ; (NASM syntax) ; 2059h
43377 00004778 EBB9                     jmp     short dae_1
43378
43379                                 ;endif  ;MULTI_CONFIG
43380
43381                                 ;%endif    ; 02/11/2022
43382
43383
43384                                 ; 19/04/2019 - Retro DOS v4.0
43385
43386                                 ;--------------------------------------------------------------------------
43387                                 ;
43388                                 ; procedure : delim
43389                                 ;
43390                                 ;--------------------------------------------------------------------------
43391
43392                                 ; 01/01/2023 - Retro DOS v4.2 (Modified MSDOS 6.21 IO.SYS)
43393                                 ; (SYSINIT:4C45h)
43394
43395                                 ; 02/11/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
43396                                 ;%if 0
43397                                 ;;ifdef    MULTI_CONFIG
43398                                 ;
43399                                 any_delim:
43400 0000477A 3C0D                     cmp     al,cr
43401 0000477C 7427                     je      short delim_ret
43402 0000477E 3C0A                     cmp     al,lf
43403 00004780 7423                     je      short delim_ret
43404 00004782 3C5B                     cmp     al,'['
43405 00004784 741F                     je      short delim_ret
43406 00004786 3C5D                     cmp     al,']'
43407 00004788 741B                     je      short delim_ret
43408                                 ;
43409                                 ;;endif ;MULTI_CONFIG
43410                                 ;%endif ; 02/11/2022
43411
43412                                     ; 02/11/2022 - Retrodos v4.0 (Modified MSDOS 5.0 IO.SYS)
43413                                     ; (SYSINIT:3450h)
43414                                 delim:
43415 0000478A 3C2F                     cmp     al,'/'          ; ibm will assume "/" as an delimeter.
43416 0000478C 7417                     je      short delim_ret
43417
43418 0000478E 3C00                     cmp     al,0            ; special case for sysinit!!!
43419 00004790 7413                     je      short delim_ret
43420
43421                                 org_delim:                ; used by organize routine except for getting
43422 00004792 3C20                     cmp     al,' '          ; the filename.
43423 00004794 740F                     je      short delim_ret
43424 00004796 3C09                         cmp     al,tab ; 9
43425 00004798 740B                     je      short delim_ret
43426 0000479A 3C3D                     cmp     al,'='
43427 0000479C 7407                     je      short delim_ret
43428 0000479E 3C2C                     cmp     al,','
43429 000047A0 7403                     je      short delim_ret
43430 000047A2 3C3B                     cmp     al,';'
43431
43432                                     ; 02/11/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
43433
43434                                 ; 04/01/2023 - Retro DOS v4.2
43435                                 ;ifdef      MULTI_CONFIG
43436                                 ;   Make sure there's no chance of a false EOF indication
43437 000047A4 F8                       clc
43438                                 ;endif
43439                                     ; 02/11/2022
43440                                 delim_ret:
43441                                     ; 04/01/2023
43442                                     ; cf = 0
43443                                 nl_ret:     ; 10/09/2023
43444 000047A5 C3                       retn
43445
43446                                 ;--------------------------------------------------------------------------
43447                                 ;
43448                                 ; procedure : newline
43449                                 ;
43450                                 ;  newline returns with first character of next line
43451                                 ;
43452                                 ;--------------------------------------------------------------------------
43453
43454                                 newline:
43455 000047A6 E80600                   call    getchr              ;skip non-control characters
43456 000047A9 72FA                     jc      short nl_ret
43457 000047AB 3C0A                     cmp     al,lf               ;look for line feed
43458 000047AD 75F7                     jne     short newline
43459
43460                                     ; 10/09/2023 - Retro DOS v4.2 IO.SYS (Optimization)
43461                                     ;call   getchr
43462                                 ;nl_ret:
43463                                     ;retn
43464                                     ; 10/09/2023
43465                                     ;jmp    short getchr
43466
43467                                 ; 10/09/2023 - Retro DOS v4.2 IO.SYS (Optimization)
43468                                 %if 1
43469
43470                                 ;--------------------------------------------------------------------------
43471                                 ;
43472                                 ; procedure : getchr
43473                                 ;
43474                                 ;--------------------------------------------------------------------------
43475
43476                                     ; 24/10/2022
43477                                 getchr:
43478                                     ; 12/12/2022
43479                                     ;push   cx
43480                                     ;mov    cx,[count]
43481                                     ;jcxz   nochar
43482                                     ; 12/12/2022
43483 000047AF 833E[5603]01             cmp     word [count],1
43484 000047B4 720F                     jb      short nochar ; cf=1 ([count] = 0)
43485
43486 000047B6 8B36[5A03]                mov     si,[chrptr]
43487 000047BA 268A04                   mov     al,[es:si]
43488 000047BD FF0E[5603]               dec     word [count]
43489 000047C1 FF06[5A03]               inc     word [chrptr]
43490                                   ; 12/12/202
43491                                   ; cf=0
43492                                   ;clc
43493                                 ;get_ret:
43494                                   ;pop    cx
43495                                   ;retn
```

```
                                nochar:
                                    ; 12/12/2022
                                    ; cf=1
                                    ;stc
                                    ;jmp    short get_ret
                                
000047C5 C3                         retn
                                %endif
                                
                                ;--------------------------------------------------------------------------
                                ;
                                ; procedure : mapcase
                                ;
                                ;--------------------------------------------------------------------------
                                
                                    ; 02/11/2022 - Retro DOS 4.0 (Modified MSDOS 5.0 IO.SYS)
                                
                                    ; 04/01/2023 - Retro DOS 4.2 (Modified MSDOS 6.21 IO.SYS)
                                    ; (SYSINIT:4C7Eh)
                                mapcase:
000047C6 51                         push    cx
000047C7 56                         push    si
000047C8 1E                         push    ds
                                
000047C9 06                         push    es
000047CA 1F                         pop     ds
                                
                                    ; 02/11/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
                                
                                ; 04/01/2023 - Retro DOS 4.2
                                
                                ;ifdef      MULTI_CONFIG
000047CB 88C3                       mov     bl,al                   ; same cmd code this line
                                ;else
                                ;   xor     si,si
                                ;endif
                                    ; 02/11/2022
                                    ; 04/01/2023 - Retro DOS 4.2
                                    ;xor    si, si
                                
                                convloop:
000047CD AC                         lodsb
000047CE 3C61                       cmp     al,'a'
000047D0 7209                       jb      short noconv
000047D2 3C7A                       cmp     al,'z'
000047D4 7705                       ja      short noconv
000047D6 2C20                       sub     al,20h
000047D8 8844FF                     mov     [si-1],al
                                noconv:
                                
                                    ; 02/11/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
                                
                                ; 04/01/2023 - Retro DOS 4.2
                                ;ifdef      MULTI_CONFIG
                                
                                ;   When MULTI_CONFIG enabled, "mapcase" is used to map everything to
                                ;   upper-case a line at a time, after we've been able to figure out whether
                                ;   the line is a SET command or not (since we don't want to upper-case
                                ;   anything after the "=" in a SET)
                                ;
000047DB 80FB56                     cmp     bl,CONFIG_SET  ; 'V'  ; preserve case for part of the line?
000047DE 7504                       jne     short check_eol             ; no, just check for end-of-line
000047E0 3C3D                       cmp     al,'='                  ; separator between SET var and value?
000047E2 740A                       je      short convdone          ; yes
                                check_eol:
000047E4 3C0D                       cmp     al,cr
000047E6 7406                       je      short convdone
000047E8 3C0A                       cmp     al,lf
000047EA 7402                       je      short convdone
                                ;endif
                                    ; 02/11/2022
000047EC E2DF                       loop    convloop
                                convdone:
000047EE 1F                         pop     ds
000047EF 5E                         pop     si
000047F0 59                         pop     cx
000047F1 C3                         retn
                                
                                ;--------------------------------------------------------------------------
                                ;
                                ; procedure : round
                                ;
                                ; round the values in memlo and memhi to paragraph boundary.
                                ; perform bounds check.
                                ;
                                ;--------------------------------------------------------------------------
                                
                                round:
                                    ; 10/09/2023 - Retro DOS v4.2 IO.SYS (Optimization)
000047F2 1E                         push    ds
000047F3 0E                         push    cs
000047F4 1F                         pop     ds
                                
000047F5 50                         push    ax
                                    ;mov    ax,[cs:memlo]
000047F6 A1[6203]                   mov     ax,[memlo]
                                
000047F9 E819CB                     call    ParaRound               ; para round up
                                
                                    ;add    [cs:memhi],ax
000047FC 0106[6403]                 add     [memhi],ax
                                    ;mov    word [cs:memlo],0
00004800 C706[6203]0000             mov     word [memlo],0
                                    ;mov    ax,[cs:memhi]           ; ax = new memhi
00004806 A1[6403]                   mov     ax,[memhi]
                                    ;cmp    ax,[cs:ALLOCLIM]        ; if new memhi >= alloclim, error
00004809 3B06[A502]                 cmp     ax,[ALLOCLIM]
                                    ;jae    short mem_err
                                    ; 13/04/2024
0000480D 7322                       jae     short mem_err2 ; ds = cs
                                    ;test   byte [cs:setdevmarkflag],for_devmark ; 2
0000480F F606[6919]02               test    byte [setdevmarkflag],for_devmark ; 2
00004814 7416                       jz      short skip_set_devmarksize
00004816 06                         push    es
00004817 56                         push    si
                                    ;mov    si,[cs:devmark_addr]
00004818 8B36[6719]                 mov     si,[devmark_addr]
0000481C 8EC6                       mov     es,si
0000481E 29F0                       sub     ax,si
00004820 48                         dec     ax
                                    ;mov    [es:3],ax
00004821 26A30300                   mov     [es:devmark.size],ax  ; paragraph
                                    ;and    byte [cs:setdevmarkflag],not_for_devmark ; 0FDh
00004825 8026[6919]FD               and     byte [setdevmarkflag],not_for_devmark ; 0FDh
```

```
43620 0000482A 5E                       pop     si
43621 0000482B 07                       pop     es
43622                                 skip_set_devmarksize:
43623 0000482C 58                       pop     ax
43624
43625                                   ; 10/09/2023
43626 0000482D 1F                       pop     ds
43627
43628                                   ; 11/12/2022
43629                                   ; cf = 0
43630                                   ; 02/11/2022
43631                                   ;clc    ; ? (not needed here) ; clear carry
43632 0000482E C3                       retn
43633
43634                                 ;--------------------------------------------------------------------------
43635
43636                                 mem_err:
43637                                   ; 11/12/2022
43638 0000482F 0E                       push    cs
43639 00004830 1F                       pop     ds
43640                                 mem_err2:
43641 00004831 BA[4951]                 mov     dx,badmem
43642                                   ;push   cs
43643                                   ;pop    ds
43644 00004834 E81D02                   call    print
43645 00004837 E914CB                   jmp     stall
43646
43647                                 ;--------------------------------------------------------------------------
43648                                 ;
43649                                 ; procedure : calldev
43650                                 ;
43651                                 ;--------------------------------------------------------------------------
43652
43653                                   ; 02/11/2022 - Retrodos v4.0 (Modified MSDOS 5.0 IO.SYS)
43654                                   ; (SYSINIT:34E0h)
43655
43656                                   ; 13/04/2024 - Retrodos v5.0 (Modified PCDOS 7.1 IBMBIO.COM)
43657                                   ; (PCDOS 7.1 IBMBIO.COM - SYSINIT:4F3Eh)
43658
43659                                 calldev:
43660 0000483A 2E8E1E[3B24]             mov     ds,[cs:DevEntry+2]
43661 0000483F 2E031E[3924]             add     bx,[cs:DevEntry]      ; do a little relocation
43662 00004844 8B07                     mov     ax,[bx]
43663
43664 00004846 2EFF36[3924]              push    word [cs:DevEntry]
43665 0000484B 2EA3[3924]               mov     [cs:DevEntry],ax
43666 0000484F BB[6F03]                 mov     bx,packet
43667 00004852 2EFF1E[3924]             call    far [cs:DevEntry]
43668 00004857 2E8F06[3924]             pop     word [cs:DevEntry]
43669 0000485C C3                       retn
43670
43671                                 ;--------------------------------------------------------------------------
43672                                 ;
43673                                 ; procedure : todigit
43674                                 ;
43675                                 ;--------------------------------------------------------------------------
43676
43677                                 todigit:
43678 0000485D 2C30                     sub     al,'0'
43679                                   ;jb     short notdig  ; 02/11/2022
43680                                   ; 12/12/2022
43681 0000485F 7203                     jb      short notdig2
43682                                   ;cmp    al,9
43683                                   ;ja     short notdig
43684                                   ;clc
43685                                   ;retn
43686                                   ; 12/12/2022
43687 00004861 3C0A                     cmp     al,10
43688 00004863 F5                       cmc
43689                                 notdig:
43690                                   ;stc
43691                                 notdig2:
43692 00004864 C3                       retn
43693
43694                                 ;--------------------------------------------------------------------------
43695                                 ;
43696                                 ; procedure : getnum
43697                                 ;
43698                                 ; getnum parses a decimal number.
43699                                 ; returns it in ax, sets zero flag if ax = 0 (may be considered an
43700                                 ; error), if number is bad carry is set, zero is set, ax=0.
43701                                 ;
43702                                 ;--------------------------------------------------------------------------
43703
43704                                 getnum:
43705 00004865 53                       push    bx
43706 00004866 31DB                     xor     bx,bx                 ; running count is zero
43707                                 b2:
43708 00004868 E8F2FF                   call    todigit               ; do we have a digit ?
43709 0000486B 7247                     jc      short badnum          ; no, bomb
43710
43711 0000486D 93                       xchg    ax,bx                 ; put total in ax
43712 0000486E 53                       push    bx                    ; save digit (0 to 9)
43713                                   ;mov    bx,10                 ; base of arithmetic
43714                                   ; 12/12/2022
43715 0000486F B30A                     mov     bl,10
43716 00004871 F7E3                     mul     bx                    ; shift by one decimal digit
43717 00004873 5B                       pop     bx                    ; get back digit (0 to 9)
43718 00004874 00D8                     add     al,bl                 ; get total
43719 00004876 80D400                   adc     ah,0                  ; make that 16 bits
43720 00004879 7239                     jc      short badnum          ; too big a number
43721
43722 0000487B 93                       xchg    ax,bx                 ; stash total
43723
43724 0000487C E830FF                   call    getchr                ;get next digit
43725 0000487F 722D                     jc      short b1              ; no more characters
43726 00004881 3C20                     cmp     al,' '                ; space?
43727 00004883 741F                     je      short b15             ; then end of digits
43728 00004885 3C2C                     cmp     al,','                ; ',' is a seperator!!!
43729 00004887 741B                     je      short b15             ; then end of digits.
43730 00004889 3C09                     cmp     al, tab ; 9           ; tab
43731 0000488B 7417                     je      short b15
43732 0000488D 2E3A06[AE02]             cmp     al,[cs:sepchr]        ; allow 0 or special separators
43733 00004892 7410                     je      short b15
43734 00004894 3C2F                     cmp     al,'/'                ; see if another switch follows
43735                                   ; 12/12/2022
43736                                   ; 02/11/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
43737                                   ;nop                           ; cas - remnant of old bad code
43738                                   ;nop
43739 00004896 740C                     je      short b15
43740 00004898 3C0A                     cmp     al,lf                 ; line-feed?
43741 0000489A 7408                     je      short b15
43742 0000489C 3C0D                     cmp     al,cr                 ; carriage return?
43743 0000489E 7404                     je      short b15
```

```
000048A0 08C0                       or      al,al                   ; end of line separator?
000048A2 75C4                       jnz     short b2                ; no, try as a valid char...
                                b15:
000048A4 2EFF06[5603]               inc     word [cs:count]                 ; one more character to s...
000048A9 2EFF0E[5A03]               dec     word [cs:chrptr]        ; back up over separator
                                b1:
000048AE 89D8                       mov     ax,bx                   ; get proper count
000048B0 09C0                       or      ax,ax                   ; clears carry, sets zero accordingly
000048B2 5B                         pop     bx
000048B3 C3                         retn
                                badnum:
                                    ; 02/11/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
                                    ;mov    byte [cs:sepchr],0
000048B4 31C0                       xor     ax,ax                   ; set zero flag, and ax = 0
                                    ; 12 /12/2022
000048B6 2EA2[AE02]                 mov     [cs:sepchr],al ; 0
000048BA 5B                         pop     bx
000048BB F9                         stc                             ; and carry set
000048BC C3                         retn

                                ;******************************************************************************

                                setdoscountryinfo:

                                ;------------------------------------------------------------------------------
                                ;input: es:di -> pointer to dos_country_cdpg_info
                                ;   ds:0  -> buffer.
                                ;   si = 0
                                ;   ax = country id
                                ;   dx = code page id. (if 0, then use ccsyscodepage as a default.)
                                ;   bx = file handle
                                ;   this routine can handle maximum 438 country_data entries.
                                ;
                                ;output: dos_country_cdpg_info set.
                                ;    carry set if any file read failure or wrong information in the file.
                                ;    carry set and cx = -1 if cannot find the matching country_id,
                                ;    codepage_id in the file.
                                ;------------------------------------------------------------------------------

                                ; 04/01/2023 - Retrodos v4.2 (Modified MSDOS 6.21 IO.SYS)
                                ; (SYSINIT:4D83h)

                                ; 10/09/2023 - Retrodos v4.2 IO.SYS (Optimization)
                                ; (PCDOS 7.1 IBMBIO.COM - SYSINIT:4FCAh)

000048BD 57                     push    di
000048BE 50                     push    ax
000048BF 52                     push    dx

000048C0 31C9                   xor     cx,cx
000048C2 31D2                   xor     dx,dx
000048C4 B80002                 mov     ax,512                  ;read 512 bytes
000048C7 E84301                 call    readincontrolbuffer     ;read the file header
000048CA 724A                   jc      short setdosdata_fail

000048CC 06                     push    es
000048CD 56                     push    si

000048CE 0E                     push    cs
000048CF 07                     pop     es

000048D0 BF[204B]               mov     di,country_file_signature ; db 0FFh,'COUNTRY'
000048D3 B90800                 mov     cx,8                    ;length of the signature
000048D6 F3A6                   repz    cmpsb

000048D8 5E                     pop     si
000048D9 07                     pop     es
000048DA 753A                   jnz     short setdosdata_fail ;signature mismatch

000048DC 83C612                 add     si,18                   ;si -> county info type
000048DF 803C01                 cmp     byte [si],1             ;only accept type 1 (currently only 1 header type)
000048E2 7532                   jne     short setdosdata_fail ;cannot proceed. error return

000048E4 46                     inc     si                      ;si -> file offset
000048E5 8B14                   mov     dx,[si]                 ;get the info file offset.
000048E7 8B4C02                 mov     cx,[si+2]
000048EA B80018                 mov     ax,6144                 ;read 6144 bytes.
000048ED E81D01                 call    readincontrolbuffer     ;read info
000048F0 7224                   jc      short setdosdata_fail

000048F2 8B0C                   mov     cx,[si]                 ;get the # of country, codepage combination entries
000048F4 81F9B601               cmp     cx,438                  ;cannot handle more than 438 entries.
000048F8 771C                   ja      short setdosdata_fail

000048FA 46                     inc     si
000048FB 46                     inc     si                      ;si -> entry information packet
000048FC 5A                     pop     dx                      ;restore code page id
000048FD 58                     pop     ax                      ;restore country id
000048FE 5F                     pop     di

                                setdoscntry_find:               ;search for desired country_id,codepage_id.
000048FF 3B4402                 cmp     ax,[si+2]               ;compare country_id
00004902 7509                   jne     short setdoscntry_next

                                ;cmp    dx,0                    ;no user specified code page ?
                                ;je     short setdoscntry_any_codepage ;then no need to match code page id.
                                ; 10/09/2023
00004904 09D2                   or      dx,dx ; cmp dx,0
00004906 7413                   jz      short setdoscntry_any_codepage
00004908 3B5404                 cmp     dx,[si+4]               ;compare code page id
0000490B 7411                   je      short setdoscntry_got_it

                                setdoscntry_next:
0000490D 0334                   add     si,[si]                 ;next entry
0000490F 46                     inc     si
00004910 46                     inc     si                      ;take a word for size of entry itself
00004911 E2EC                   loop    setdoscntry_find

                                ;mov    cx,-1                   ;signals that bad country id entered.
                                ; 10/09/2023
00004913 49                     dec     cx ; 0 -> -1
                                setdoscntry_fail:
00004914 F9                     stc
00004915 C3                     retn

                                setdosdata_fail:
00004916 5E                     pop     si
00004917 59                     pop     cx
00004918 5F                     pop     di
00004919 EBF9                   jmp     short setdoscntry_fail

                                setdoscntry_any_codepage:       ;use the code_page_id of the country_id found.
0000491B 8B5404                 mov     dx,[si+4]
```

```
43868                                   setdoscntry_got_it:                     ;found the matching entry
43869 0000491E 2E8916[284B]                 mov     [cs:cntrycodepage_id],dx ;save code page id for this country.
43870 00004923 8B540A                       mov     dx,[si+10]              ;get the file offset of country data
43871 00004926 8B4C0C                       mov     cx,[si+12]
43872 00004929 B80002                       mov     ax,512                  ;read 512 bytes
43873 0000492C E8DE00                       call    readincontrolbuffer
43874 0000492F 72E3                         jc      short setdoscntry_fail
43875
43876 00004931 8B0C                         mov     cx,[si]                 ;get the number of entries to handle.
43877 00004933 46                           inc     si
43878 00004934 46                           inc     si                      ;si -> first entry
43879
43880                                   setdoscntry_data:
43881 00004935 57                           push    di                      ;es:di -> dos_country_cdpg_info
43882 00004936 51                           push    cx                      ;save # of entry left
43883 00004937 56                           push    si                      ;si -> current entry in control buffer
43884
43885 00004938 8A4402                       mov     al,[si+2]               ;get data entry id
43886 0000493B E8A400                       call    getcountrydestination ;get the address of destination in es:di
43887 0000493E 727C                         jc      short setdoscntry_data_next ;no matching data entry id in dos
43888
43889 00004940 8B5404                       mov     dx,[si+4]               ;get offset of data
43890 00004943 8B4C06                       mov     cx,[si+6]
43891 00004946 B80042                       mov     ax,4200h
43892 00004949 F9                           stc
43893 0000494A CD21                         int     21h                     ;move pointer
43894 0000494C 72C8                         jc      short setdosdata_fail
43895
43896 0000494E BA0002                       mov     dx,512                  ;start of data buffer
43897 00004951 B91400                       mov     cx,20                   ;read 20 bytes only. we only need to
43898 00004954 B43F                         mov     ah,3Fh                  ;look at the length of the data in the file.
43899 00004956 F9                           stc
43900 00004957 CD21                         int     21h                     ;read the country.sys data
43901 00004959 72BB                         jc      short setdosdata_fail ;read failure
43902
43903 0000495B 39C8                         cmp     ax,cx
43904 0000495D 75B7                         jne     short setdosdata_fail ; 13/05/2019
43905
43906 0000495F 8B5404                       mov     dx,[si+4]               ;get offset of data again.
43907 00004962 8B4C06                       mov     cx,[si+6]
43908 00004965 B80042                       mov     ax,4200h
43909 00004968 F9                           stc
43910 00004969 CD21                         int     21h                     ;move pointer back again
43911 0000496B 72A9                         jc      short setdosdata_fail
43912
43913 0000496D 56                           push    si
43914 0000496E BE0802                       mov     si,(512+8)              ;get length of the data from the file
43915 00004971 8B0C                         mov     cx,[si]
43916 00004973 5E                           pop     si
43917 00004974 BA0002                       mov     dx,512                  ;start of data buffer
43918 00004977 83C10A                       add     cx,10                   ;signature + a word for the length itself
43919 0000497A B43F                         mov     ah,3Fh                  ;read the data from the file.
43920 0000497C F9                           stc
43921 0000497D CD21                         int     21h
43922 0000497F 7295                         jc      short setdosdata_fail
43923
43924 00004981 39C8                         cmp     ax,cx
43925 00004983 7591                         jne     short setdosdata_fail
43926
43927 00004985 8A4402                       mov     al,[si+2]               ;save data id for future use.
43928 00004988 BE0802                       mov     si,(512+8)              ;si-> data buffer + id tag field
43929 0000498B 8B0C                         mov     cx,[si]                 ;get the length of the file
43930 0000498D 41                           inc     cx                      ;take care of a word for lenght of tab
43931 0000498E 41                           inc     cx                      ;itself.
43932 0000498F 81F9F805                     cmp     cx,(2048-512-8)         ; 1528 ;fit into the buffer?
43933 00004993 7781                         ja      short setdosdata_fail
43934
43935                                   ;if     bugfix
43936 00004995 E83100                   call    setdbcs_before_copy
43937                                   ;endif
43938
43939 00004998 3C01                         cmp     al,SetCountryInfo ; 1 ;is the data for setcountryinfo table?
43940 0000499A 7511                         jne     short setdoscntry_mov ;no, don't worry
43941
43942 0000499C 26FF7518                     push    word [es:di+country_cdpg_info.ccMono_Ptr-country_cdpg_info.ccCountryInfoLen]
43943                                   ;push   word [es:di+24]                 ;cannot destroy ccmono_ptr address. save them.
43944 000049A0 26FF751A                    push    word [es:di+country_cdpg_info.ccMono_Ptr-country_cdpg_info.ccCountryInfoLen+2]
43945                                   ;push   word [es:di+26]                 ;at this time di -> cccountryinfolen
43946
43947 000049A4 57                          push    di                      ;save di
43948
43949                                   ;push   ax
43950                                   ;mov    ax,[cs:cntrycodepage_id] ;do not use the code page info in country_info
43951                                   ;mov    [si+4],ax               ;use the saved one for this !!!!
43952                                   ;pop    ax
43953                                   ; 10/09/2023
43954 000049A5 2EFF36[284B]            push    word [cs:cntrycodepage_id]
43955 000049AA 8F4404                  pop     word [si+4]
43956
43957                                   setdoscntry_mov:
43958 000049AD F3A4                         rep     movsb                   ;copy the table into dos
43959 000049AF 3C01                         cmp     al,SetCountryInfo       ;was the ccmono_ptr saved?
43960 000049B1 7509                         jne     short setdoscntry_data_next
43961
43962 000049B3 5F                            pop     di                      ;restore di
43963 000049B4 268F451A                     pop     word [es:di+country_cdpg_info.ccMono_Ptr-country_cdpg_info.ccCountryInfoLen+2]
43964                                   ;pop    word [es:di+26]                 ;restore
43965 000049B8 268F4518                     pop     word [es:di+country_cdpg_info.ccMono_Ptr-country_cdpg_info.ccCountryInfoLen]
43966                                   ;pop    word [es:di+24]
43967
43968                                   setdoscntry_data_next:
43969 000049BC 5E                           pop     si                      ;restore control buffer pointer
43970 000049BD 59                           pop     cx                      ;restore # of entries left
43971 000049BE 5F                           pop     di                      ;restore pointer to dso_country_cdpg
43972 000049BF 0334                         add     si,[si]                 ;try to get the next entry
43973 000049C1 46                           inc     si
43974 000049C2 46                           inc     si                      ;take a word of entry length itself
43975 000049C3 49                           dec     cx
43976                                       ; 10/09/2023
43977 000049C4 741B                         jz      short setdoscntry_ok
43978                                       ;cmp    cx,0
43979                                       ;je     short setdoscntry_ok
43980 000049C6 E96CFF                       jmp     setdoscntry_data
43981
43982                                       ; 18/12/2022
43983                                   ;setdoscntry_ok:
43984                                       ;retn
43985
43986                                   ;--------------------------------------------------------------------------
43987
43988                                       ;if     bugfix
43989
43990                                       ; 10/09/2023 - Retrodos v4.2 IO.SYS (Optimization)
43991
```

```
                                setdbcs_before_copy:
000049C9 3C07                       cmp     al,SetDBCS ; 7          ; dbcs vector set?
000049CB 7514                       jne     short sdbcsbc           ; jump if not
                                    ; 10/09/2023
000049CD 50                         push    ax
000049CE 31C0                       xor     ax,ax
000049D0 263905                     cmp     [es:di],ax ; 0
000049D3 740B                       je      short sdbcsbc_pop
                                    ;cmp    word [es:di],0          ; zero byte data block?
                                    ;je     short sdbcsbc           ; jump if so
000049D5 57                         push    di
                                    ; 10/09/2023
                                    ;push   ax
000049D6 51                         push    cx
000049D7 268B0D                     mov     cx,[es:di]              ; load block length
                                    ;add    di,2                    ; points actual data
000049DA 47                         inc     di
000049DB 47                         inc     di
                                    ;xor    al,al                   ; fill bytes
000049DC F3AA                       rep     stosb                   ; clear data block
000049DE 59                         pop     cx
                                    ;pop    ax
000049DF 5F                         pop     di
                                sdbcsbc_pop:        ; 10/09/2023
000049E0 58                         pop     ax
                                sdbcsbc:
                                setdoscntry_ok:     ; 18/12/2022
000049E1 C3                         retn
                                    ;endif
                                ;--------------------------------------------------------------------------
                                getcountrydestination:
                                ;--------------------------------------------------------------------------
                                ;get the destination address in the dos country info table.
                                ;
                                ;input: al - data id
                                ;   es:di -> dos_country_cdpg_info
                                ;on return:
                                ;   es:di -> destination address of the matching data id
                                ;   carry set if no matching data id found in dos.
                                ;--------------------------------------------------------------------------
                                    ; 04/01/2023 - Retrodos v4.2 (Modified MSDOS 6.21 IO.SYS)
                                    ; (SYSINIT:4EB2h)
                                    ; 10/09/2023 - Retrodos v4.2 IO.SYS (Optimization)
                                    ; (PCDOS 7.1 IBMBIO.COM - SYSINIT:50F9h)
000049E2 51                         push    cx
                                    ;add    di,74
000049E3 83C74A                     add     di,country_cdpg_info.ccNumber_of_entries
                                                                    ;skip the reserved area, syscodepage etc.
000049E6 268B0D                     mov     cx,[es:di]              ;get the number of entries
000049E9 47                         inc     di
000049EA 47                         inc     di                      ;si -> the first start entry id
                                getcntrydest:
000049EB 263805                     cmp     byte [es:di],al
000049EE 7413                       je      short getcntrydest_ok
000049F0 26803D01                   cmp     byte [es:di],SetCountryInfo ;was it setcountryinfo entry?
000049F4 7405                       je      short getcntrydest_1
000049F6 83C705                     add     di,5                    ;next data id
000049F9 EB03                       jmp     short getcntrydest_loop
                                getcntrydest_1:
                                    ;add    di,41
000049FB 83C729                     add     di,NEW_COUNTRY_SIZE+3 ;next data id
                                getcntrydest_loop:
000049FE E2EB                       loop    getcntrydest
00004A00 F9                         stc
                                    ;jmp    short getcntrydest_exit
                                getcntrydest_exit:
                                    ; 10/09/2023
00004A01 59                         pop     cx
00004A02 C3                         retn
                                getcntrydest_ok:
                                    ; 10/09/2023
00004A03 47                         inc     di
                                ;   cmp     al,SetCountryInfo ; 1 ;select country info?
                                ;   jne     short getcntrydest_ok1
                                ;
                                ;   ;inc    di                      ;now di -> cccountryinfolen
                                ;   jmp     short getcntrydest_exit
                                    ; 10/09/2023
00004A04 3C01                       cmp     al,SetCountryInfo ; 1 ;select country info?
00004A06 74F9                       je      short getcntrydest_exit
                                getcntrydest_ok1:
                                    ;les    di,[es:di+1]            ;get the destination in es:di
                                    ; 10/09/2023
00004A08 26C43D                     les     di,[es:di]
                                ;getcntrydest_exit:
00004A0B 59                         pop     cx
00004A0C C3                         retn
                                ;--------------------------------------------------------------------------
                                readincontrolbuffer:
                                ;--------------------------------------------------------------------------
                                ;move file pointer to cx:dx
                                ;read ax bytes into the control buffer. (should be less than 2 kb)
                                ;si will be set to 0 hence ds:si points to the control buffer.
                                ;
                                ;entry:  cx,dx offset from the start of the file where the read/write pointer
                                ;    be moved.
                                ;    ax - # of bytes to read
                                ;    bx - file handle
                                ;    ds - buffer seg.
                                ;return: the control data information is read into ds:0 - ds:0200.
                                ;    cx,dx value destroyed.
                                ;    carry set if error in reading file.
```

```
                        ;--------------------------------------------------------------------------
                        
00004A0D 50                 push    ax                  ;# of bytes to read
00004A0E B80042             mov     ax,4200h
00004A11 F9                 stc
00004A12 CD21               int     21h                 ;move pointer
00004A14 59                 pop     cx                  ;# of bytes to read
00004A15 7209               jc      short ricb_exit

00004A17 31D2               xor     dx,dx               ;ds:dx -> control buffer
00004A19 31F6               xor     si,si
00004A1B B43F               mov     ah,3Fh              ;read into the buffer
00004A1D F9                 stc
00004A1E CD21               int     21h                 ;should be less than 1024 bytes.
                        ricb_exit:
00004A20 C3                 retn

                        ;--------------------------------------------------------------------------

                        ;! set_country_path procedure is not called from anywhere !
                        ; Erdogan Tan - 04/08/2023
                        %if 0

                        set_country_path:

                        ;--------------------------------------------------------------------------
                        ;in:  ds - sysinitseg, es - confbot, si -> start of the asciiz path string
                        ;     dosinfo_ext, cntry_drv, cntry_root, cntry_path
                        ;     assumes current directory is the root directory.
                        ;out: ds:di -> full path (cntry_drv).
                        ;     set the cntry_drv string from the country=,,path command.
                        ;     ds, es, si value saved.
                        ;--------------------------------------------------------------------------

                            ; 04/01/2023 - Retrodos v4.2 (Modified MSDOS 6.21 IO.SYS)
                            ; (SYSINIT:4EF4h)

                            ; 10/09/2023 - Retrodos v4.2 IO.SYS (Optimization)
                            ; (Retrodos v5.0 Pre-Works)
                            ; (PCDOS 7.1 IBMBIO.COM - SYSINIT:513Bh)

                            push    si

                            push    ds                  ;switch ds, es
                            push    es
                            pop     ds
                            pop     es                  ;now ds -> confbot, es -> sysinitseg

                            call    chk_drive_letter    ;current ds:[si] is a drive letter?
                            jc      short scp_default_drv ;no, use current default drive.

                            mov     al,[si]
                            inc     si
                            inc     si                  ;si -> next char after ":"
                            jmp     short scp_setdrv

                        scp_default_drv:
                            mov     ah,19h
                            int     21h
                            add     al,"A"              ;convert it to a character.

                        scp_setdrv:
                            mov     [cs:cntry_drv],al   ;set the drive letter.
                            mov     di,cntry_path
                            mov     al,[si]
                            cmp     al, "\"
                            je      short scp_root_dir

                            cmp     al,"/"              ;let's accept "/" as an directory delim
                            ;je     short scp_root_dir
                            ;jmp    short scp_path
                            ; 04/01/2023
                            jne     short scp_path

                        scp_root_dir:
                            dec     di                  ;di -> cntry_root
                        scp_path:
                            call    move_asciiz         ;copy it

                            mov     di,cntry_drv
                        scpath_exit:

                            push    ds                  ;switch ds, es
                            push    es
                            pop     ds
                            pop     es                  ;ds, es value restored

                            pop     si
                            retn

                        ;--------------------------------------------------------------------------

                        chk_drive_letter:

                        ;check if ds:[si] is a drive letter followed by ":".
                        ;assume that every alpha character is already converted to upper case.
                        ;carry set if not.

                            ; 04/01/2023 - Retrodos v4.2

                            push    ax
                            cmp     byte [si],"A"
                            ;jb     short cdletter_no
                            jb      short cdletter_exit
                            cmp     byte [si],"Z"
                            ja      short cdletter_no
                            cmp     byte [si+1],":"
                            ;jne    short cdletter_no
                            ;jmp    short cdletter_exit
                            ; 04/01/2023
                            je      short cdletter_exit

                        cdletter_no:
                            stc
                        cdletter_exit:
                            pop     ax
                            retn

                        %endif

                        ;--------------------------------------------------------------------------

                        move_asciiz:
```

```
                                ;in: ds:si -> source es:di -> target
                                ;out: copy the string until 0.
                                ;assumes there exists a 0.

                                    ; 10/09/2023 - Retrodos v4.2 IO.SYS (Optimization)
                                    ; (MSDOS 6.21 IO.SYS - SYSINIT:4F40h)
                                    ; (PCDOS 7.1 IBMBIO.COM - SYSINIT:5187h)

                                masciiz_loop:
                                    ; 10/09/2023
00004A21 F604FF                     test   byte [si],0FFh
00004A24 A4                         movsb
                                    ;cmp    byte [si-1],0  ; was it 0?
                                    ;jne    short masciiz_loop
00004A25 75FA                       jnz    short masciiz_loop ; 10/09/2023
00004A27 C3                         retn

                                ;--------------------------------------------------------------------------

                                ;   ds:dx points to string to output (asciz)
                                ;
                                ;   prints <badld_pre> <string> <badld_post>

                                badfil:
00004A28 0E                         push   cs
00004A29 07                         pop    es

00004A2A 89D6                       mov    si,dx
                                badload:
00004A2C BA[B350]                   mov    dx,badld_pre   ; want to print config error
00004A2F BB[7050]                   mov    bx,crlfm
                                prnerr:
00004A32 0E                         push   cs
00004A33 1F                         pop    ds ; *
00004A34 E81D00                     call   print
                                prn1:
00004A37 268A14                     mov    dl,[es:si]
00004A3A 08D2                       or     dl,dl
00004A3C 7407                       jz     short prn2
00004A3E B402                       mov    ah,STD_CON_OUTPUT ; 2
00004A40 CD21                       int    21h
00004A42 46                         inc    si
00004A43 EBF2                       jmp    short prn1
                                prn2:
00004A45 89DA                       mov    dx,bx
00004A47 E80A00                     call   print
                                    ; 11/12/2022
                                    ; ds = cs ; *
00004A4A 803E[5503]01               cmp    byte [donotshownum],1
                                                        ; suppress line number when handling command.com
                                    ;cmp    byte [cs:donotshownum],1
00004A4F 7407                       je     short prnexit

                                    ; 18/12/2022
                                    ;call   error_line
00004A51 E966E4                     jmp    error_line
                                ;prnexit:
                                    ;retn

                                ;--------------------------------------------------------------------------

                                print:
00004A54 B409                       mov    ah,STD_CON_STRING_OUTPUT ; 9
00004A56 CD21                       int    21h
                                prnexit:    ; 18/12/2022
00004A58 C3                         retn

                                ;--------------------------------------------------------------------------

                                ;  open device pointed to by dx, al has access code
                                ;   if unable to open do a device open null device instead

                                    ; 02/11/2022 - Retrodos v4.0 (Modified MSDOS 5.0 IO.SYS)
                                    ; (SYSINIT:3764h)
                                open_dev:
00004A59 E80500                     call   open_file
00004A5C 7309                       jnc    short open_dev3

                                open_dev1:
00004A5E BA[C14A]                   mov    dx,nuldev
                                    ; 18/12/2022
                                    ;call   open_file
                                ;of_retn:
                                    ;retn
                                    ; 18/12/2022
                                    ;jmp    short open_file
                                open_file:
00004A61 B43D                       mov    ah,OPEN ; 3Dh
00004A63 F9                         stc
00004A64 CD21                       int    21h
                                of_retn:    ; 18/12/2022
00004A66 C3                         retn

                                open_dev3:
00004A67 89C3                       mov    bx,ax                ; handle from open to bx
                                    ;;xor   ax,ax                ; get device info
                                    ;;mov   ah,IOCTL ; 44h
                                    ;mov    ax,(IOCTL<<8) ; 13/05/2019
                                    ; 02/11/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
                                    ;xor    ax,ax
                                    ;mov    ah,44h  ; IOCTL
                                    ; 11/12/2022
00004A69 B80044                     mov    ax,4400h ; IOCTL<<8

00004A6C CD21                       int    21h

00004A6E F6C280                     test   dl,10000000b ; 80h
00004A71 75F3                       jnz    short of_retn

00004A73 B43E                       mov    ah,CLOSE ; 3Eh
00004A75 CD21                       int    21h
00004A77 EBE5                       jmp    short open_dev1

                                ;--------------------------------------------------------------------------

                                ; 18/12/2022
                                %if 0
                                open_file:
                                    mov    ah,OPEN ; 3Dh
                                    stc
                                    int    21h
                                    retn
                                %endif
```

```
                                ;--------------------------------------------------------------------------

                                ; test int24. return back to dos with the fake user response of "fail"

                                int24:
00004A79 B003                       mov     al,3                    ; fail the system call
00004A7B CF                         iret                            ; return back to dos.

                                ; 19/04/2019 - Retro DOS v4.0

                                ;--------------------------------------------------------------------------
                                ; DATA
                                ;--------------------------------------------------------------------------

                                ;include copyrigh.inc                   ; copyright statement

                                ; MSDOS 6.21 IO.SYS - SYSINIT:4FA3h

                                ;MsDosVersion6Copyr:
                                ;   db      'MS DOS Version 6 (C)Copyright 1981-1993 Microsoft Corp '
                                ;   db      'Licensed Material - Property of Microsoft All rights reserved '

                                ; 22/10/2022
                                ; MSDOS 5.0 IO.SYS - SYSINIT:378Ch

                                ; 28/12/2022
                                %if 0
                                    ; 02/11/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
                                MsDosVersion5Copyr:
                                    db      'MS DOS Version 5.00 (C)Copyright 1981-1991 Microsoft Corp '
                                    db      'Licensed Material - Property of Microsoft All rights reserved '
                                %endif

                                ; 13/04/2024 - Retro DOS v5.0
                                ; PCDOS 7.1 IBMBIO.COM - SYSINIT:51EAh   (IBMBIO.COM offset 42266)
                                %if 0
                                IBMDOSV71COPYR:
                                    db 'IBM DOS Version 7.1 (C)Copyright 1981-2002 IBM Corporation '
                                    db 'Licensed Material - Property of IBM All rights reserved '
                                %endif

                                ; 02/11/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
                                ; 22/10/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 IO.SYS)
                                ; 20/04/2019 - Retro DOS v4.0
                                ;BOOTMES:
                                ;   db      13
                                ;   db      10
                                ;   db      "MS-DOS version "
                                ;   db      MAJOR_VERSION + "0"
                                ;   db      "."
                                ;   db      (MINOR_VERSION / 10) + "0"
                                ;   db      (MINOR_VERSION % 10) + "0"
                                ;   db      13,10
                                ;   ;db     "Copyright 1981-1993 Microsoft Corp.",13,10,"$"
                                ;   ; 22/10/2022
                                ;   db      "Copyright 1981-1991 Microsoft Corp.",13,10,"$"
                                ;   ;
                                ;   db      0

                                    ; 01/01/2023 - Retro DOS v4.2

                                    ; 28/12/2022 - Retro DOS v4.1
                                ;MsDosVersion5Copyr:
                                ;   db      13,10,"MS DOS Version 5.0"
                                ;   db      13,10,"Copyright 1981-1991 Microsoft Corp.",13,10,"$",0

                                    ; 12/12/2022
00004A7C 00                         db      0
                                ; 12/12/2022
                                BOOTMES:
00004A7D 0D0A                       db      13,10
                                    ;;;db   "Retro DOS v4.0 (Modified MSDOS 5.0) "
                                    ; 28/12/2022
                                    ;;db    "Retro DOS v4.1 (Modified MSDOS 5.0) "
                                    ; 01/01/2023
                                    ;db     "Retro DOS v4.2 (Modified MSDOS 6.22) "
                                    ; 30/12/2023
00004A7F 526574726F20444F53-        db      "Retro DOS v5.0 (Modified PCDOS 7.1) "
00004A88 2076352E3020284D6F-
00004A91 646966696564205043-
00004A9A 444F5320372E312920

00004AA3 0D0A                       db      13,10
00004AA5 6279204572646F6761-        db      "by Erdogan Tan [2024] " ; 01/01/2024
00004AAE 6E2054616E205B3230-
00004AB7 32345D20
00004ABB 0D0A                       db      13,10
00004ABD 0D0A2400                   db      13,10,"$",0

00004AC1 4E554C00               nuldev:     db      "NUL",0
00004AC5 434F4E00               condev:     db      "CON",0
00004AC9 41555800               auxdev:     db      "AUX",0
00004ACD 50524E00               prndev:     db      "PRN",0

                                ;IFDEF      CONFIGPROC
00004AD1 5C434F4E4649472E53-    config:     db      "\CONFIG.SYS",0
00004ADA 595300

00004ADD 413A                   cntry_drv:  db      "A:"
00004ADF 5C                     cntry_root: db      "\"
00004AE0 434F554E5452592E53-    cntry_path: db      "COUNTRY.SYS",0
00004AE9 595300
                                            ;db 52 dup (0)
00004AEC 00<rep 34h>                        times 52 db 0

                                country_file_signature:
00004B20 FF434F554E545259           db      0FFh,'COUNTRY'

                                cntrycodepage_id:
00004B28 0000                       dw      0

                                ;ENDIF ; CONFIGPROC

                                ; 01/01/2023 - Retro DOS v4.2 (Modified MSDOS 6.21 IO.SYS)
                                ; (SYSINIT:5081h)

                                ; 02/11/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
                                ;ifdef      MULTI_CONFIG
00004B2A 00                     newcmd:  db 0                       ; non-zero if non-std shell specified
00004B2B 40                     tmplate: db 64                        ; must precede commnd
                                ;endif

                                ;ifdef ROMEXEC
                                ;   db      7                         ; size of commnd line (excl. null)
```

```
                                ;commnd: db "COMMAND",0
                                ;   db      56 dup (0)
                                ;else
                                    ; 02/11/2022
00004B2C 0C                         db      12                      ; size of commnd line (excl. null)
00004B2D 5C434F4D4D414E442E-     commnd:    db      "\COMMAND.COM",0
00004B36 434F4D00
                                    ;db     51 dup (0)
00004B3A 00<rep 33h>                times   51 db 0
                                ;endif

                                ; 02/11/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
                                ;ifdef      MULTI_CONFIG
00004B6D 5C434F4D4D414E442E-     commnd2: db        "\COMMAND.COM",0      ; alternate commands to exec,
00004B76 434F4D00
00004B7A 022F5000                    db     2,"/P",0              ; followed by their respective alternate
00004B7E 5C4D53444F535C434F-     commnd3: db "\MSDOS\COMMAND.COM",0; command lines
00004B87 4D4D414E442E434F4D-
00004B90 00
00004B91 0B413A5C4D53444F53-         db     11,"A:\MSDOS /P",0    ;(the drive letter are dynamically replaced)
00004B9A 202F5000
00004B9E 5C444F535C434F4D4D-     commnd4: db "\DOS\COMMAND.COM",0  ;
00004BA7 414E442E434F4D00
00004BAF 09413A5C444F53202F-         db     9,"A:\DOS /P",0               ;
00004BB8 5000
                                def_swchr:
00004BBA 00                          db     0                     ; default switchchar (referenced as command_line-1)
                                ;endif
                                    ; 30/10/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
                                command_line:
00004BBB 022F50                      db     2,"/P"                ; default command.com args
                                     ;db    125 dup (0)
00004BBE 00<rep 7Dh>                 times  125 db 0

                                pathstring:
                                     ;db    64 dup (0)
00004C3B 00<rep 40h>                 times  64 db 0

                                ; 01/01/2023 - Retro DOS v4.2 (Modified MSDOS 6.21 IO.SYS)
                                ; (SYSINIT:51D3h)
                                ; 02/11/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
                                ;%if 0

                                dae_flag:
00004C7B 00                         db      0 ; MSDOS 6.21 IO.SYS - SYSINIT:51D2h

                                ;ifdef      MULTI_CONFIG

                                ; 04/03/2022- Retro DOS v4.2 (Modified MSDOS 6.21 IO.SYS, SYSINIT)
                                MAX_MULTI_CONFIG equ 9    ; max # of multi-config menu items supported

                                ;   Beware of byte pairs accessed as words (see all "KEEP AFTER" notes below)

00004C7C 07                     bMenuColor: db      07h ; 1Fh       ; default fgnd/bgnd color
00004C7D 00                     bMenuPage:  db      0               ; menu video page (KEEP AFTER bMenuColor)
00004C7E 05                                 db      5               ; video page function # (KEEP AFTER bMenuPage)
00004C7F 00                     bLastCol:   db      0               ; ending column on status line
00004C80 18                     bLastRow:   db      24              ; row # of status line (KEEP AFTER bLastCol)
00004C81 00                     bDisableUI: db      0               ; 1=disable clean/interactive
                                                                        ; 2=disable default 2-second delay
00004C82 00                     bCRTPage:   db      0               ; value saved from BIOS data area
00004C83 0000                   wCRTStart:  dw      0               ; value saved from BIOS data area
00004C85 00                     bQueryOpt:  db      0               ; 0=off, 1=prompt all, 2=prompt none, 4=skip all
00004C86 01                     bDefBlock:  db      1               ; default block #
00004C87 00                     bMaxBlock:  db      0               ; maxmimum block #
00004C88 0000                   offDefBlock:        dw      0               ; offset of name of default block (if any)
00004C8A FF                     secTimeOut: db      -1 ; 0FFh       ; # of seconds for timeout (-1 == indefinite)
00004C8B 00                     secElapsed: db      0               ; # of seconds elapsed so far (KEEP AFTER secTimeOut)
00004C8C 00<rep Ah>             abBlockType:        times MAX_MULTI_CONFIG+1 db 0 ; array of block types
00004C96 0000<rep Ah>           aoffBlockName:      times MAX_MULTI_CONFIG+1 dw 0 ; array of offsets of block names
00004CAA 0000<rep Ah>           aoffBlockDesc:      times MAX_MULTI_CONFIG+1 dw 0 ; array of offsets of block descriptions

00004CBE 434F4E4649473D00       szBoot:             db      "CONFIG=",0
00004CC6 4D454E5500             szMenu:             db      "MENU",0
00004CCB 434F4D4D4F4E00         szCommon:   db      "COMMON",0

                                ;endif      ;MULTI_CONFIG

                                    ; 10/09/2023
                                    ; MSDOS 6.21 IO.SYS - SYSINIT:5229h
                                    ; (PCDOS 7.1 IBMBIO.COM - SYSINIT:546Eh)

                                comtab:     ; label byte

                                ;           cmd len    command         cmd code
                                ;           -------    -------         --------
                                ;ifdef MULTI_CONFIG
00004CD2 015B5B                             db      1,      "[",            CONFIG_BEGIN
                                ;endif
00004CD5 05425245414B43                     db      5,      "BREAK",        CONFIG_BREAK
00004CDC 074255464645525342                 db      7,      "BUFFERS",      CONFIG_BUFFERS
00004CE5 07434F4D4D454E5459                 db      7,      "COMMENT",      CONFIG_COMMENT
00004CEE 07434F554E54525951                 db      7,      "COUNTRY",      CONFIG_COUNTRY
00004CF7 0644455649434544                   db      6,      "DEVICE",       CONFIG_DEVICE
00004CFF 0A4445564943454849-                db      10,     "DEVICEHIGH",   CONFIG_DEVICEHIGH
00004D08 474855
00004D0B 03444F5348                         db      3,      "DOS",          CONFIG_DOS
00004D10 0844524956504152 4D-               db      8,      "DRIVPARM",     CONFIG_DRIVPARM
00004D19 50
00004D1A 044643425358                       db      4,      "FCBS",         CONFIG_FCBS
00004D20 0546494C455346                     db      5,      "FILES",        CONFIG_FILES
                                ;ifdef MULTI_CONFIG
00004D27 07494E434C5544454A                 db      7,      "INCLUDE",      CONFIG_INCLUDE
                                ;endif
00004D30 07494E5354414C4C49                 db      7,      "INSTALL",      CONFIG_INSTALL
00004D39 0B494E5354414C4C48-                db      11,     "INSTALLHIGH",  CONFIG_INSTALLHIGH
00004D42 49474857
00004D46 094C4153544524956-                 db      9,      "LASTDRIVE",    CONFIG_LASTDRIVE
00004D4F 454C
                                ;ifdef MULTI_CONFIG
00004D51 075355424D454E554F                 db      7,      "SUBMENU",      CONFIG_SUBMENU
00004D5A 094D454E55434F4C4F-                db      9,      "MENUCOLOR",    CONFIG_MENUCOLOR
00004D63 5252
00004D65 0B4D454E5544454641-                db      11,     "MENUDEFAULT",  CONFIG_MENUDEFAULT
00004D6E 554C5441
00004D72 084D454E554954454D-                db      8,      "MENUITEM",     CONFIG_MENUITEM
00004D7B 45
                                ;endif
00004D7C 0A4D554C5449545241-                db      10,     "MULTITRACK",   CONFIG_MULTITRACK
00004D85 434B4D
                                ;ifdef MULTI_CONFIG
00004D88 074E554D4C4F434B4E                 db      7,      "NUMLOCK",      CONFIG_NUMLOCK
                                ;endif
```

```
44590 00004D91 0352454D30                       db      3,      "REM",          CONFIG_REM
44591                                   ;ifdef MULTI_CONFIG
44592 00004D96 0353455456                       db      3,      "SET",          CONFIG_SET
44593                                   ;endif
44594 00004D9B 055348454C4C53                   db      5,      "SHELL",        CONFIG_SHELL
44595                                   ;if    STACKSW
44596 00004DA2 06535441434B534B                 db      6,      "STACKS",       CONFIG_STACKS
44597                                   ;endif
44598 00004DAA 085357495443484553-              db      8,      "SWITCHES",     CONFIG_SWITCHES
44598 00004DB3 31
44599                                       ; 18/03/2025 (BugFix)
44600                                       ;db    0
44601
44602                                       ; 10/09/2023
44603                                   ;aDosdata:  ; PCDOS 7.1 IBMBIO.COM - SYSINIT:5550h
44604                                       ; 13/04/2024 - Retro DOS v5.0
44605 00004DB4 07444F534441544154           db      7,      "DOSDATA",      CONFIG_DOSDATA ; 'T'
44606 00004DBD 00                           db      0
44607
44608                                   ;%endif ; 02/11/2022
44609
44610                                   ; 01/01/2023 - Retro DOS v4.2
44611                                   %if 0
44612
44613                                   comtab:
44614                                       ; 02/11/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
44615                                       ; (SYSINIT:38EDh)
44616                                       db      7,      "BUFFERS",      CONFIG_BUFFERS
44617                                       db      5,      "BREAK",        CONFIG_BREAK
44618                                       db      6,      "DEVICE",       CONFIG_DEVICE
44619                                       db      10,     "DEVICEHIGH",   CONFIG_DEVICEHIGH
44620                                       db      5,      "FILES",        CONFIG_FILES
44621                                       db      4,      "FCBS",         CONFIG_FCBS
44622                                       db      9,      "LASTDRIVE",    CONFIG_LASTDRIVE
44623                                       db      10,     "MULTITRACK",   CONFIG_MULTITRACK
44624                                       db      8,      "DRIVPARM",     CONFIG_DRIVPARM
44625                                       db      6,      "STACKS",       CONFIG_STACKS
44626                                       db      7,      "COUNTRY",      CONFIG_COUNTRY
44627                                       db      5,      "SHELL",        CONFIG_SHELL
44628                                       db      7,      "INSTALL",      CONFIG_INSTALL
44629                                       db      7,      "COMMENT",      CONFIG_COMMENT
44630                                       db      3,      "REM",          CONFIG_REM
44631                                       db      8,      "SWITCHES",     CONFIG_SWITCHES
44632                                       db      3,      "DOS",          CONFIG_DOS
44633                                       db     0
44634
44635                                   %endif
44636
44637                                   ; 01/01/2023 - Retro DOS v4.2 (Modified MSDOS 6.21 IO.SYS)
44638                                   ; (SYSINIT:530Ch)
44639
44640                                   ; 13/04/2024 - Retro DOS v5.0
44641                                   ; (PCDOS 7.1 IBMBIO.COM - SYSINIT:555Ah)
44642
44643                                   deviceparameters:
44644                                       ; A_DEVICEPARAMETERS <0,dev_3inch720kb,0,80>
44645                                   devp.specialfunc:  ; deviceparameters +
44646 00004DBE 00                           db      0       ; A_DEVICEPARAMETERS.DP_SPECIALFUNCTIONS
44647                                   devp.devtype:
44648 00004DBF 02                           db      2       ; A_DEVICEPARAMETERS.DP_DEVICETYPE
44649                                   devp.devattr:
44650 00004DC0 0000                         dw      0       ; A_DEVICEPARAMETERS.DP_DEVICEATTRIBUTES
44651                                   devp.cylinders:
44652 00004DC2 5000                         dw      80      ; A_DEVICEPARAMETERS.DP_CYLINDERS
44653
44654                                   ; 04/08/2023 - Retro DOS v4.2 IO.SYS (optimization)
44655
44656                                       ;times  286     db 0
44657                                   devp.mediatype:             ; A_DEVICEPARAMETERS.DP_MEDIATYPE
44658 00004DC4 00                           db      0
44659                                   devp.bpb:           ; A_DEVICEPARAMETERS.DP_BPB
44660                                   devp.bps:           ; A_DEVICEPARAMETERS.DP_BPB+A_BPB.BYTESPERSECTOR
44661 00004DC5 0000                         dw      0
44662                                   devp.secperclus:    ; A_DEVICEPARAMETERS.DP_BPB+A_BPB.SECTORSPERCLUSTER
44663 00004DC7 00                           db      0
44664 00004DC8 0000                         dw      0       ; A_DEVICEPARAMETERS.DP_BPB+A_BPB.RESERVEDSECTORS
44665 00004DCA 00                           db      0       ; A_DEVICEPARAMETERS.DP_BPB+A_BPB.NUMBEROFFATS
44666 00004DCB 0000                         dw      0       ; A_DEVICEPARAMETERS.DP_BPB+A_BPB.ROOTENTRIES
44667                                   devp.totalsecs:             ; A_DEVICEPARAMETERS.DP_BPB+A_BPB.TOTALSECTORS
44668 00004DCD 0000                         dw      0
44669                                   devp.mediaid:               ; A_DEVICEPARAMETERS.DP_BPB+A_BPB.MEDIADESCRIPTOR
44670 00004DCF 00                           db      0
44671 00004DD0 0000                         dw      0       ; A_DEVICEPARAMETERS.DP_BPB+A_BPB.SECTORSPERFAT
44672                                   devp.spt:           ; A_DEVICEPARAMETERS.DP_BPB+A_BPB.SECTORSPERTRACK
44673 00004DD2 0000                         dw      0
44674                                   devp.heads:         ; A_DEVICEPARAMETERS.DP_BPB+A_BPB.HEADS
44675 00004DD4 0000                         dw      0
44676
44677                                       ; 13/04/2024 - Retro DOS v5.0
44678                                       ; (PCDOS 7.1 IBMBIO.COM)
44679 00004DD6 00<rep 44h>                  times   68 db 0 ; PCDOS 7.1 (FAT32 BPB)
44680                                       ;;times 14 db 0 ; MSDOS 6.21
44681                                       ;dw     0       ; A_DEVICEPARAMETERS.DP_BPB+A_BPB.HIDDENSECTORS
44682                                       ;dw     0
44683                                       ;dw     0       ; A_DEVICEPARAMETERS.DP_BPB+A_BPB.BIGTOTALSECTORS
44684                                       ;dw     0
44685                                       ;times  6 db 0
44686
44687                                   devp.trktblents:
44688 00004E1A 0000                         dw      0       ; A_DEVICEPARAMETERS.DP_TRACKTABLEENTRIES
44689                                   devp.sectbl:                ; A_DEVICEPARAMETERS.DP_SECTORTABLE
44690 00004E1C 00<rep FCh>                  times   252 db 0 ; MAX_SECTORS_IN_TRACK * A_SECTORTABLE.size
44691                                                       ; 63*4 bytes
44692
44693                                   ; 01/01/2023 - Retro DOS v4.2 (Modified MSDOS 6.21 IO.SYS)
44694                                   ; (SYSINIT:5430h)
44695
44696                                   ; 13/04/2024 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMBIO.COM)
44697                                   ; (PCDOS 7.1 IBMBIO.COM - SYSINIT:56B4h)
44698
44699 00004F18 0200                   hlim:       dw      2
44700 00004F1A 0900                   slim:       dw      9
44701
44702 00004F1C 00                     drive:      db      0
44703
44704                                 switches:
44705 00004F1D 0000                     dw      0
44706
44707                                 ; 01/01/2023 - Retro DOS v4.2 (Modified MSDOS 6.21 IO.SYS)
44708                                 ; (SYSINIT:5437h)
44709
44710                                 ; the following are the recommended bpbs for the media that
44711                                 ; we know of so far.
44712
```

```
                                ; 02/11/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 IO.SYS)
                                ; MSDOS 5.0 IO.SYS - SYSINIT:3AA9h

                                ; 27/12/2023 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMBIO.COM)
                                ; PCDOS 7.1 IBMBIO.COM - SYSINIT:56BBh

                                ; 48 tpi diskettes

00004F1F 0002                   bpb48t:     dw      512
00004F21 02                         db      2
00004F22 0100                       dw      1
00004F24 02                         db      2
00004F25 7000                       dw      112
00004F27 D002                       dw      2*9*40 ; 720
00004F29 FD                         db      0FDh
00004F2A 0200                       dw      2
00004F2C 0900                       dw      9
00004F2E 0200                       dw      2
00004F30 00000000                   dd      0
00004F34 00000000                       dd      0
                                    ; 27/12/2023
00004F38 00<rep 1Ch>                times   28 db 0             ; FAT32 extensions (to BDS)
00004F54 90                         db      90h

                                ; 96tpi diskettes

00004F55 0002                   bpb96t:     dw      512
00004F57 01                         db      1
00004F58 0100                       dw      1
00004F5A 02                         db      2
00004F5B E000                       dw      224
00004F5D 6009                       dw      2*15*80 ; 2400
00004F5F F9                         db      0F9h
00004F60 0700                       dw      7
00004F62 0F00                       dw      15
00004F64 0200                       dw      2
00004F66 00000000                   dd      0
00004F6A 00000000                       dd      0
                                    ; 27/12/2023
00004F6E 00<rep 1Ch>                times   28 db 0             ; FAT32 extensions (to BDS)
00004F8A 90                         db      90h

                                ; 3 1/2 inch diskette bpb

00004F8B 0002                   bpb35:      dw      512
00004F8D 02                         db      2
00004F8E 0100                       dw      1
00004F90 02                         db      2
00004F91 7000                       dw      112
00004F93 A005                       dw      2*9*80 ; 1440
00004F95 F9                         db      0F9h
00004F96 0300                       dw      3
00004F98 0900                       dw      9
00004F9A 0200                       dw      2
00004F9C 00000000                   dd      0
00004FA0 00000000                       dd      0
                                    ; 27/12/2023
00004FA4 00<rep 1Ch>                times   28 db 0             ; FAT32 extensions (to BDS)
00004FC0 90                         db      90h

00004FC1 0002                   bpb35h:     dw      512
00004FC3 01                         db      1
00004FC4 0100                       dw      1
00004FC6 02                         db      2
00004FC7 E000                       dw      224
00004FC9 400B                       dw      2*18*80 ; 2880
00004FCB F0                         db      0F0h
00004FCC 0900                       dw      9
00004FCE 1200                       dw      18
00004FD0 0200                       dw      2
00004FD2 00000000                   dd      0
00004FD6 00000000                       dd      0
                                    ; 27/12/2023
00004FDA 00<rep 1Ch>                times   28 db 0             ; FAT32 extensions (to BDS)
00004FF6 90                         db      90h

                                ; m037 - BEGIN

00004FF7 0002                   bpb288:     dw      512
00004FF9 02                         db      2
00004FFA 0100                       dw      1
00004FFC 02                         db      2
00004FFD F000                       dw      240
00004FFF 8016                       dw      2*36*80 ; 5760
00005001 F0                         db      0F0h
00005002 0900                       dw      9
00005004 2400                       dw      36
00005006 0200                       dw      2
00005008 00000000                   dd      0
0000500C 00000000                       dd      0
                                    ; 27/12/2023
00005010 00<rep 1Ch>                times   28 db 0             ; FAT32 extensions (to BDS)
0000502C 90                         db      90h

                                ; m037 - END

                                ; 12/05/2019

0000502D 90                     align 2

                                ; 02/11/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 IO.SYS)
                                ; MSDOS 5.0 IO.SYS - SYSINIT:3B26h

                                ; 13/04/2024 - Retro DOS v5.0
                                ; (PCDOS 7.1 IBMBIO.COM - SYSINIT:5738h)

0000502E [1F4F]                 bpbtable:   dw          bpb48t              ; 48tpi drives
00005030 [554F]                         dw      bpb96t              ; 96tpi drives
00005032 [8B4F]                         dw      bpb35               ; 3.5" drives
                                ; the following are not supported, so default to 3.5" media layout
00005034 [8B4F]                         dw      bpb35               ; not used - 8" drives
00005036 [8B4F]                         dw      bpb35               ; not used - 8" drives
00005038 [8B4F]                         dw      bpb35               ; not used - hard files
0000503A [8B4F]                         dw      bpb35               ; not used - tape drives
0000503C [C14F]                         dw      bpb35h              ; 3-1/2" 1.44mb drive
0000503E [8B4F]                         dw      bpb35               ; ERIMO                         m037
00005040 [F74F]                         dw      bpb288              ; 2.88 MB diskette drives       m037

                                switchlist:
00005042 08464853544449434E         db      8,"FHSTDICN"        ; preserve the positions of n and c.

                                ;--------------------------------------------------------------------------
                                ; Messages
                                ;--------------------------------------------------------------------------
```

```
                                ; 19/04/2019 - Retro DOS v4.0
                                ; MSDOS 6.21 IO.SYS - SYSINIT:54D1h
0000504B 00                         db      0
                                ; 02/11/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 IO.SYS)
                                ; MSDOS 5.0 IO.SYS - SYSINIT:3B44h
                                ; 13/04/2024
                                ; MSDOS 6.22 IO.SYS - SYSINIT:559Eh
                                ; 13/04/2024 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMBIO.COM)
                                ; PCDOS 7.1 IBMBIO.COM - SYSINIT:5756h
                                badopm:
0000504C 0D0A                       db      0Dh,0Ah
0000504E 556E7265636F676E69-        db      'Unrecognized command in CONFIG.SYS'
00005057 7A656420636F6D6D61-
00005060 6E6420696E20434F4E-
00005069 4649472E535953
                                crlfm:
00005070 0D0A24                     db      0Dh,0Ah,'$'
                                badparm:
00005073 0D0A                       db      0Dh,0Ah
00005075 42616420636F6D6D61-        db      'Bad command or parameters - $'
0000507E 6E64206F7220706172-
00005087 616D6574657273202D-
00005090 2024
                                badsiz_pre:
00005092 0D0A                       db      0Dh,0Ah
00005094 536563746F72207369-        db      'Sector size too large in file $'
0000509D 7A6520746F6F206C61-
000050A6 72676520696E206669-
000050AF 6C652024
                                badld_pre:
000050B3 0D0A                       db      0Dh,0Ah
000050B5 426164206F72206D69-        db      'Bad or missing $'
000050BE 7373696E672024
                                badcom:
000050C5 436F6D6D616E642049-        db      'Command Interpreter',0
000050CE 6E7465727072657465-
000050D7 7200
                                badcountry:
000050D9 0D0A                       db      0Dh,0Ah
000050DB 496E76616C69642063-        db      'Invalid country code or code page',0Dh,0Ah,'$'
000050E4 6F756E74727920636F-
000050ED 6465206F7220636F64-
000050F6 6520706167650D0A24
                                badcountrycom:
000050FF 0D0A                       db      0Dh,0Ah
00005101 4572726F7220696E20-        db      'Error in COUNTRY command',0Dh,0Ah,'$'
0000510A 434F554E5452592063-
00005113 6F6D6D616E640D0A24
                                insufmemory:
0000511C 0D0A                       db      0Dh,0Ah
0000511E 496E73756666696369-        db      'Insufficient memory for COUNTRY.SYS file',0Dh,0Ah,'$'
00005127 656E74206D656D6F72-
00005130 7920666F7220434F55-
00005139 4E5452592E53595320-
00005142 66696C650D0A24
                                badmem:
00005149 0D0A                       db      0Dh,0Ah
0000514B 436F6E666967757261-        db      'Configuration too large for memory',0Dh,0Ah,'$'
00005154 74696F6E20746F6F20-
0000515D 6C617267652066F72-
00005166 206D656D6F72790D0A-
0000516F 24
                                badblock:
00005170 0D0A                       db      0Dh,0Ah
00005172 546F6F206D616E7920-        db      'Too many block devices',0Dh,0Ah,'$'
0000517B 626C6F636B20646576-
00005184 696365730D0A24
                                badstack:
0000518B 0D0A                       db      0Dh,0Ah
0000518D 496E76616C69642053-        db      'Invalid STACK parameters',0Dh,0Ah,'$'
00005196 5441434B2070617261-
0000519F 6D65746572730D0A24
                                    ; 18/12/2022
                                ;badorder:
                                    ;db     0Dh,0Ah
                                    ;db     'Incorrect order in CONFIG.SYS line $'
                                errorcmd:
000051A8 4572726F7220696E20-        db      'Error in CONFIG.SYS line $'
000051B1 434F4E4649472E5359-
000051BA 53206C696E652024
                                ; 01/01/2023 - Retro DOS v4.2 (Modified MSDOS 6.21 IO.SYS)
                                ; (SYSINIT:566Eh)
                                ; 02/11/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
                                ;%if 0
000051C2 4F4E                   OnOff:      db      'ON'
000051C4 4F4646                 OnOff2:     db      'OFF'
                                    ; 04/01/2023 - Retro DOS v4.2 (Modified MSDOS 6.21 IO.SYS)
                                    ; (SYSINIT:5673h)
                                ;StartMsg:
                                ;   db      'Starting MS-DOS...',0Dh,0Ah
                                ;   db      0Ah,0
                                    ; 17/12/2023 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMBIO.COM)
                                    ; (SYSINIT:58F7h)
                                StartMsg:
000051C7 5374617274696E6720-        db      'Starting PC DOS...',0Dh,0Ah
000051D0 504320444F532E2E2E-
000051D9 0D0A
000051DB 0A00                       db      0Ah,0
                                _$PauseMsg:
                                    ; 17/12/2023
                                    ;db     'Press any key to continue . . .',0Dh,0Ah,'$'
                                    ; 04/08/2023 (PCDOS 7.10 - IBMBIO.COM SYSINIT:590Dh)
000051DD 507265737320616E79-        db      'Press any key to continue...',0Dh,0Ah,'$'
000051E6 206B657920746F2063-
000051EF 6F6E74696E75652E2E-
000051F8 2E0D0A24
                                _$CleanMsg:
                                    ;db     'MS-DOS is bypassing your CONFIG.SYS and AUTOEXEC.BAT files.',0Dh,0Ah,'$'
                                    ; 17/12/2023
000051FC 504320444F53206973-        db      'PC DOS is bypassing your CONFIG.SYS and AUTOEXEC.BAT files.',0Dh,0Ah,'$'
00005205 20627970617373696E-
```

```
44923 0000520E 6720796F757220434F-
44923 00005217 4E4649472E53595320-
44923 00005220 616E64204155544F45-
44923 00005229 5845432E4241542066-
44923 00005232 696C65732E0D0A24
44924                           _$InterMsg:
44925                               ;db     'MS-DOS will prompt you to confirm each CONFIG.SYS command.',0Dh,0Ah,'$'
44926                               ; 17/12/2023
44927 0000523A 504320444F53207769-      db      'PC DOS will prompt you to confirm each CONFIG.SYS command.',0Dh,0Ah,'$'
44927 00005243 6C6C2070726F6D7074-
44927 0000524C 20796F7520746F2063-
44927 00005255 6F6E6669726D206561-
44927 0000525E 636820434F4E464947-
44927 00005267 2E53595320636F6D6D-
44927 00005270 616E642E0D0A24
44928                           _$MenuHeader:
44929 00005277 0D0A                 db      0Dh,0Ah
44930                               ; 17/12/2023
44931                               ;db     '  MS-DOS 6.2 Startup Menu',0Dh,0Ah
44932                               ;db     '  '
44933                               ;times  23 db (0CDh)  ; ALT 205 ; '=======================' ; 06/08/2023
44934                               ;db     0Dh,0Ah,'$'
44935                               ; 04/08/2023 (PCDOS 7.10 - IBMBIO.COM SYSINIT:59A7h)
44936 00005279 2020504320444F5320-      db      '  PC DOS 7.1 Startup Menu',0Dh,0Ah
44936 00005282 372E31205374617274-
44936 0000528B 7570204D656E750D0A
44937 00005294 2020                 db      '  '
44938 00005296 CD<rep 17h>          times   23 db (0CDh)  ; ALT 205 ; '=======================' ; 06/08/2023
44939 000052AD 0D0A24               db      0Dh,0Ah,'$'
44940                           _$MenuPrmpt:
44941 000052B0 2020456E7465722061-      db      '  Enter a choice: $'
44941 000052B9 2063686F6963653A20-
44941 000052C2 24
44942                           _$StatusLine:
44943 000052C3 46353D427970617373-      db      'F5=Bypass startup files F8=Confirm each line of CONFIG.SYS '
44943 000052CC 207374617274757020-
44943 000052D5 66696C65732046383D-
44943 000052DE 436F6E6669726D2065-
44943 000052E7 616368206C696E6520-
44943 000052F0 6F6620434F4E464947-
44943 000052F9 2E53595320
44944 000052FE 616E64204155544F45-      db      'and AUTOEXEC.BAT [ ]$'
44944 00005307 5845432E424154205B-
44944 00005310 205D24
44945                           _$InterPrmpt:
44946                               ;db     ' [Y,N]?$'
44947                               ; 13/04/2024
44948                               ; 04/08/2023
44949 00005313 205B592C4E2C455343-      db      ' [Y,N,ESC]?$' ; PCDOS 7.1 - IBMBIO.COM
44949 0000531C 5D3F24
44950 0000531F 59455324             _$YES:      db      'YES$'
44951 00005323 4E4F2024             _$NO:       db      'NO $'
44952                           _$TimeOut:
44953 00005327 54696D652072656D61-      db      'Time remaining: $'
44953 00005330 696E696E673A2024
44954                           badcomprmpt:
44955                               ;db     'Enter correct name of Command Interpreter (eg, C:\COMMAND.COM)'
44956                               ; 13/04/2024 (PCDOS 7.1 IBMBIO.COM)
44957 00005338 456E74657220636F72-      db      'Enter correct name of Command Interpreter (for example, C:\COMMAND.COM)'
44957 00005341 72656374206E616D65-
44957 0000534A 206F6620436F6D6D61-
44957 00005353 6E6420496E74657270-
44957 0000535C 726574657220028666F-
44957 00005365 72206578616D706C65-
44957 0000536E 2C20433A5C434F4D4D-
44957 00005377 414E442E434F4D29
44958 0000537F 0D0A24               db      0Dh,0Ah,'$'
44959                           _$AutoPrmpt:
44960 00005382 50726F636573732041-      db      'Process AUTOEXEC.BAT [Y,N]?$'
44960 0000538B 55544F455845432E42-
44960 00005394 4154205B592C4E5D3F-
44960 0000539D 24
44961
44962                           ;%endif ; 02/11/2022
44963
44964                           ; 01/01/2023 - Retro DOS v4.2 (Modified MSDOS 6.21 IO.SYS)
44965                           ; (SYSINIT:5840h)
44966
44967                           ; 02/11/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 IO.SYS)
44968                           ; MSDOS 5.0 IO.SYS - SYSINIT:3CE0h
44969
44970                           TooManyDrivesMsg:
44971 0000539E 5741524E494E472120-      db      'WARNING! Logical drives past Z: exist and will be ignored',0Dh,0Ah,'$'
44971 000053A7 4C6F676963616C2064-
44971 000053B0 726976657320706173-
44971 000053B9 74205A3A2065786973-
44971 000053C2 7420616E642077696C-
44971 000053CB 6C2062652069676E6F-
44971 000053D4 7265640D0A24
44972
44973                           ; MSDOS 6.21 IO.SYS - SYSINIT:587Ch
44974                               ;db     'Wrong DBLSPACE.BIN version',0Dh,0Ah,'$'
44975                               ;db     7 dup(0)
44976
44977                               ;times  7 db 0
44978                               ; 02/11/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
44979                           ; MSDOS 5.0 IO.SYS - SYSINIT:3D1Ch
44980                               ; 09/12/2022
44981                               ;times 4 db 0
44982
44983                               ; 08/04/2024 - Retro DOS v5.0
44984                           ; PCDOS 7.1 IBMBIO.COM - SYSINIT:5B0Bh
44985                           baddblspace:
44986 000053DA 526571756972656420-      db      'Required system component is not installed',0Dh,0Ah,'$',0
44986 000053E3 73797374656D20636F-
44986 000053EC 6D706F6E656E742069-
44986 000053F5 73206E6F7420696E73-
44986 000053FE 74616C6C65640D0A24-
44986 00005407 00
44987                               ;db     7 dup(0)
44988
44989                           ;--------------------------------------------------------------------------
44990                                       ; 09/12/2022
44991                                       ;db 0
44992
44993                           number3div  equ ($-SYSINIT$)
44994                           number3mod  equ (number3div % 16)
44995
44996                           %if (number3mod>0) & (number3mod<16) ; 17/09/2023
44997 00005408 00<rep 8h>                   times (16-number3mod) db 0
44998                           %endif
44999
45000                           ;--------------------------------------------------------------------------
45001                           ; 09/12/2022 - MSDOS 5.0 IO.SYS:3D20h ;;; SI_end = 3D20h for MSDOS 5.0 IO.SYS
```

```
45002                                   ;--------------------------------------------------------------------------
45003
45004                                   ;MSDOS 6.21 IO.SYS - SYSINIT:5899h
45005
45006                                   ;--------------------------------------------------------------------------
45007                                   ; 20/04/2019 - Retro DOS v4.0
45008
45009                                   ; 09/12/2022
45010                                   ;
45011                                   ;bss_start:
45012                                   ;
45013                                   ;ABSOLUTE bss_start
45014                                   ;
45015                                   ;alignb 16
45016
45017                                   SI_end:  ; SI_end equ $
45018
45019                                   ;--------------------------------------------------------------------------
45020
45021                                   ;sysinitseg ends
45022
45023                                   ; ****************************************************************************
45024
45025                                   ; 04/01/2023 - MSDOS 6.21 SYSINIT:SI_end = SYSINIT:58A0h (IOSYS:9F46h)
45026                                   ; 09/12/2022 - MSDOS 5.0 SYSINIT:SI_end = SYSINIT:3D20h
45027
45028                                   SYSINITSIZE equ SI_end - SYSINIT$
45029                                   DOSLOADSEG  equ SYSINITSEG+((SYSINITSIZE+15)/16)
45030
45031                                   ;--------------------------------------------------------------------------
45032                                   ; End of Retro DOS v5.0 IBMBIO.COM (IO.SYS) source by Erdogan Tan (2023)
45033                                   ;--------------------------------------------------------------------------
45034
45035                                   ; 21/12/2022 - Retro DOS v4.0 (Modified MSDOS 5.0)
45036                                   ; 02/10/2023 - Retro DOS v5.0 (Modified PCDOS 7.1)
45037                                   ;--------------------------------------------------------------------------
45038                                   ;--------------------------------------------------------------------------
45039
45040                                   ; ------------------------------------------------------------------------------
45041                                   ; START OF PCDOS 7.1 -IBMDOS.COM- KERNEL CODE (MSDOS.SYS) -will be relocated-
45042                                   ; ------------------------------------------------------------------------------
45043                                   ; 02/10/2023 - Retro DOS v5.0
45044                                   ; 04/01/2023 - Retro DOS v4.2
45045                                   ; 29/12/2022 - Retro DOS v4.1
45046                                   ; 18/03/2019 - Retro DOS v4.0
45047                                   ; 11/06/2018 - Retro DOS v3.0
45048
45049                                   ;MSDOS_BIN_OFFSET:
45050                                   IBMDOS_BIN_OFFSET: ; this offset must be paragraph aligned
45051                                               ;; 28/06/2019 ('msdos6.s')
45052                                               ;incbin 'MSDOS6.BIN' ; Retro DOS 4.0 - MSDOS 6.21 KERNEL
45053
45054                                               ; 29/12/2022
45055                                               ;incbin 'MSDOS51.BIN' ; Retro DOS 4.1 - MSDOS 5.0+ KERNEL
45056
45057                                               ; 29/09/2023 (PARASTART=3DE0h)
45058                                               ; 27/09/2023 (BugFix) ((PARASTART=3DD0h))
45059                                               ; 04/01/2023
45060                                               ;incbin 'MSDOS6.BIN' ; Retro DOS 4.2 - MSDOS 6.21+ KERNEL
45061
45062                                               ; 06/08/2025
45063                                               ; 10/07/2024
45064                                               ; 07/07/2024
45065                                               ; 08/05/2024
45066                                               ; 14/04/2024
45067                                               ; 02/10/2023 - Retro DOS v5.0 - PCDOS 7.1 KERNEL
45068 00005410 <bin A326h>                          incbin  'IBMDOS7.BIN'
45069
45070                                               ;; 28/12/2022 (BugFix)
45071                                               ;; 22/12/2022
45072                                               ;; 21/12/2022 ('msdos5.s')
45073                                               ;incbin 'MSDOS5.BIN'  ; Retro DOS 4.0 - MSDOS 5.0+ KERNEL
45074
45075                                   ; 28/09/2023
45076                                   ;msdos_bin_size equ $ - MSDOS_BIN_OFFSET
45077
45078                                   align 2
45079
45080                                   ; 21/12/2022
45081                                   ;;END_OF_KERNEL:
45082                                   ;END_OF_KERNEL equ $
45083
45084                                   ; 28/09/2023
45085                                   S3SIZE equ $-$$
45086                                   KERNEL_SIZE equ S1SIZE+S2SIZE+S3SIZE
45087
45088                                   ;============================================================================
45089                                   ;   END
45090                                   ;============================================================================
45091                                   ; Retro DOS v5.0 by Erdogan Tan (Redevelopment of PC-DOS 7.1 KERNEL via NASM)
45092                                   ; ----------------------------
45093                                   ; APRIL 2024, ISTANBUL - TURKIYE.
```