```
                        ; ****************************************************************************
                        ; RETRODOS.SYS (PCDOS 7.1 Kernel) - RETRO DOS v5.0 by ERDOGAN TAN - 12/09/2023
                        ; ----------------------------------------------------------------------------
                        ; Last Update: 19/01/2026 - 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 E9951B                 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 E8BB00                         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 E98500                         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 E83601                         call    bcd_verify     ; verify bcd values in range
0000189F 726C                           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 E8DD00                         call    date_verify
000018A8 7263                           jb      short r_d_ret
                                        ;mov    word [cs:daycnt2], 0
                                        ; 08/08/2023
000018AA C606[0006]00                   mov     byte [daycnt2], 0
000018AF E8A300                         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
                                        ; 19/01/2026 (BugFix)
000018FF 740C                           jz      short r_d_ret

                                        ; 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
00001901 B400                           mov     ah, 0
00001903 BE[8F04]                       mov     si, month_tab
                                r_d_sum_loop:
00001906 AC                             lodsb
00001907 0106[0006]                     add     [daycnt2], ax
0000190B E2F9                           loop    r_d_sum_loop
                                r_d_ret:
                                        ;mov    si, [cs:daycnt2]
                                        ; 08/08/2023
0000190D 8B36[0006]                     mov     si, [daycnt2]
00001911 5A                             pop     dx
00001912 59                             pop     cx
00001913 5B                             pop     bx
00001914 58                             pop     ax
00001915 C3                             retn

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

                                r_t_retj:
00001916 31C9                           xor     cx, cx
00001918 31D2                           xor     dx, dx
0000191A 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
0000191C B402                           mov     ah, 2
0000191E 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
00001920 72F4                           jb      short r_t_retj
00001922 882E[FC05]                     mov     [bin_date_time], ch ; hours
00001926 880E[FD05]                     mov     [bin_date_time+1], cl ; minutes
0000192A 8836[FE05]                     mov     [bin_date_time+2], dh ; seconds
0000192E C606[FF05]00                   mov     byte [bin_date_time+3], 0 ; unused for time
00001933 E89F00                         call    bcd_verify
00001936 72DE                           jb      short r_t_retj
00001938 E88500                         call    time_verify
0000193B 72D9                           jb      short r_t_retj
0000193D E81500                         call    in_bin          ; from bcd to bin
00001940 8A2E[FC05]                     mov     ch, [bin_date_time]
00001944 8A0E[FD05]                     mov     cl, [bin_date_time+1]
00001948 8A36[FE05]                     mov     dh, [bin_date_time+2]
0000194C 8A16[FF05]                     mov     dl, [bin_date_time+3]
                                        ; 16/10/2022
                                        ; 17/09/2022
                                        ; 31/05/2019
00001950 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:
00001954 C3                             retn

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

                                ;   in_bin converts bin_date_time values from bcd to bin

                                in_bin:         ; proc near
00001955 A0[FC05]                       mov     al, [bin_date_time] ; century or hours
00001958 E81F00                         call    bcd_to_bin
0000195B A2[FC05]                       mov     [bin_date_time], al
0000195E A0[FD05]                       mov     al, [bin_date_time+1] ; years or minutes
00001961 E81600                         call    bcd_to_bin
00001964 A2[FD05]                       mov     [bin_date_time+1], al
00001967 A0[FE05]                       mov     al, [bin_date_time+2] ; months or seconds
0000196A E80D00                         call    bcd_to_bin
0000196D A2[FE05]                       mov     [bin_date_time+2], al
00001970 A0[FF05]                       mov     al, [bin_date_time+3] ; days (not used for time)
00001973 E80400                         call    bcd_to_bin
00001976 A2[FF05]                       mov     [bin_date_time+3], al
00001979 C3                             retn

                                ; =============== S U B   R O U T I 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
0000197A 88C4                           mov     ah, al
0000197C 240F                           and     al, 0Fh
0000197E B104                           mov     cl, 4
00001980 D2EC                           shr     ah, cl
00001982 D50A                           aad
00001984 C3                             retn

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

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

                                date_verify:        ; proc near
00001985 803E[FC05]20                           cmp     byte [bin_date_time], 20h ; century check
0000198A 7732                                   ja      short date_error
0000198C 740E                                   jz      short century_20 ; jmp in 21th century
0000198E 803E[FC05]19                           cmp     byte [bin_date_time], 19h ; century check
                                                ;jb     short date_error
                                                ; 12/12/2022
00001993 722A                                   jb      short date_err2
00001995 803E[FD05]80                           cmp     byte [bin_date_time+1], 80h ; year check
                                                ;jb     short date_error
                                                ; 12/12/2022
0000199A 7223                                   jb      short date_err2
                                century_20:
0000199C 803E[FD05]99                           cmp     byte [bin_date_time+1], 99h ; year check
000019A1 771B                                   ja      short date_error
000019A3 803E[FE05]12                           cmp     byte [bin_date_time+2], 12h ; month check
000019A8 7714                                   ja      short date_error
000019AA 803E[FE05]00                           cmp     byte [bin_date_time+2], 0
                                                ;jbe    short date_error
000019AF 760D                                   jna     short date_error
000019B1 803E[FF05]31                           cmp     byte [bin_date_time+3], 31h ; day check
000019B6 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
000019B8 803E[FF05]01                           cmp     byte [bin_date_time+3], 1 ; day check
000019BD C3                                     retn
                                ;---------------------------------------------------------------------------

                                date_error:
000019BE F9                             stc
                                date_err2:
000019BF C3                             retn

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

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

                                time_verify:        ; proc near
000019C0 803E[FC05]24                           cmp     byte [bin_date_time], 24h ; hour check
000019C5 770C                                   ja      short time_error
000019C7 803E[FD05]59                           cmp     byte [bin_date_time+1], 59h ; minute check
000019CC 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
000019CE 803E[FE05]5A                           cmp     byte  [bin_date_time+2], 5Ah
                                time_error:
                                bv_error:
000019D3 F5                                     cmc     ; cf=0 -> cf=1, cf=1 -> cf=0
000019D4 C3                                     retn

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

                                ;time_error:
                                                ;stc
                                                ;retn

                                ; =============== S U B   R O U T I 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
000019D5 B90400                                 mov     cx, 4           ; 4 bytes to check
000019D8 BB[FC05]                               mov     bx, bin_date_time
                                bv_loop:
000019DB 8A07                                   mov     al, [bx]        ; get a bcd number (0..99)
000019DD 88C4                                   mov     ah, al
000019DF 250FF0                                 and     ax, 0F00Fh      ; 10's place in high ah, 1's in al
                                                                        ; is 1's place in range?
000019E2 3C0A                                   cmp     al, 10
```

```
000019E4 77ED                           ja      short bv_error ; jmp out of range
000019E6 D0EC                           shr     ah, 1
000019E8 D0EC                           shr     ah, 1
000019EA D0EC                           shr     ah, 1
000019EC D0EC                           shr     ah, 1
000019EE 80E40F                         and     ah, 0Fh         ; get rid of any erroneous bits
000019F1 80FC0A                         cmp     ah, 10          ; is 10's place in range
000019F4 77DD                           ja      short bv_error ; jmp out of range
000019F6 43                             inc     bx              ; next byte
000019F7 49                             dec     cx
000019F8 75E1                           jnz     short bv_loop
000019FA F8                             clc                     ; set success flag
000019FB 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)

000019FC 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:
000019FE 0000                           dw 0
00001A00 0000                           dw 0
00001A02 0000                   doscnt:         dw 0                    ; how many sectors to read
                                ;fbigfat:   db 0                ; flags for drive
00001A04 0000                   fatloc:         dw 0                    ; seg addr of fat sector
00001A06 0000                   init_bootseg:   dw 0                    ; seg addr of buffer for reading boot record
                                ; 09/12/2023
00001A08 00                     fbigfat:    db 0                ; flags for drive
00001A09 80                     rom_drv_num:    db 80h                  ; rom drive number
00001A0A 0002                   md_sectorsize:  dw 200h                 ; used by get_fat_sector proc.
                                ; 12/12/2023
                                ;temp_cluster:  dw 0                    ; used by get_fat_sector proc.
00001A0C 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.

00001A0E 02                     num_heads:  db 2    ; dw 2          ; number of heads returned by rom
00001A0F 00                             db 0    ; 09/12/2023
                                ;sec_trk:   db 9                    ; sec/trk returned by rom
00001A10 28                     num_cyln:   db 40   ; dw 40         ; number of cylinders returned by rom
00001A11 00                             db 0    ; 09/12/2023
                                ; 09/12/2023
00001A12 09                     sec_trk:    db 9                    ; sec/trk returned by rom
00001A13 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:
00001A14 0000                           dw 0
00001A16 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
```

```
4361 00001A18 0000A87F0308000200-     disktable2: dw 0, 32680, 0803h, 512, 0 ; for compatibility.
4361 00001A21 00
4362                                                                        ; (32680 sectors, 16340 KB)
4363 00001A22 040000000204000240-                 dw 4, 0, 0402h, 512, 40h   ; covers upto 134 mb media.
4363 00001A2B 00
4364                                                                        ; fbig = 40h  ; (40000h sectors = 128 MB)
4365 00001A2C 080000000308000240-                 dw 8, 0, 0803h, 512, 40h   ; upto 268 mb ; (80000h sectors = 256 MB)
4365 00001A35 00
4366 00001A36 100000000410000240-                 dw 16, 0, 1004h, 512, 40h  ; upto 536 mb ; (100000h sectors = 512 MB)
4366 00001A3F 00
4367 00001A40 200000000520000240-                 dw 32, 0, 2005h, 512, 40h  ; upto 1072 mb ; (200000h sectors = 1024 MB)
4367 00001A49 00
4368 00001A4A 400000000640000240-                 dw 64, 0, 4006h, 512, 40h  ; upto 2144 mb ; (400000h sectors = 2048 MB)
4368 00001A53 00
4369                                               ; 09/12/2023
4370                                               ;dw 128, 0, 8007h, 512, 40h ; upto 4288 mb ; (800000h sectors = 4096 MB)
4371 00001A54 FFFFFFFF0308000060-                 dw 0FFFFh, 0FFFFh, 0803h, 0, 60h ; FAT32 (> 2144MB)
4371 00001A5D 00
4372                                                                        ; (fbig and fbigbig flags are set)
4373
4374                                   ; ----------------------------------------------------------------------------
4375
4376                                   ;******************************************************
4377                                   ;variables for mini disk initialization
4378                                   ;******************************************************
4379
4380                                   ; 01/10/2022
4381                                   ; [ Note: Minidisk == logical dos drive (in extended dos partition) ]
4382
4383 00001A5E 00                       rom_minidisk_num: db 0                  ; temp variable       for phys unit
4384 00001A5F 00                       hnum:             db 0                  ; real number of hardfiles
4385 00001A60 [3C05]                   last_dskdrv_table: dw dskdrvs           ; index into dskdrv table
4386 00001A62 [4C08]                   end_of_bdss:      dw bdss               ; offset value of the ending address
4387                                                                     ; of bds table. needed to figure out
4388                                                                     ; the dosdatasg address.
4389 00001A64 0000                     mini_hdlim: dw 0
4390 00001A66 0000                     mini_seclim:      dw 0
4391
4392                                   ; 19/12/2023
4393                                   ; 09/12/2023
4394                                   ; (PCDOS 7.1 IBMBIO.COM, BIOSDATA:1A7Ah)
4395                                   ;ld_p_number:     dw 2BADh              ; (for 'find_mini_partition' proc)
4396
4397                                   ;end of mini disk init variables **********************
4398
4399                                   ; ----------------------------------------------------------------------------
4400
4401 00001A68 30312F31302F383400       bios_date:  db '01/10/84',0       ; used for checking at rom bios      date.
4402
4403                                   ; 13/12/2022
4404                                   %if 0
4405
4406                                   ;align 2
4407                                              db  90h
4408
4409                                   ; the following are the recommended bpbs for the media that we know of so far.
4410
4411                                   ;struc bpbx
4412                                   ;   resw 1 ; 512
4413                                   ;   resb 1
4414                                   ;   resw 1 ; 1
4415                                   ;   resb 1 ; 2
4416                                   ;   resw 1
4417                                   ;   resw 1
4418                                   ;   resb 1
4419                                   ;   resw 1
4420                                   ;   resw 1
4421                                   ;   resw 1 ; 2
4422                                   ;   resw 1
4423                                   ;   resw 1 ; hidden sector high
4424                                   ;   resd 1 ; extended total sectors
4425                                   ;.size:
4426                                   ;endstruc
4427
4428                                   ; 08/01/2019 - Retro DOS v4.0
4429
4430                                   ; 20/04/2019
4431
4432                                   ; 01/10/2022 - Retro DOS v4.0 (MSDOS 5.0) IO.SYS
4433
4434                                   ; 09/12/2023 - Retro DOS v5.0 (PCDOS 7.1) IBMBIO.COM
4435                                   ; (PCDOS 7.1 IBMBIO.COM, BIOSDATA:1A86h)
4436
4437                                   ; 09/12/2022
4438                                   BPB48T:
4439                                   ;bpb48t:    ; bpbx <512, 2, 1, 2, 112, 720, 0FDh, 2, 9, 2, 0, 0, 0, 0>
4440                                               ; 48 tpi diskettes    ;
4441                                               dw     512            ; physical sector size in bytes
4442                                               db     2              ; sectors/allocation unit
4443                                               dw     1              ; reserved sectors for dos
4444                                               db     2              ; number of allocation tables
4445                                               dw     112            ; number of directory entries
4446                                               dw     720 ; 2*9*40   ; number of sectors (at 512 bytes each)
4447                                               db     0FDh           ; media descriptor
4448                                               dw     2              ; number of fat sectors
4449                                               dw     9              ; sectors per track
4450                                               dw     2              ; heads
4451                                               dw     0              ; hidden sector count (low word)
4452                                               dw     0              ; hidden sector (high)
4453                                               dw     0              ; number of sectors (low)
4454                                               dw     0              ; number of sectors (high)
4455                                               ; 09/12/2023
4456                                               ; FAT32 extensions (to BDS)
4457                                               times  28 db 0
4458                                               ;
4459                                               db 90h
4460                                   ;align 2
4461                                   BPB96T:
4462                                   ;bpb96t:    ; bpbx <512, 1, 1, 2, 224, 2400, 0F9h, 7, 15, 2, 0, 0, 0, 0>
4463                                               ; 96 tpi diskettes    ;
4464                                               dw     512            ; physical sector size in bytes
4465                                               db     1              ; sectors/allocation unit
4466                                               dw     1              ; reserved sectors for dos
4467                                               db     2              ; number of allocation tables
4468                                               dw     224            ; number of directory entries
4469                                               dw     2400 ; 2*15*80 ; number of sectors (at 512 bytes each)
4470                                               db     0F9h           ; media descriptor
4471                                               dw     7              ; number of fat sectors
4472                                               dw     15             ; sectors per track
4473                                               dw     2              ; heads
4474                                               dw     0              ; hidden sector count (low word)
4475                                               dw     0              ; hidden sector (high)
4476                                               dw     0              ; number of sectors (low)
4477                                               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
```

```
4602                                ;     then transfer control to Bios_Code:bp using a couple of pushes
4603                                ;     and a retf
4604                                
4605                                ; 16/10/2022
4606                                ;BC_RETF equ bc_retf - DOSBIOSEG_2C7h
4607                                ; 09/12/2022
4608                                BC_RETF equ bc_retf
4609                                
4610                                ; 09/12/2023
4611                                ;PCDOS 7.1 IBMBIO.COM bc_retf offset = 0CAh (in BIOSCODE segment = 364h)
4612                                
4613                                addr_of_bcretf:    ;dw 0C8h              ; dw bc_retf
4614                                                                       ; 2C7h:0C8h = 70h:2638h
4615                                                                       ; 09/12/2023
4616                                                                       ; 364h:0CAh = 70h:300Ah ; PCDOS 7.1
4617 00001A71 [CA00]                              dw BC_RETF              ; dw 0CAh
4618                                
4619                                ; ---------------------------------------------------------------------------
4620                                
4621                                call_bios_code:    ; proc far
4622 00001A73 2EFF36[711A]                         push   word [cs:addr_of_bcretf]
4623                                                                       ; set up near return to far return
4624 00001A78 2EFF36[0406]                         push   word [cs:cdev+2] ; push Bios_Code segment
4625 00001A7D 55                                   push   bp               ; save offset of utility function
4626 00001A7E CB                                   retf                    ; far jump to (DOS)BIOS code
4627                                
4628                                ; ---------------------------------------------------------------------------
4629                                
4630                                                   ; 09/12/2023 - Retro DOS v5.0 IBMBIO.COM/IO.SYS
4631                                                   ; 20/12/2022
4632 00001A7F 00                   flp_drvs:   db 0
4633                                                   ; 11/12/2023
4634                                                   ; (PCDOS 7.1 IBMBIO.COM - BIOSDATA:1B81h)
4635                                firstcluster_hw:
4636 00001A80 0000                                     dw 0    ; 06/04/2024
4637 00001A82 00                   Boot_Drv:   db 0
4638                                
4639                                ; ---------------------------------------------------------------------------
4640                                
4641                                ; 09/12/2023 - Retro DOS v5.0 (PCDOS 7.1) IBMBIO.COM
4642                                ;---------------------------------------------------------------------------
4643                                ; PCDOS 7.1 CD BOOT option code
4644                                ;---------------------------------------------------------------------------
4645                                ; (PCDOS 7.1 IBMBIO.COM, BIOSDATA:1B84h)
4646                                
4647                                cd_boot_option:
4648 00001A83 50                                       push   ax
4649 00001A84 1E                                       push   ds
4650 00001A85 06                                       push   es
4651 00001A86 52                                       push   dx
4652                                cdbo_1:
4653 00001A87 B401                                     mov    ah, 1
4654 00001A89 CD16                                     int    16h                     ; KEYBOARD - status
4655 00001A8B 7406                                     jz     short cdbo_2
4656 00001A8D 30E4                                     xor    ah, ah
4657 00001A8F CD16                                     int    16h                     ; KEYBOARD - READ CHAR FROM BUFFER, WAIT IF EMPTY
4658                                                                                  ; Return: AH = scan code, AL = character
4659 00001A91 EBF4                                     jmp    short cdbo_1
4660                                cdbo_2:
4661 00001A93 0E                                       push   cs
4662 00001A94 1F                                       pop    ds
4663 00001A95 BE[6D1B]                                 mov    si, cd_boot_msg              ; "Press the ENTER key to boot from CD"...
4664 00001A98 AC                                       lodsb
4665                                cdbo_3:
4666 00001A99 BB0700                                   mov    bx, 7
4667 00001A9C B40E                                     mov    ah, 0Eh
4668 00001A9E CD10                                     int    10h                     ; - VIDEO - WRITE CHARACTER AND ADVANCE CURSOR (TTY WRITE)
4669                                                                                  ; AL = character, BH = display page (alpha modes)
4670                                                                                  ; BL = foreground color (graphics modes)
4671 00001AA0 AC                                       lodsb
4672 00001AA1 08C0                                     or     al, al
4673 00001AA3 75F4                                     jnz    short cdbo_3
4674 00001AA5 B84000                                   mov    ax, 40h
4675 00001AA8 8ED8                                     mov    ds, ax
4676                                                   ;mov    bx, [6Ch]               ; 0:46Ch = Daily timer counter (4 bytes)
4677                                                   ; 09/12/2023
4678 00001AAA 8B166C00                                 mov    dx, [6Ch]
4679 00001AAE 8B366E00                                 mov    si, [6Eh]
4680                                wait_for_key:
4681                                                   ;push  bx
4682                                                   ;mov   bx, 7
4683                                                   ; bx = 7
4684 00001AB2 B8080E                                   mov    ax, 0E08h
4685 00001AB5 CD10                                     int    10h                     ; - VIDEO - WRITE CHARACTER AND ADVANCE CURSOR (TTY WRITE)
4686                                                                                  ; AL = character, BH = display page (alpha modes)
4687                                                                                  ; BL = foreground color (graphics modes)
4688 00001AB7 B8200E                                   mov    ax, 0E20h
4689 00001ABA CD10                                     int    10h                     ; - VIDEO - WRITE CHARACTER AND ADVANCE CURSOR (TTY WRITE)
4690                                                                                  ; AL = character, BH = display page (alpha modes)
4691                                                                                  ; BL = foreground color (graphics modes)
4692 00001ABC B8080E                                   mov    ax, 0E08h
4693 00001ABF CD10                                     int    10h                     ; - VIDEO - WRITE CHARACTER AND ADVANCE CURSOR (TTY WRITE)
4694                                                                                  ; AL = character, BH = display page (alpha modes)
4695                                                                                  ; BL = foreground color (graphics modes)
4696                                                   ;pop    bx
4697                                                   ;add    bx, 18                  ; 18.2 ticks per second
4698                                                   ; 09/12/2023
4699 00001AC1 83C212                                   add    dx, 18
4700 00001AC4 83D600                                   adc    si, 0                   ; next second (if carry flag is 1)
4701                                continue_to_wait:
4702 00001AC7 B401                                     mov    ah, 1
4703 00001AC9 CD16                                     int    16h                     ; KEYBOARD - status
4704 00001ACB 741B                                     jz     short cdbo_5
4705 00001ACD B400                                     mov    ah, 0
4706 00001ACF CD16                                     int    16h                     ; KEYBOARD - READ CHAR FROM BUFFER, WAIT IF EMPTY
4707                                                                                  ; Return: AH = scan code, AL = character
4708                                
4709                                                   ; 09/12/2023
4710                                                   ;cmp    ax, 11Bh ; ESC key
4711                                                   ;jz     short cdb0_7
4712                                ;cdbo_4:
4713                                                   ;push   ax ; *
4714 00001AD1 89C2                                     mov    dx, ax ; *
4715                                
4716                                                   ; CRLF (next line)
4717                                                   ;mov    bx, 7
4718                                                   ; bx = 7
4719 00001AD3 B80D0E                                   mov    ax, 0E0Dh
4720 00001AD6 CD10                                     int    10h                     ; - VIDEO - WRITE CHARACTER AND ADVANCE CURSOR (TTY WRITE)
4721                                                                                  ; AL = character, BH = display page (alpha modes)
4722                                                                                  ; BL = foreground color (graphics modes)
4723 00001AD8 B80A0E                                   mov    ax, 0E0Ah
4724 00001ADB CD10                                     int    10h                     ; - VIDEO - WRITE CHARACTER AND ADVANCE CURSOR (TTY WRITE)
4725                                                                                  ; AL = character, BH = display page (alpha modes)
```

```
                                                                        ; BL = foreground color (graphics modes)
                                        ; 09/12/2023
                                        ;pop    ax ; *

00001ADD 81FA1B01                       cmp     dx, 11Bh
                                        ;cmp    ax, 11Bh ; ESC key (to cancel CD/DVD boot)
00001AE1 7418                           je      short cdbo_7

                                cdbo_4:         ; 10/12/2023
00001AE3 5A                             pop     dx
00001AE4 07                             pop     es
00001AE5 1F                             pop     ds
00001AE6 58                             pop     ax
00001AE7 C3                             retn
                                cdbo_5:
00001AE8 3B366E00                       cmp     si, [6Eh]
00001AEC 7504                           jnz     short cdbo_6
                                        ; 09/12/2023
00001AEE 3B166C00                       cmp     dx, [6Ch]
                                        ;cmp    bx, [6Ch]
                                cdbo_6:
00001AF2 73D3                           jnb     short continue_to_wait
00001AF4 2EFE0E[6C1B]                   dec     byte [cs:time_counter]
00001AF9 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)

00001AFB 0E                             push    cs
00001AFC 1F                             pop     ds
                                        ; 09/12/2023
00001AFD 1E                             push    ds
00001AFE 07                             pop     es
                                        ; es = ds = cs

00001AFF B8004B                         mov     ax, 4B00h
                                        ;xor    dl, dl
                                        ; 09/12/2023
00001B02 31D2                           xor     dx, dx
                                        ; dl = disk drive = 0  ; fd
                                        ;mov    si, 1C93h
00001B04 BE[591B]                       mov     si, empty_dap_buff
00001B07 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
00001B09 B81300                         mov     ax, 19
00001B0C 89F7                           mov     di, si
                                        ;mov    byte [si], 13h
                                        ;mov    [si+1], al
00001B0E AB                             stosw
                                        ;mov    [si+2], dx
00001B0F B080                           mov     al, 80h
00001B11 AB                             stosw
00001B12 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
00001B14 B90F00                         mov     cx, 15
00001B17 F3AA                           rep     stosb
                                        ; dl = disk drive = 80h ; hd
00001B19 B8004B                         mov     ax, 4B00h
00001B1C CD13                           int     13h                     ; DISK - Bootable CD-ROM - AL = TERMINATE DISK EMULATION
00001B1E 31C0                           xor     ax, ax
                                        ; 09/12/2023
                                        ;mov    dx, 80h
                                        ; dx = 80h
00001B20 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

00001B22 B80102                         mov     ax, 201h
                                        ;mov    bx, 152h
00001B25 BB[5201]                       mov     bx, disksector
                                        ;mov    cx, 1
                                        ; 09/12/2023
00001B28 41                             inc     cx ; cx = 1
                                        ;mov    dx, 80h
                                        ; dx = 80h
00001B29 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
00001B2B 72B6                           jc      short cdbo_4

00001B2D 2681BFFE0155AA                 cmp     word [es:bx+1FEh], 0AA55h
                                        ;jz     short cdbo_9
                                        ; 10/12/2023
00001B34 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
00001B36 31C0                           xor     ax, ax  ; 0
00001B38 BF007C                         mov     di, 7C00h
00001B3B 8EC0                           mov     es, ax
00001B3D 89DE                           mov     si, bx
00001B3F 06                             push    es
00001B40 57                             push    di
00001B41 B90001                         mov     cx, 100h ; 256
                                        ; 10/12/2023
                                        ;cld    ; not necessary (direction flag is already cleared)
00001B44 F3A5                           rep movsw
00001B46 8ED8                           mov     ds, ax
00001B48 BE7800                         mov     si, 78h
00001B4B 2EA1[141A]                     mov     ax, [cs:Orig_Int1Eh_Table]
00001B4F 8904                           mov     [si], ax
00001B51 2EA1[161A]                     mov     ax, [cs:Orig_Int1Eh_Table+2]
00001B55 894402                         mov     [si+2], ax
00001B58 CB                             retf

                                ; ---------------------------------------------------------------------------
                                dap_buffer: ; 16/12/2023

00001B59 13                     empty_dap_buff:     db 19
                                        ;db 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0  ; db 18 dup(0)
00001B5A 00<rep 12h>                    times 18 db 0
00001B6C 05                     time_counter:       db 5    ; 5 seconds
00001B6D 0D0A                   cd_boot_msg:        db 0Dh,0Ah
                                        ;db 'Press the ENTER key to boot from CD or DVD......',0
                                        ; 09/12/2023
00001B6F 507265737320616E79-            db 'Press any key to boot from CD or DVD ...',0
00001B78 206B657920746F2062-
00001B81 6F6F742066726F6D20-
00001B8A 4344206F7220445644-
00001B93 202E2E2E00

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

                                ; 10/12/2023 - Retro DOS v5.0 (PCDOS 7.1)

                                ; 01/10/2022 - Retro DOS v4.0 (MSDOS 5.0, classic/old MICROSOFT DOS method)
                                ; 20/12/2022 - Retro DOS v4.0 (combined kernel files, original/new method) (*)
                                ;      (*) (for using Retro DOS kernel 'MSDOS.SYS' with Retro DOS boot sector)

                                ;---------------------------------------------------------------------------
                                ; entry point from boot sector
                                ;---------------------------------------------------------------------------

                                init:               ; 27/12/2018
                                        ; MSDOS 6.0 (MSINIT.ASM)
                                        ;============================================================
                                        ;
                                        ; entry from boot sector. the register contents are:
                                        ;
                                        ;   dl = int 13 drive number we booted from
                                        ;   ch = media byte
                                        ;   bx = first data sector on disk.
                                        ;   ax = first data sector (high)
                                        ;   di = sectors/fat for the boot media.

                                        ; 10/12/2023
                                        ; Retro DOS v5.0 (IBMBIO.COM)
                                        ;============================================================
                                        ; PCDOS 7.1 IBMBIO.COM - registers from MSLOAD section
                                           ; DL = [BootDrive]
                                        ; CH = [MediaByte]
                                        ; AX:BX = First data Sector
                                        ; DS:SI = Original INT 1Eh table address
                                        ;
                                        ; Stack: INT 1Eh vector (0:78h) !not used! (dword [sp])
                                        ;        INT 1Eh table address !not used! (dword [sp+4])
                                        ; DI = 78h !not used!

                                        ; 07/04/2018
                                        ;============================================================
                                        ; Retro DOS v2.0 - registers from FD Boot Sector
                                           ; DL = [bsDriveNumber]
                                        ; DH = [bsMedia]
                                        ; AX = [bsSectors] ; Total sectors
                                        ; DS = 0, SS = 0
                                        ; BP = 7C00h

                                        ; 29/09/2023
                                        ; SP = 0FFFEh (for Retro DOS v2&v3 boot sector)
                                        ;    = 07C00h (for MSDOS 5.0 boot sector)

                                        ; 10/12/2023 - Retro DOS v5.0
                                        ; ------------------------------------------------------------
                                        ; INPUT (registers from Retro DOS v4-v5 boot sector):
                                        ;  DL = [bsDriveNumber]
                                        ;  DH = [bsMedia]
                                        ;  SS = 0
                                        ;  BP = 7C00h (boot sector address)
                                        ;
                                        ; If the boot drive is a CD (CDROM) or DVD
                                        ;    and CD boot option is enabled/requested:
                                        ;    AX = 'CD'
                                        ; If the boot drive is a FD or HD
                                        ;    or CD boot option is not enabled/requested:
                                        ;    AX <> 'CD'

                                ; 20/12/2022
                                ; Changing original MSDOS 5.0 IO.SYS init code with Retro DOS v4.0 init code.
                                %if 0
                                        cli

                                        push    ax
                                        xor     ax, ax
                                        mov     ds, ax
                                        pop     ax
                                %endif

                                ; 20/12/2022 - Retro DOS v4.0 (combined kernel)
                                ; 10/12/2023 - Retro DOS v5.0 (combined kernel)

                                KERNEL_SEGMENT equ 70h     ; (DOS BIOSDATA SEGMENT)
                                BSSECPERTRACK equ 18h      ; boot sector offset 18h (for Retro DOS & MSDOS)

                                ;---------------------------------------------------------------------------
                                ; initialization - stage 1
                                ;---------------------------------------------------------------------------
                                ; 02/06/2018 - Retro DOS v3.0

                                        ; 10/12/2023
00001B98 FC                             cld     ; may not be necessary
```

```
                                        ; 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
00001B99 3D4344                         cmp     ax, 'CD' ; is CD boot option enabled or not ?
00001B9C 7503                           jne     short init0

00001B9E E8E2FE                         call    cd_boot_option
                                init0:
00001BA1 0E                             push    cs
00001BA2 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

00001BA3 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
00001BA6 B90080                         mov     cx, 32768 ; 65536 bytes

                                        ; 21/12/2022
                                        ; 07/04/2018
00001BA9 BB7000                         mov     bx, KERNEL_SEGMENT ; 0070h
                                        ;mov    bl, KERNEL_SEGMENT
00001BAC 8EC3                           mov     es, bx
00001BAE 31FF                           xor     di, di
00001BB0 89FE                           mov     si, di

                                        ; Move KERNEL file from 1000h:0 to 0070h:0
                                        ; (Retro DOS v2 BS loads 'MSDOS.SYS' at 1000h:0000h)
00001BB2 F3A5                           rep     movsw

                                        ; 20/03/2019 - Retro DOS v4.0
00001BB4 53                             push    bx
                                        ;push   init0
00001BB5 68[B91B]                       push    init1   ; 10/12/2023
00001BB8 CB                             retf
                                ;init0:
                                        ; 10/12/2023 - Retro DOS 5.0
                                init1:
                                        ; 20/12/2022
                                        ; (combined kernel file > 64KB)

                                        ; 20/03/2019
00001BB9 B520                           mov     ch, 20h
00001BBB 8ED9                           mov     ds, cx ; 2000h
                                        ;mov    cx, 1070h
00001BBD B97010                         mov     cx, KERNEL_SEGMENT+1000h ; 20/12/2022
00001BC0 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)
00001BC2 B9E31E                         mov     cx, NXWORDCOUNT
                                        ;mov    cx, NXBYTECOUNT
                                        ;shr    cx, 1 ; 28/09/2023
                                        ;xor    si, si
                                        ;xor    di, di
00001BC5 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

00001BC7 8EC1                           mov     es, cx ; 0
00001BC9 8ED9                           mov     ds, cx ; 0

00001BCB B82205                         mov     ax, SEC9

                                        ;mov    bx, 1Eh*4  ; [0078h] ; INT 1Eh vector/pointer
00001BCE 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
00001BD0 C537                           lds     si, [bx]
```

```
                                        ; 10/12/2023 - Retro DOS v5.0
                                        ;mov    [cs:Orig_Int1Eh_Table+2], ds
                                        ;mov    [cs:Orig_Int1Eh_Table], si

00001BD2 89C7                           mov     di, ax  ; SEC9
00001BD4 B10B                           mov     cl, 11
                                        ;cld
00001BD6 F3A4                           rep     movsb

                                        ; Set INT 1Eh vector/pointer to the new DPT address
00001BD8 8ED9                           mov     ds, cx ; 0
00001BDA 8907                           mov     [bx], ax ; SEC9       ; [007Eh] ; 1Eh*4  ; offset
00001BDC 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

00001BDF 0E                             push    cs              ; Save a peck of interrupt vectors...
00001BE0 07                             pop     es
                                        ;push   cx
                                        ;push   di

                                        ; 20/12/2022
00001BE1 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
00001BE3 BE[0001]                       mov     si, RomVectors
```

```
                                        ; 10/12/2023
00001BE6 FA                             cli
                                next_int_:
00001BE7 2E                             cs      ; 16/10/2022
00001BE8 AC                             lodsb
                                        ;lods   byte ptr cs:[si] ; cs lodsb
00001BE9 98                             cbw                     ; ax = interrupt number
00001BEA D1E0                           shl     ax, 1
00001BEC D1E0                           shl     ax, 1           ; int no * 4
00001BEE 89C7                           mov     di, ax          ; interrupt vector address
00001BF0 87FE                           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
00001BF2 A5                             movsw
00001BF3 A5                             movsw

00001BF4 87FE                           xchg    si, di
00001BF6 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
00001BF8 C7064C00[ED06]                 mov     word [13h*4], block13
                                        ;mov    word ptr ds:4Ch, offset      block13 ; 13h*4
                                                                ; set up int 13       for new action
00001BFE 8C0E4E00                       mov     [13h*4+2], cs
                                        ;mov    word ptr ds:4Eh, cs ; 13h*4+2
00001C02 C7065400[9907]                 mov     word [15h*4], Int15
                                        ;mov    word ptr ds:54h, offset      Int15 ; 15h*4
                                                                ; set up int 15       for new action
00001C08 8C0E5600                       mov     [15h*4+2], cs
                                        ;mov    word ptr ds:56h, cs ; 15h*4+2
00001C0C C7066400[5907]                 mov     word [19h*4], int19
                                        ;mov    word ptr ds:64h, offset      int19 ; 19h*4
                                                                ; set up int 19       for new action
00001C12 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)
00001C16 A16800                         mov     ax, [68h]       ; 1Ah*4
00001C19 8B3E6A00                       mov     di, [6Ah]       ; 1Ah*4+2
00001C1D C7066800[AF06]                 mov     word [68h], Int1A
00001C23 8C0E6A00                       mov     [6Ah], cs

                                        ; 20/12/2022
00001C27 0E                             push    cs
00001C28 1F                             pop     ds

                                        ; 10/12/2023
00001C29 A3[AB06]                       mov     [Orig1A], ax
00001C2C 893E[AD06]                     mov     [Orig1A+2], di

00001C30 A1[0601]                       mov     ax, [Old13]     ; save old13 in orig13 also
00001C33 A3[B400]                       mov     [Orig13], ax
00001C36 A1[0801]                       mov     ax, [Old13+2]
00001C39 A3[B600]                       mov     [Orig13+2], ax
                                                                ; ;
00001C3C FB                             sti
00001C3D 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
00001C3F 88C1                           mov     cl, al

                                        ; 12/12/2022
00001C41 A801                           test    al, 1
                                        ;test   ax, 1           ; floppy drives       present?
00001C43 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
00001C45 52                             push    dx
                                        ;push   di
00001C46 06                             push    es

00001C47 B408                           mov     ah, 8
00001C49 B200                           mov     dl, 0
00001C4B 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
00001C4D 7202                           jc      short _gdskp_error
                                        ;;mov   [cs:flp_drvs], dl
                                        ; 20/12/2022
                                        ; ds = cs
                                        ;mov    [flp_drvs], dl
00001C4F 88D1                           mov     cl, dl
                                _gdskp_error:
                                        ; 20/12/2022
00001C51 07                             pop     es
                                        ;pop    di
00001C52 5A                             pop     dx
                                        ;pop    cx
                                        ;pop    bx
                                        ;pop    ax

00001C53 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
00001C55 08C9                           or      cl, cl ; [flp_drvs]
00001C57 7403                           jz      short _set_fake_flpdrv
00001C59 49                             dec     cx
00001C5A EB0B                           jmp     short got_num_flp_drvs
                                ; ---------------------------------------------------------------------------

                                _set_fake_flpdrv:
                                        ; 20/12/2022
                                        ; ds = cs
                                        ;inc    cl      ; cl = 1
                                        ; 10/12/2023
00001C5C 41                             inc     cx      ; cl = 1
00001C5D 880E[131A]                     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
00001C61 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
00001C63 D0C1                           rol     cl, 1
00001C65 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
00001C67 80E103                         and     cl, 3
00001C6A 7505                           jnz     short notsingle         ; zero means single drive system
                                        ; 20/12/2022
00001C6C 41                             inc     cx
                                        ;inc    ax              ; pretend it's a two drive system
                                settwodrive:                            ; set this to two fakedrives
                                        ; 20/12/2022
                                        ; ds = cs
00001C6D 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
00001C71 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
00001C72 88C8                           mov     al, cl  ; 26/05/2019
```

```
00001C74 F6C280                         test    dl, 80h         ; boot from floppy ?
00001C77 7502                           jnz     short gothrd    ; no.
00001C79 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
00001C7B 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

00001C7D 51                             push    cx ; (***)      ; save number of floppies and media byte

00001C7E 88EC                           mov     ah, ch          ; FAT ID to AH
00001C80 50                             push    ax ; (**)       ; save boot drive number and media byte

                                ; let model_byte, secondary_model_byte be set here!!!

00001C81 B4C0                           mov     ah, 0C0h
00001C83 CD15                           int     15h     ; SYSTEM - GET CONFIGURATION (XT after 1/10/86,AT mdl 3x9,CONV,XT286,PS)
00001C85 7215                           jb      short no_rom_system_conf ; just       use Model_Byte
00001C87 80FC00                         cmp     ah, 0
00001C8A 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
00001C8C 268A4702                       mov     al, [es:bx+2]  ; [es:bx+ROMBIOS_DESC.bios_sd_modelbyte]
00001C90 A2[AF05]                       mov     [model_byte], al
                                        ;mov    [cs:model_byte], al
                                                                ; get/save model byte
00001C93 268A4703                       mov     al, [es:bx+3]  ; [es:bx+ROMBIOS_DESC.bios_sd_scnd_modelbyte]
00001C97 A2[B005]                       mov     [secondary_model_byte], al
                                        ;mov    [cs:secondary_model_byte], al
                                                                ; get/save secondary model byte
00001C9A EB0C                           jmp     short turn_timer_on
                                ;-------------------------------------------------------------------------------

                                no_rom_system_conf:
00001C9C BEFFFF                         mov     si, 0FFFFh
00001C9F 8EC6                           mov     es, si
                                        ; 20/12/2022
00001CA1 26A00E00                       mov     al, [es:0Eh]   ; get model byte (from 0FFFFh:0Eh)
00001CA5 A2[AF05]                       mov     [model_byte], al
                                        ;mov    [cs:model_byte], al ; save model byte
                                turn_timer_on:
00001CA8 B020                           mov     al, 20h ; ' '   ; turn on the timer
00001CAA 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
00001CAC 803E[AF05]00                   cmp     byte [model_byte], 0
                                        ;cmp    byte [cs:model_byte], 0 ; next to last      byte in rom bios
00001CB1 7510                           jnz     short not_olivetti_m24 ; skip for all other machines
                                                                ; (except olivetti m24)
00001CB3 E466                           in      al, 66h         ; is 8530 installed?
00001CB5 A820                           test    al, 20h
00001CB7 740A                           jz      short not_olivetti_m24 ; we're done if not
00001CB9 B00F                           mov     al, 0Fh         ; double check
00001CBB E650                           out     50h, al
00001CBD E450                           in      al, 50h
00001CBF A801                           test    al, 1           ; this test was       copied from olivetti
00001CC1 7414                           jz      short skip_aux_port_init ; take      this branch if 8530 installed

                                not_olivetti_m24:
00001CC3 B003                           mov     al, 3           ; init com4
```

```
00001CC5 E8DB09                         call    aux_init
00001CC8 B002                           mov     al, 2           ; init com3
00001CCA E8D609                         call    aux_init
00001CCD B001                           mov     al, 1           ; init com2
00001CCF E8D109                         call    aux_init
00001CD2 30C0                           xor     al, al          ; init com1
00001CD4 E8CC09                         call    aux_init

                                skip_aux_port_init:
00001CD7 B002                           mov     al, 2           ; init lpt3
00001CD9 E8BF09                         call    print_init
00001CDC B001                           mov     al, 1           ; init lpt2
00001CDE E8BA09                         call    print_init
00001CE1 30C0                           xor     al, al          ; init lpt1
00001CE3 E8B509                         call    print_init

                                        ; 11/12/2023 - Retro DOS v5.0 (PCDOS 7.1 IBMBIO.COM)
                                        ;mov    di, 534h        ; offset INITSPOT
                                        ;;mov   di, INITSPOT    ; 0534h
                                        ;                       ; IBMDOS.COM's first cluster - high word
                                        ;                       ; 520h (the 2nd entry of root dir) + 14h
                                        ;mov    ax, [di]
                                        ;mov    [firstcluster_hw], ax

00001CE6 31D2                           xor     dx, dx  ; 0
00001CE8 8EDA                           mov     ds, dx          ; to initialize       print screen vector
00001CEA 8EC2                           mov     es, dx
00001CEC 31C0                           xor     ax, ax
                                        ; 16/10/2022
00001CEE BF3405                         mov     di, INITSPOT    ; 0534h
                                        ;mov    di, 534h        ; INITSPOT (0000h:0534h)
                                                                ; IBM wants 4 zeros here
00001CF1 AB                             stosw
00001CF2 AB                             stosw
00001CF3 8CC8                           mov     ax, cs          ; fetch segment
00001CF5 C7066C00[0E06]                 mov     word [BRKADR], cbreak
                                        ;mov    word ptr ds:6Ch, offset       cbreak ; [BRKADR]
                                                                ; break entry point
00001CFB A36E00                         mov     [BRKADR+2], ax
                                        ;mov    ds:6Eh, ax      ; vector for break
00001CFE C706A400[8206]                 mov     word [CHROUT*4], outchr
                                        ;mov    word ptr ds:0A4h, offset outchr       ; [CHROUT*4]
00001D04 A3A600                         mov     [CHROUT*4+2], ax
                                        ;mov    ds:0A6h, ax     ; [CHROUT*4+2]

00001D07 BF0400                         mov     di, 4
00001D0A BB[1406]                       mov     bx, intret ; 19/10/2022
                                        ;mov    bx, offset intret ; intret (cs:intret)
                                                                ; will initialize rest of interrupts
00001D0D 93                             xchg    ax, bx
00001D0E AB                             stosw                   ; location 4
00001D0F 93                             xchg    ax, bx          ; cs:
00001D10 AB                             stosw                   ; int 1 ; location 6
00001D11 83C704                         add     di, 4
00001D14 93                             xchg    ax, bx
00001D15 AB                             stosw                   ; location 12
00001D16 93                             xchg    ax, bx          ; cs:
00001D17 AB                             stosw                   ; int 3 ; location 14
00001D18 93                             xchg    ax, bx
00001D19 AB                             stosw                   ; location 16
00001D1A 93                             xchg    ax, bx          ; cs:
00001D1B AB                             stosw                   ; int 4 ; location 18

                                ;       ; 20/12/2022
                                ;       ; (https://stanislavs.org/helppc/bios_data_area.html)
                                ;       Address Size   Description    (BIOS/DOS Data Area)
                                ;
                                ;       50:00   byte    Print screen status byte
                                ;                        00 = PrtSc not active,
                                ;                        01 = PrtSc in progress
                                ;                        FF = error
                                ;       50:01   3 bytes Used by BASIC
                                ;       50:04   byte    DOS single diskette mode flag, 0=A:, 1=B:
                                ;       50:05   10bytes POST work area
                                ;       50:0F   byte    BASIC shell flag; set to 2 if current shell
                                ;       50:10   word    BASICs default DS value (DEF SEG)
                                ;       50:12   dword   Pointer to BASIC INT 1C interrupt handler
                                ;       50:16   dword   Pointer to BASIC INT 23 interrupt handler
                                ;       50:1A   dword   Pointer to BASIC INT 24 disk error handler
                                ;       50:20   word    DOS dynamic storage
                                ;       50:22   14bytes DOS diskette initialization table (INT 1E)
                                ;       50:30   4bytes  MODE command
                                ;       70:00           I/O drivers from IO.SYS/IBMBIO.COM

00001D1C 89160005                       mov     [0500h], dx ; 0
                                        ;mov    ds:500h, dx     ; set print screen & break = 0
00001D20 89160405                       mov     [LSTDRV], dx    ; [0504h]
                                        ;mov    ds:504h, dx     ; clean out last drive spec

                                ; we need to initialize the cs:motorstartup variable from the disk
                                ; parameter table at sec9. the offsets in this table are defined in
                                ; the disk_parms struc in msdskprm.inc. 2 locs

                                        mov     al, [SEC9+0Ah] ; 16/10/2022
                                        ;mov    al, ds:52Ch     ; [SEC9+DISK_PARMS.DISK_MOTOR_STRT]
                                                                ; [522h+0Ah]
                                        ; 20/12/2022
                                        ; ds = 0

00001D27 2EA2[2601]                     mov     [cs:motorstartup], al
00001D2B 2E803E[AF05]FD                 cmp     byte [cs:model_byte], 0FDh ; is this an old rom?
00001D31 720B                           jb      short no_diddle         ; no
00001D33 C7062B050F02                   mov     word [SEC9+09h], 20Fh
                                        ;mov    word ptr ds:52Bh, 20Fh ; [SEC9+DISK_PARMS.DISK_HEAD_STTL], 0200h+NORMSETTLE
                                                                ; set head settle and motor start on pc-1 pc-2 pc-xt hal0
00001D39 C6062205DF                     mov     byte [SEC9+0], 0DFh
                                        ;mov    byte ptr ds:522h, 0DFh ; [SEC9+DISK_PARMS.DISK_SPECIFY_1]
                                                                ;  set 1st specify byte       on pc-1 pc-2 pc-xt hal0
                                no_diddle:
00001D3E CD12                           int     12h             ; MEMORY SIZE -
                                                                ; Return: AX = number of contiguous 1K blocks of memory
00001D40 B106                           mov     cl, 6
00001D42 D3E0                           shl     ax, cl          ; convert memory size to 16-byte blocks     (segment no.)

                                        ; 20/12/2022
                                        ; 03/07/2018 - 27/12/2018
                                        ;pop    cx ; (**)
                                        ;mov    [cs:drvfat], cx

00001D44 50                             push    ax ; (*)        ; save real top       of memory

                                        ; 27/12/2018 - (MSDOS 6.0, 6.21)

                                ;M068 - BEGIN
```

```
                                ;------ Check if an RPL program is present at TOM and do not tromp over it
                                
                                        ; 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
00001D45 C51EBC00                       lds    bx, [2Fh*4]
                                
00001D49 817F035250                     cmp    word [bx+3], 'RP' ; 'RPL'
                                        ;cmp   word ptr [bx+3], 'PR' ; 'RPL'
00001D4E 750F                           jnz    short SkipRPL
00001D50 807F054C                       cmp    byte [bx+5], 'L'
                                        ;cmp   byte ptr [bx+5], 'L'
00001D54 7509                           jnz    short SkipRPL
00001D56 89C2                           mov    dx, ax          ; get TOM into DX
00001D58 B8064A                         mov    ax, 4A06h       ; (multMULT shl       8) + multMULTRPLTOM
00001D5B CD2F                           int    2Fh             ; Get new TOM from any RPL
00001D5D 89D0                           mov    ax, dx
                                SkipRPL:
                                        ; 20/12/2022
                                        ;pop   bx
                                        ;pop   ds
                                
                                ;M068 - END
                                        ; 20/12/2022
                                        ; 27/12/2018
00001D5F 0E                             push   cs
00001D60 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
00001D61 83E840                         sub    ax, 64
00001D64 A3[061A]                       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)
00001D67 58                             pop    ax ; (*)         ; get back real top of memory
                                
                                
                                        ; 20/12/2022
                                        ; 27/12/2018
00001D68 59                             pop    cx ; (**)
00001D69 890E[FC19]                     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)
00001D6D BAD904                         mov    dx, SYSINITSEG ; 17/10/2022
00001D70 8EDA                           mov    ds, dx
                                
                                ; set pointer to resident device driver chain
                                
                                        ; 17/10/2022
00001D72 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]
00001D78 8C0E[7702]                     mov    [DEVICELIST+2], cs
                                        ;mov   [275h], cs
                                        ;;mov  word ptr ds:275h, cs ; [SYSINIT+DEVICE_LIST+2]
                                
00001D7C A3[9402]                       mov    [MEMORYSIZE], ax
                                        ;mov   [292h], ax
                                        ;;mov  ds:292h, ax     ; [SYSINIT+MEMORY_SIZE]
                                
00001D7F FEC1                           inc    cl
00001D81 880E[9802]                     mov    [DEFAULTDRIVE], cl
                                        ;mov   [296h], cl
                                        ;;mov  ds:296h, cl     ; [SYSINIT+DEFAULT_DRIVE]
                                
                                        ;mov   word [CURRENTDOSLOCATION], 0AF8h ; 10/12/2023
00001D85 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
00001D8B 8EC2                           mov    es, dx ; SYSINITSEG
00001D8D 31C0                           xor    ax, ax ; 0
00001D8F 8ED8                           mov    ds, ax          ; segment 0
                                        ;mov   ax, ds:3Eh      ; [0Fh*4+2]
00001D91 A13E00                         mov    ax, [0Fh*4+2]  ; segment for INT 0Fh
                                        ; 18/10/2022
00001D94 263B06[9402]                   cmp    ax, [es:MEMORYSIZE] ; es:292h
                                        ;cmp   ax, es:292h    ; [ES:MEMORY_SIZE]  ; (condition 1)
00001D99 7605                           jbe    short resetintf
00001D9B 3D00F0                         cmp    ax, 0F000h     ; (condition 2)
00001D9E 750A                           jnz    short keepintf
                                resetintf:
00001DA0 C7063C00[1406]                 mov    word [0Fh*4], intret
```

```
                                        ;mov    word ptr ds:3Ch, offset      intret ; [0Fh*4]
00001DA6 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

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

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

                                        ;test   cl, 00010000b ; 10h
00001DB0 F6C110                         test    cl, 10h         ; extended keyboard ?
00001DB3 740A                           jz      short org_key   ; no, original keyboard

                                        ; 20/12/2022
                                        ;  ds = cs
00001DB5 C606[7E04]10                   mov     byte [keyrd_func], 10h ; extended keyboard
00001DBA 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

00001DBF 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
00001DC2 C706[0000][4B08]               mov     word [hdrv_pat], harddrv

                                        ; 20/12/2022
                                        ; 28/12/2018 - Retro DOS v4.0 (MSDOS 6.21)
00001DC8 58                             pop     ax ; (***)      ; number of floppies and FAT ID

00001DC9 30E4                           xor     ah, ah          ; chuck fat id byte
00001DCB A2[7500]                       mov     [drvmax], al    ; remember which drive is hard disk
00001DCE A2[2501]                       mov     [dsktnum], al   ; and set initial number of drives
00001DD1 D1E0                           shl     ax, 1
00001DD3 0106[601A]                     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))
                                        ; ........
00001DD7 1E                             push    ds
00001DD8 B800F0                         mov     ax, 0F000h
00001DDB 8ED8                           mov     ds, ax

00001DDD 813EEAFF434F                   cmp     word [0FFEAh], 'CO' ; 'COMPAQ'
00001DE3 751F                           jne     short skip_mode2
00001DE5 813EECFF4D50                   cmp     word [0FFECh], 'MP'
00001DEB 7517                           jne     short skip_mode2
00001DED 813EEEFF4151                   cmp     word [0FFEEh], 'AQ'
00001DF3 750F                           jne     short skip_mode2

00001DF5 B800E4                         mov     ax, 0E400h      ; get advanced system info (COMPAQ ROMBIOS)
00001DF8 CD15                           int     15h
00001DFA 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
00001DFC 83CB40                         or      bx, 40h         ; enable mode 2 (dual harddisk controller)
00001DFF B880E4                         mov     ax, 0E480h      ; set advanced system info (COMPAQ ROMBIOS)
00001E02 CD15                           int     15h
                                skip_mode2:
00001E04 1F                             pop     ds
                                        ; ........

00001E05 B280                           mov     dl, 80h
00001E07 B408                           mov     ah, 8
00001E09 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
00001E0B 7204                           jc      short enddrv
00001E0D 8816[5F1A]                     mov     [hnum], dl      ; save number of hard disk drives
                                enddrv:
                                        ; 21/12/2022
00001E11 0E                             push    cs
00001E12 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)
                                
00001E13 30D2                           xor     dl, dl

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

00001E15 C606[2C01]09                   mov     byte [eot], 9
00001E1A 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.

00001E1D 803E[131A]01                   cmp     byte [fakefloppydrv], 1
00001E22 750B                           jnz     short loop_drive
00001E24 8B3D                           mov     di, [di]       ; [di+BDS.link]
                                                               ; di <- first bds link
00001E26 8B3D                           mov     di, [di]       ; [di+BDS.link]
                                                               ; di <- second bds link
00001E28 C705FFFF                       mov     word [di], 0FFFFh ; -1 ; set end of link
00001E2C E98801                         jmp     dohard         ; allocate/initialise bds for harddrives
                                ;---------------------------------------------------------------------------

                                loop_drive:
00001E2F 3A16[7500]                     cmp     dl, [drvmax]
00001E33 7203                           jb      short got_more
00001E35 E97B01                         jmp     done_drives
                                ;---------------------------------------------------------------------------

                                got_more:
                                        ; 12/12/2023
                                        ;xor    cx, cx         ; zero all flags
00001E38 8B3D                           mov     di, [di]       ; [di+BDS.link]
                                                               ; get next bds
                                        ; ........
                                        ; 10/12/2023 - Retro DOS v5.0
                                        ; (PCDOS 7.1 IBMBIO.COM BIOSDATA:2046h)
00001E3A 83FFFF                         cmp     di, 0FFFFh      ; end of link ?
00001E3D 7516                           jne     short not_last_bds
00001E3F 88D0                           mov     al, dl          ; drive number (0 based)
00001E41 98                             cbw
00001E42 01C0                           add     ax, ax
00001E44 05[3C05]                       add     ax, dskdrvs
00001E47 A3[601A]                       mov     [last_dskdrv_table], ax
00001E4A 8B3E[621A]                     mov     di, [end_of_bdss]
00001E4E E8FD09                         call    xinstall_bds
00001E51 FE0E[7500]                     dec     byte [drvmax]
                                not_last_bds:
                                        ; ........

00001E55 B600                           mov     dh, 0          ; ff48tpi
                                                               ; set form factor to 48      tpi
00001E57 C606[101A]28                   mov     byte [num_cyln], 40 ; 40 tracks per side

                                        ; 20/12/2022
                                        ;push   ds ; 11/05/2019
00001E5C 57                             push    di
00001E5D 52                             push    dx
                                        ;push   cx ; not necessary (10/12/2023)
00001E5E 06                             push    es ; ((*)) ; 20/12/2022

                                        ; ..........
                                        ; 10/12/2023 - Retro DOS v5.0 (PCDOS 7.1 IBMBIO.COM)
                                        ;xor    bx, bx
                                        ;xor    cx, cx
00001E5F 52                             push    dx  ; dl = drive number

00001E60 B408                           mov     ah, 8
00001E62 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
00001E64 58                             pop     ax  ; al = drive number
00001E65 7303                           jnc     short chk_drv_type
00001E67 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)
                                        ;
00001E6A 80FB10                         cmp     bl, 10h        ; ATAPI Removable Media Device
00001E6D 7554                           jne     short not_atapi_removable

                                        ; save ds:si
00001E6F 1E                             push    ds
                                        ;push   si      ; not necessary (10/12/2023)

00001E70 88C2                           mov     dl, al
00001E72 83EC1A                         sub     sp, 26
00001E75 31C0                           xor     ax, ax ; 0
00001E77 50                             push    ax
00001E78 B81E00                         mov     ax, 30
00001E7B 50                             push    ax
```

```
6086 00001E7C 89E6                              mov     si, sp          ; DS:SI = segment:offset pointer to Result Buffer
6087 00001E7E 16                                push    ss
6088 00001E7F 1F                                pop     ds
6089 00001E80 B448                              mov     ah, 48h
6090 00001E82 CD13                              int     13h             ; DISK - IBM/MS Extension
6091                                                                    ; GET DRIVE PARAMETERS (DL - drive, DS:SI - buffer)
6092 00001E84 7239                              jb      short ext_gdp_err
6093 00001E86 8B4408                            mov     ax, [si+8]      ; physical number of heads
6094 00001E89 A3[0E1A]                          mov     [num_heads], ax
6095 00001E8C 8B4404                            mov     ax, [si+4]      ; physical number of cylinders
6096 00001E8F A3[101A]                          mov     [num_cyln], ax
6097 00001E92 8A440C                            mov     al, [si+0Ch]    ; physical number of sectors per track
6098 00001E95 A2[121A]                          mov     [sec_trk], al
6099 00001E98 3A06[2C01]                        cmp     al, [eot]
6100 00001E9C 7603                              jbe     short _eotok
6101 00001E9E A2[2C01]                          mov     [eot], al
6102
6103                                    _eotok:             ; 10/12/2023
6104                                                ;xor    al, al
6105 00001EA1 31C9                              xor     cx, cx ; 0
6106 00001EA3 F6440210                          test    byte [si+2], 10h ; information flags
6107                                                                    ; bit 4 = Device has change line support
6108 00001EA7 7403                              jz      short not_chgline_sup
6109                                                ;or     al, 2           ; change line support
6110 00001EA9 80C902                            or      cl, 2
6111                                    not_chgline_sup:
6112 00001EAC 83C41E                            add     sp, 30
6113                                                ;pop    si      ; (10/12/2023)
6114 00001EAF 1F                                pop     ds
6115                                                ;
6116 00001EB0 07                                pop     es      ; es=cs=ds (21/12/2022)
6117                                                ;pop    cx      ; (10/12/2023)
6118 00001EB1 5A                                pop     dx
6119 00001EB2 5F                                pop     di
6120                                                ;pop    ds      ; (21/12/2022)
6121
6122                                                ; 10/12/2023
6123 00001EB3 F6C102                            test    cl, 2
6124                                                ;test   al, 2
6125                                                ;jz     short gotother_j
6126 00001EB6 7450                              jz      short gotother
6127                                                ;or     cl, al
6128 00001EB8 C606[7700]01                      mov     byte [fhave96], 1 ; Device has change line support
6129                                    gotother_j:
6130 00001EBD EB49                              jmp     short gotother
6131                                    ext_gdp_err:
6132 00001EBF 83C41E                            add     sp, 30
6133                                                ;pop    si      ; (10/12/2023)
6134 00001EC2 1F                                pop     ds
6135
6136                                                ; 10/12/2023
6137                                    not_atapi_removable:
6138                                                ; ...........
6139
6140
6141                                    ; if cmos is bad, it gives es,ax,bx,cx,dh,di=0. cy=0.
6142                                    ; in this case, we are going to put bogus informations to bds table.
6143                                    ; we are going to set ch=39,cl=9,dh=1 to avoid divide overflow when
6144                                    ; they are calculated at the later time. this is just for the diagnostic
6145                                    ; diskette which need msbio,msdos to boot up before it sets cmos.
6146                                    ; this should only happen with drive b.
6147
6148 00001EC3 80FD00                                cmp     ch, 0           ; if ch=0, then       cl,dh=0 too.
6149 00001EC6 7505                              jnz     short pfr_ok
6150
6151                                                ;mov    ch, 39          ; rom gave wrong info.
6152                                                ;mov    cl, 9           ; let's default to 360k.
6153                                                ; 20/12/2022
6154 00001EC8 B90927                            mov     cx, 2709h
6155
6156 00001ECB B601                              mov     dh, 1
6157                                    pfr_ok:
6158                                                ;inc    dh              ; make number of heads 1-based
6159                                                ;mov    [num_heads], dh       ; save parms returned by rom
6160                                                ; 10/12/2023
6161 00001ECD 86F2                              xchg    dl, dh
6162 00001ECF 30F6                              xor     dh, dh
6163 00001ED1 42                                inc     dx              ; make number of heads 1-based
6164 00001ED2 8916[0E1A]                        mov     [num_heads], dx
6165
6166                                                ;inc    ch              ; make number of cylinders 1-based
6167                                                ;and    cl, 3Fh
6168                                                ;mov    [sec_trk], cl
6169                                                ;mov    [num_cyln], ch ; assume less than 256 cylinders!!
6170                                                ; 10/12/2023
6171 00001ED6 88CA                              mov     dl, cl
6172 00001ED8 80E23F                            and     dl, 3Fh
6173 00001EDB 8816[121A]                        mov     [sec_trk], dl
6174 00001EDF 86E9                              xchg    cl, ch
6175 00001EE1 D0C5                              rol     ch, 1
6176 00001EE3 D0C5                              rol     ch, 1
6177 00001EE5 80E503                            and     ch, 3
6178 00001EE8 41                                inc     cx              ; make number of cylinders 1-based
6179 00001EE9 890E[101A]                        mov     [num_cyln], cx
6180
6181                                    ; make sure that eot contains the max number of sec/trk in system of floppies
6182
6183                                                ;mov    cl, [sec_trk] ; 10/12/2023
6184                                                ;cmp    cl, [eot]       ; may set carry
6185                                                ;;jbe   short eot_ok
6186                                                ;; 09/12/2022
6187                                                ;;jne   short eotok  ; wrong ! 14/08/2023
6188                                                ;; 14/08/2023
6189                                                ;jbe    short eotok
6190                                                ;mov    [eot], cl
6191                                                ; 10/12/2023
6192 00001EED 3A16[2C01]                        cmp     dl, [eot] ; dl = [sec_trk]
6193 00001EF1 7604                              jbe     short eotok
6194 00001EF3 8816[2C01]                        mov     [eot], dl
6195                                    ;eot_ok:
6196                                    eotok:
6197                                                ; 10/12/2023
6198                                                ; !!!
6199                                                ; (following pops are moved to 'chk_changeline' procedure)
6200                                                ;
6201                                                ; 20/12/2022
6202                                                ;pop    es ; ((*)) es = cs = ds
6203                                                ;;pop   cx      ; 10/12/2023
6204                                                ;pop    dx
6205                                                ;pop    di
6206
6207                                                ; 20/12/2022
6208                                                ;pop    ds
6209
```

```
                                ; Check     for presence of       changeline
                                
                                ; 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
00001EF7 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.
                                
00001EFA 803E[101A]28                       cmp     byte [num_cyln], 40
00001EFF 750B                               jnz     short try_80
00001F01 803E[121A]09                       cmp     byte [sec_trk], 9
00001F06 765F                               jbe     short nextdrive
                                gotother:
                                            ; 10/12/2023
                                            ; ch = 0, cl = 2 or 0
                                
00001F08 B607                               mov     dh, 7           ; ffOther
                                                                    ; we have a "strange" medium
00001F0A EB5B                               jmp     short nextdrive
                                ;--------------------------------------------------------------------------
                                
                                ; 80 cylinders and 9 sectors/track => 720 kb device
                                ; 80 cylinders and 15 sec/trk => 96 tpi medium
                                
                                try_80:
00001F0C 803E[101A]50                       cmp     byte [num_cyln], 80
00001F11 75F5                               jnz     short gotother
00001F13 B609                               mov     dh, 9 ; ff288  ; assume 2.88 MB drive
00001F15 803E[121A]24                       cmp     byte [sec_trk], 36 ; is it ?
00001F1A 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
                                
00001F1C 803E[121A]0F                       cmp     byte [sec_trk], 15
00001F21 740B                               jz      short got96
                                
00001F23 803E[121A]09                       cmp     byte [sec_trk], 9
00001F28 75DE                               jnz     short gotother
                                
00001F2A B602                               mov     dh, 2           ; ffSmall
00001F2C EB39                               jmp     short nextdrive
                                ; --------------------------------------------------------------------------
                                
                                got96:
00001F2E B601                               mov     dh, 1           ; ff96tpi
00001F30 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:
00001F32 59                                 pop     cx ; near call return address
                                
                                            ; (pop es, dx, di for 'eotok' and 'noparmsfromrom' procs)
00001F33 07                                 pop     es ; es=cs=ds ; 21/12/2022
                                            ;pop    cx      ; (10/12/2023)
00001F34 5A                                 pop     dx
00001F35 5F                                 pop     di ; BDS address/offset
                                
00001F36 51                                 push    cx ; near call return address
                                
                                            ;xor    cx, cx ; 0
                                            ;push   cx
00001F37 52                                 push    dx
                                
00001F38 B415                               mov     ah, 15h
00001F3A 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
00001F3C 5A                                 pop     dx
                                            ;pop    cx
00001F3D B90000                             mov     cx, 0
00001F40 720D                               jc      short chk_chgl_1
                                
00001F42 80FC02                             cmp     ah, 2           ; is there changeline?
00001F45 7508                               jne     short chk_chgl_2 ; *
                                
00001F47 80C902                             or      cl, 2
                                            ;or     cl, ah ; 2
00001F4A C606[7700]01                       mov     byte [fhave96], 1 ; fchangeline
                                            ; cf = 0
                                chk_chgl_1:
                                chk_chgl_2:
00001F4F C3                                 retn
                                
                                ;chk_chgl_2:       ; *
```

```
                                ;           ; 10/12/2023
                                ;           ; ah = 1 ; harddisk type (ah = 3) return not possible here for floppies
                                ;           ;stc
                                ;           ; cf = 1
                                ;           retn

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

                                ; we have an old rom, so we either have a 48tpi or 96tpi drive. if the drive
                                ; has changeline, we assume it is a 96tpi, otherwise we treat it as a 48tpi.

                                noparmsfromrom:
                                            ; 10/12/2023
                                            ; !!!
                                            ; (following pops are moved to 'chk_changeline' procedure)
                                            ;
                                            ; 20/12/2022
                                            ;pop   es ; ((*))
                                            ;;pop  cx     ; (10/12/2023)
                                            ;pop   dx
                                            ;pop   di

                                            ; 20/12/2022
                                            ;pop   ds
                                ; 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
00001F50 E8DFFF                             pop    dx
00001F53 7212                               ;pop   cx
                                            mov    cx, 0 ; 12/12/2023
                                            jc     short nextdrive

                                            cmp    ah, 2          ; is there changeline?
                                            jnz    short nextdrive

                                            or     cl, 2
                                            mov    byte [fhave96], 1 ; fchangeline
                                %endif
                                            ; 10/12/2023
                                            call   chk_changeline
                                            jc     short nextdrive

                                            ; change line support, [fhave96] = 1

00001F55 C606[101A]50                       mov    byte [num_cyln], 80
00001F5A B601                               mov    dh, 1          ; ff96tpi
00001F5C B00F                               mov    al, 15
00001F5E 3A06[2C01]                         cmp    al, [eot]
00001F62 7603                               jbe    short nextdrive
00001F64 A2[2C01]                           mov    [eot], al
                                ; ------------------------------------------------------------------------

                                ;eot_ok2:
                                nextdrive:
                                            ; 10/12/2023
                                            ; ch = 0, cl = 2 or 0

00001F67 80C920                             or     cl, 20h ; fi_own_physical
                                                           ; set this true       for all drives
00001F6A 88D7                               mov    bh, dl         ; save int13 drive number

                                ; we need to do special things if we have a single drive system and are setting
                                ; up a logical drive. it needs to have the same int13 drive number as its
                                ; counterpart, but the next drive letter. also reset ownership flag.
                                ; we detect the presence of this situation by examining the flag single for the
                                ; value 2.
00001F6C 803E[7800]02                       cmp    byte [single], 2
00001F71 7505                               jnz    short not_special
00001F73 FECF                               dec    bh             ; int13 drive number same for logical drive
00001F75 80F120                             xor    cl, 20h ; fi_own_physical
                                                           ; reset ownership flag for logical drive
                                not_special:

                                ; the values that we put in for BDS_RBPB.BPB_HEADS and
                                ; BDS_RBPB.BPB_SECTORSPERTRACK will only remain if the
                                ; form factor is of type "ffother".

                                ;xor   ax, ax         ; fill BDS for drive
                                ;mov   al, [num_heads]
                                ; 10/12/2023 - Retro DOS v5.0 (PCDOS 7.1 IBMBIO.COM) ; *
00001F78 A1[0E1A]               mov    ax, [num_heads]
                                ;mov   [di+36h], ax   ; [di+BDS.rheads]
00001F7B 894552                 mov    [di+52h], ax   ; [di+BDS.rheads] ; *
00001F7E 31C0                   xor    ax, ax         ; *
00001F80 A0[121A]               mov    al, [sec_trk]
                                ;mov   [di+34h], ax   ; [di+BDS.rsecpertrack]
00001F83 894550                 mov    [di+50h], ax   ; [di+BDS.rsecpertrack] ; *
                                ;mov   [di+23h], cx   ; [di+BDS.flags]
00001F86 894D3F                 mov    [di+3Fh], cx   ; [di+BDS.flags] ; *
                                ;mov   [di+22h], dh   ; [di+BDS.formfactor]
00001F89 88753E                 mov    [di+3Eh], dh   ; [di+BDS.formfactor] ; *
00001F8C 885505                 mov    [di+5], dl     ; [di+BDS.drivelet]
00001F8F 887D04                 mov    [di+4], bh     ; [di+BDS.drivenum]
                                ;mov   bl, [num_cyln]
                                ;mov   [di+25h], bl   ; [di+BDS.cylinders]
                                ; 10/12/2023
00001F92 A1[101A]               mov    ax, [num_cyln]
00001F95 894541                 mov    [di+41h], ax   ; [di+BDS.cylinders] ; *

00001F98 803E[7800]01           cmp    byte [single], 1 ; Special case for single drive system
00001F9D 7510                   jnz    short no_single
                                ;mov   byte [single], 2 ; Don't forget we have
                                                        ; single drive system
                                ; 10/12/2023
00001F9F FE06[7800]             inc    byte [single]  ; [single] = 2
                                ; 18/12/2022
00001FA3 80C910                 or     cl, 10h
                                ;or    cx, 10h        ; fi_am_mult
                                                      ; set that this       is one of several drives
                                ;or    [di+23h], cx   ; [di+BDS.flags]
00001FA6 094D3F                 or     [di+3Fh], cx   ; [di+BDS.flags] ; *
                                                      ; save flags
00001FA9 8B3D                   mov    di, [di]       ; [di+BDS.link]
```

```
                                                        ; move to next BDS in list
00001FAB FEC2                           inc     dl      ; add a number
00001FAD EBB8                           jmp     short nextdrive       ; Use same info       for BDS as previous
                                ; ---------------------------------------------------------------------------

                                no_single:
                                        ;inc    dl
                                        ; 18/12/2022
00001FAF 42                             inc     dx
00001FB0 E97CFE                         jmp     loop_drive
                                ; ---------------------------------------------------------------------------

                                        ; 11/12/2023 - Retro DOS v5.0 IO.SYS/IBMBIO.COM
                                        ; (PCDOS 7.1 IBMBIO.COM - BIOSDATA:21E8h)
                                done_drives:
                                        ;mov    word [di+BDS.link], -1
00001FB3 C705FFFF                       mov     word [di], -1  ; set link to null

                                ; set up all the hard drives in   the system

                                        ; 20/12/2022
                                        ; 28/12/2018 - Retro DOS v4.0 (MSDOS 6.21)
                                dohard:
00001FB7 8A36[5F1A]                     mov     dh, [hnum]
00001FBB 08F6                           or      dh, dh          ; done if no hardfiles
00001FBD 7459                           jz      short static_configure
00001FBF B280                           mov     dl, 80h
                                dohard1:
00001FC1 52                             push    dx
00001FC2 8B3E[621A]                     mov     di, [end_of_bdss]
00001FC6 8A1E[7500]                     mov     bl, [drvmax]
00001FCA B700                           mov     bh, 0           ; first primary partition (or active)
00001FCC E89A01                         call    sethard
00001FCF 7208                           jb      short hardfile_err
00001FD1 E86508                         call    dmax_check      ; error if already 26 drives
00001FD4 7303                           jnb     short hardfile_err
00001FD6 E87508                         call    xinstall_bds    ; insert new bds into linked list
                                hardfile_err:
00001FD9 5A                             pop     dx
                                        ;inc    dl              ; next hard drive
                                        ; 12/12/2023
00001FDA 42                             inc     dx
00001FDB FECE                           dec     dh
00001FDD 75E2                           jnz     short dohard1

                                ; end of physical drive   initialization

                                ; *** do not change the position of the following statement.
                                ; *** domini routine will use [drvmax] value for the start of the logical
                                ; *** drive number of mini disk(s).

00001FDF E8CA06                         call    domini          ; for setting up mini disks, if found

                                ; -- begin added section

00001FE2 8A36[5F1A]                     mov     dh, [hnum]      ; we already know this is >0
00001FE6 B280                           mov     dl, 80h
                                dohardx1:
00001FE8 B701                           mov     bh, 1           ; do all subsequent primary partitions
                                dohardx2:
00001FEA 52                             push    dx
00001FEB 53                             push    bx
00001FEC 8B3E[621A]                     mov     di, [end_of_bdss]
00001FF0 8A1E[7500]                     mov     bl, [drvmax]
00001FF4 E87201                         call    sethard
00001FF7 720E                           jb      short dohardx4 ; move to next hardfile if error
00001FF9 E83D08                         call    dmax_check      ; make sure <=26 drives
00001FFC 7309                           jnb     short dohardx4 ; skip if error
00001FFE E84D08                         call    xinstall_bds    ; insert new bds into linked list
00002001 5B                             pop     bx              ; get partition number
00002002 5A                             pop     dx              ; restore physical drive counts
00002003 FEC7                           inc     bh
00002005 EBE3                           jmp     short dohardx2 ; keep looping until we fail
                                ; ---------------------------------------------------------------------------

                                dohardx4:
00002007 5B                             pop     bx              ; unjunk partition number from stack
00002008 5A                             pop     dx              ; restore physical drive counts
                                        ;inc    dl              ; next hard drive
                                        ; 12/12/2023
00002009 42                             inc     dx
0000200A FECE                           dec     dh
0000200C 75DA                           jnz     short dohardx1

                                ; -- end changed section

                                ;********************************************************************************
                                ; if more than 2 diskette drives on the system, then it is necessary to remap
                                ; the bds chain to adjust the logical drive num (drive letter) with greater
                                ; than two diskette drives
                                ;
                                ; new scheme:      if more than 2 disktte 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.
                                ;********************************************************************************

0000200E 803E[2501]02                   cmp     byte [dsktnum], 2 ; >2 diskette drives
                                        ;jbe    short static_configure ; no - no need for remapping
00002013 7603                           jbe     short no_remap
00002015 E8D500                         call    remap           ; remap bds chain to adjust driver letters
                                no_remap:

                                ; End of drive initialization.

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

                                ;we now decide, based on the configurations available so far, what
                                ;code or data we need to keep as a stay resident code. the following table
                                ;shows the configurations under consideration. they are listed in the order
                                ;of their current position memory.
                                ;
                                ;configuration will be done in two ways:
                                ;
                                ;first, we are going to set "static configuration". static configuration will
                                ;consider from basic configuration to endof96tpi configuration. the result
                                ;of static configuration will be the address the dynamic configuration will
                                ;use to start with.
                                ;
                                ;secondly, "dynamic configuration" will be performed. dynamic configuration
                                ;involves possible relocation of code or data. dynamic configuration routine
                                ;will take care of bdsm tables and at rom fix module thru k09 suspend/resume
```

```
                                ;code individually. after these operation, [dosdatasg] will be set.
                                ;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:
00002018 8B3E[621A]                         mov     di, [end_of_bdss]
0000201C 81FF[4C08]                         cmp     di, bdss        ; 19/10/2022
                                            ;cmp    di, offset bdss         ; did we allocate any hard drive bdss?
00002020 750D                               jnz     short dynamic_configure       ; that's the end, then
                                            ; 18/10/2022
00002022 BF[4C08]                           mov     di, end96tpi
                                            ;mov    di, offset harddrv ; end96tpi
                                                                    ; keep everything up to end96tpi
00002025 803E[7700]00                       cmp     byte [fhave96], 0
0000202A 7503                               jnz     short dynamic_configure

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

                                            ; 10/12/2023
0000202F 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
00002030 BE00F0                             mov     si, 0F000h
00002033 8EC6                               mov     es, si          ; ES -> ROM BIOS segment

00002035 803E[AF05]FC                       cmp     byte [model_byte], 0FCh ; AT ?
                                            ;jnz    short checkcmosclock
                                            ; 10/12/2023
0000203A 751E                               jnz     short checkcompaqbug ; no
0000203C 803E[5F1A]00                       cmp     byte [hnum], 0 ; No hard file?
                                            ;jz     short checkcmosclock
00002041 7417                               jz      short checkcompaqbug
00002043 97                                 xchg    ax, di          ; save allocation pointer in ax
                                            ; 13/12/2023
                                            ;mov    si, 0F000h
                                            ;mov    es, si          ; ES -> ROM BIOS segment
00002044 BE[681A]                           mov     si, bios_date   ; "01/10/84"
00002047 BFF5FF                             mov     di, 0FFF5h      ; ROM BIOS string is at       F000:FFF5
0000204A B90900                             mov     cx, 9           ; bdate_l
                                                                    ; Only patch ROM for bios 01/10/84
0000204D F3A6                               repe cmpsb              ; check for date + zero       on end
0000204F 97                                 xchg    ax, di          ; restore allocation pointer

                                ; M015 -- begin changes

                                            ;jnz    short checkcmosclock
                                            ; 02/10/2022
00002050 7508                               jnz     short checkcompaqbug

                                ; install at rom fix

                                            ; 19/10/2022
                                            ;mov    cx, offset endatrom
00002052 B9[2018]                           mov     cx, endatrom
                                            ;mov    si, offset ibm_disk_io
00002055 BE[F216]                           mov     si, ibm_disk_io
00002058 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
0000205A 26813EEAFF434F                     cmp     word [es:0FFEAh], 'CO'
                                            ;cmp    word ptr es:0FFEAh, 'OC' ; look for COMPAQ
00002061 754B                               jnz     short not_compaq_patch
00002063 26813EECFF4D50                     cmp     word [es:0FFECh], 'MP'
                                            ;cmp    word ptr es:0FFECh, 'PM'
0000206A 7542                               jnz     short not_compaq_patch
0000206C 26813EEEFF4151                     cmp     word [es:0FFEEh], 'AQ'
                                            ;cmp    word ptr es:0FFEEh, 'QA'
00002073 7539                               jnz     short not_compaq_patch

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

00002075 26A1FBFF                           mov     ax, [es:0FFFBh]       ; get year
00002079 86C4                               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)

0000207B 3D3638                         cmp     ax, 3836h ; 02/10/2022 (NASM syntax)
                                        ;cmp    ax, '86'        ; 3836h
                                                                ; is it 86?
0000207E 772E                           ja      short not_compaq_patch
00002080 7218                           jb      short do_compaq_patch
00002082 26A1F5FF                       mov     ax, [es:0FFF5h]       ; get month
00002086 86C4                           xchg    ah, al
00002088 3D3830                         cmp     ax, 3038h ; 02/10/2022 (NASM syntax)
                                        ;cmp    ax, '08'        ; 3038h
                                                                ; is it 08?
0000208B 7721                           ja      short not_compaq_patch
0000208D 720B                           jb      short do_compaq_patch
0000208F 26A1F8FF                       mov     ax, [es:0FFF8h]       ; get day
00002093 86C4                           xchg    ah, al
00002095 3D3430                         cmp     ax, 3034h ; 02/10/2022 (NASM syntax)
                                        ;cmp    ax, '04'        ; 3034h
                                                                ; is it 04?
00002098 7314                           jnb     short not_compaq_patch

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

                                install_int13_patch:
000020A0 0E                             push    cs
000020A1 07                             pop     es
                                        ; 18/10/2022
000020A2 893E[B400]                     mov     [Orig13], di   ; set new rom bios int 13 vector
000020A6 8C0E[B600]                     mov     [Orig13+2], cs
000020AA 29F1                           sub     cx, si         ; size of rom fix module
000020AC F3A4                           rep movsb              ; relocate it

                                ; M065 -- end changes

                                ; ----------------------------------------------------------------------------
                                not_compaq_patch:               ; M065
                                        ; 17/10/2022
                                checkcmosclock:
                                        ; 18/10/2022
000020AE 0E                             push    cs
000020AF 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:
000020B0 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)

                                        ; ........

000020B1 B80041                         mov     ax, 4100h       ; wait for any external event (al=0)
000020B4 B304                           mov     bl, 4           ; wait for 4 clock ticks
000020B6 F9                             stc                     ; Assume we will fail
000020B7 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
                                        ; ........
000020B9 5F                             pop     di ; ?
000020BA 721B                           jc      short configdone ; 21/12/2022

000020BC C606[7900]01                   mov     byte [fhavek09], 1
                                                        ; remember we have a k09 type
000020C1 1E                             push    ds
000020C2 31C0                           xor     ax, ax
000020C4 8ED8                           mov     ds, ax

000020C6 893EB001                       mov     [6Ch*4], di
                                        ;mov    ds:1B0h, di     ; [6Ch*4]
                                                        ; new int 6Ch handler
                                        ;mov    word ptr ds:1B2h, cs ; [6Ch*4+2]
000020CA 8C0EB201                       mov     word [6Ch*4+2], cs
000020CE 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
000020CF BE[3E18]                       mov     si, int_6Ch
000020D2 B9BE01                         mov     cx, endk09-int_6Ch ; 461 in PCDOS 7.1 IBMBIO.COM
000020D5 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

000020D7 83C70F                         add     di, 15          ; round (up) to paragraph
                                        ; 10/12/2022
                                        ;shr    di, 1
                                        ;shr    di, 1
                                        ;shr    di, 1
                                        ;shr    di, 1
000020DA B104                           mov     cl, 4
000020DC D3EF                           shr     di, cl
                                        ; 10/12/2022
000020DE 83C770                         add     di, 70h ; KERNEL_SEGMENT (in fact: IO.SYS loading segment)
                                        ; 19/10/2022 - Temporary !
                                        ;db     81h, 0C7h, 70h, 0 ; add di, 0070h
000020E1 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
```

```
6954
6955                                    ; 22/12/2022
6956                                    ;;;     mov     bp, SETDRIVE
6957                                    ;;;     ;mov    bp, 4D7h        ; SetDrive (2C7h:47Dh = 70h:2A47h)
6958                                    ;;;     push    cs              ; simulate far call
6959                                    ;;;     call    call_bios_code ; get correct bds for this drive
6960
6961                                            ; 22/12/2022
6962                                            ;push   es              ; copy bds back to ds:di
6963                                            ;pop    ds
6964                                            ;pop    es              ; pop whatever was in es
6965
6966                                    ; Now we load in the MSDOS.SYS file
6967
6968                                    ; 22/12/2022
6969                                    ; -----
6970                                    ;       mov     bx, [di+6]      ; [di+BDS.BDS_BPB.BPB_BYTESPERSECTOR]
6971                                    ;       mov     [cs:md_sectorsize], bx; used by get_fat_sector proc.
6972                                    ;       mov     bl, [di+1Fh]    ; [di+BDS.fatsiz]
6973                                    ;                               ; get size of fat on media
6974                                    ;       ;mov    es:16DEh, bl
6975                                    ;       mov     [es:fbigfat], bl
6976                                    ;       mov     cl, [di+8]
6977                                    ;       mov     ax, [di+17h] ; [di+BDS.BDS_BPB.BPB_HIDDENSECTORS]
6978                                    ;       ;sub    es:16D8h, ax
6979                                    ;       sub     [es:bios_l], ax         ; subtract hidden sectors since we
6980                                    ;                               ; need a logical sector number that will
6981                                    ;                               ; be used by getclus(diskrd procedure)
6982                                    ;       mov     ax, [di+19h] ; [di+BDS.BDS_BPB.BPB_HIDDENSECTORS+2]
6983                                    ;       ;sbb    es:16DAh, ax
6984                                    ;       sbb     [es:bios_h], ax         ; subtract upper 16 bits of sector num
6985                                    ; -----
6986
6987                                            ; 11/12/2023 - Retro DOS v5.0 IO.SYS/IBMBIO.COM
6988                                    ; -----; 22/12/2022
6989                                            mov     bx, [es:di+6]  ; [di+BDS.BDS_BPB.BPB_BYTESPERSECTOR]
6990                                            mov     [md_sectorsize], bx ; used by get_fat_sector proc.
6991                                            ; 11/12/2023 ; *
6992                                            mov     bl, [es:di+3Bh]         ; [di+BDS.fatsiz] ; *
6993                                            ;mov    bl, [es:di+1Fh]         ; [di+BDS.fatsiz]
6994                                                                    ; get size of fat on media
6995                                            mov     [fbigfat], bl
6996                                            mov     cl, [es:di+8]
6997                                            mov     ax, [es:di+17h] ; [di+BDS.BDS_BPB.BPB_HIDDENSECTORS]
6998                                            sub     [First_Data_Sector], ax ; *
6999                                            ;sub    [bios_l], ax    ; subtract hidden sectors since we
7000                                                                    ; need a logical sector number that will
7001                                                                    ; be used by getclus(diskrd procedure)
7002                                            mov     ax, [es:di+19h] ; [di+BDS.BDS_BPB.BPB_HIDDENSECTORS+2]
7003                                            sbb     [First_Data_Sector+2], ax ; *
7004                                            ;sbb    [bios_h], ax    ; subtract upper 16 bits of sector num
7005                                    ; ------
7006
7007                                            xor     ch, ch  ; cx = sectors/cluster
7008
7009                                    ; the boot program has left the directory at 0:500h
7010
7011                                            ; 11/12/2023 - - Retro DOS v5.0 IBMBIO.COM/IO.SYS
7012                                            ;push   di
7013                                            push    ds
7014                                            ;xor    di, di
7015                                            ;mov    ds, di
7016                                            xor     bx, bx ; 0
7017                                            mov     ds, bx
7018                                            mov     bx, [53Ah]
7019                                            ;mov    bx, ds:53Ah             ; (First cluster of the 2nd dir entry
7020                                                                    ; of root directory in the buffer at 500h)
7021                                            pop     ds
7022                                            mov     si, [firstcluster_hw] ; 11/12/2023
7023                                                                    ; (32 bit cluster number for FAT32 fs)
7024                                            ;pop    ds
7025                                            ;pop    di
7026
7027                                            ; 12/12/2023
7028                                            ; (PCDOS 7.1 IBMBIO.COM - BIOSDATA:2397h)
7029                                            ; ...........
7030                                            ; ds = cs
7031                                            mov     al, [fbigfat]
7032                                            push    ax              ; (*) save fbigfat flags
7033                                            mov     al, [drvfat]
7034                                            or      al, [Boot_Drv]
7035                                            jnz     short boot_drv_fixed ; hard disk
7036                                    boot_drv_removable:                     ; calculate cluster count and set fbig or fbigbig flag
7037                                            push    bx              ; for removable drives
7038                                            push    cx
7039                                            ; 28/12/2023
7040                                            ;push   dx ; (not necessary)
7041
7042                                            ; 12/12/2023
7043                                            push    es
7044                                            pop     ds
7045
7046                                            mov     ax, [di+0Eh]    ; [di+BDS.totalsecs16]
7047                                            xor     dx, dx
7048                                            or      ax, ax
7049                                            jnz     short prep_totalsecs_ok
7050                                            mov     ax, [di+1Bh]    ; [di+BDS.totalsecs32]
7051                                            mov     dx, [di+1Dh]
7052                                    prep_totalsecs_ok:
7053                                            sub     ax, [di+9]      ; [di+BDS.resectors]
7054                                            sbb     dx, 0
7055                                            push    ax
7056                                            push    dx
7057                                            mov     bx, [di+11h]    ; [di+BDS.fatsecs16]
7058                                            xor     ax, ax
7059                                            or      bx, bx
7060                                            jnz     short prep_fatsecs_ok
7061                                            mov     bx, [di+1Fh]    ; [di+BDS.fatsecs32]
7062                                            mov     ax, [di+21h]
7063                                    prep_fatsecs_ok:
7064                                            mov     cl, [di+0Bh]    ; ax:bx = 32 bit count of FAT sectors
7065                                                                    ; [di+BDS.fats]
7066                                            xor     ch, ch
7067                                            mul     cx
7068                                            xchg    ax, cx
7069                                            mul     bx
7070                                            add     cx, dx
7071                                            mov     bx, ax          ; cx:bx = total (2*) fat sectors
7072                                            pop     dx
7073                                            pop     ax              ; dx:ax = totals sectors - reserved sectors
7074                                            sub     ax, bx
7075                                            sbb     dx, cx          ; dx:ax = data sectors (includes root dir sectors)
7076                                            mov     bx, [di+0Ch]    ; [di+BDS.direntries]
7077                                            add     bx, 15          ; 16 directory entries per sector
```

```
7078                                                    ; (round up sector count by adding 15)
7079                            mov     cl, 4           ; (rounded) dir entries / 16
7080                            shr     bx, cl
7081                            xor     cx, cx
7082                            sub     ax, bx
7083                            sbb     dx, cx          ; dx:ax = data sectors (except root directory sectors)
7084                                                    ; (will be used for cluster count calculation)
7085                            mov     cl, [di+8]      ; [di+BDS.secperclus]
7086
7087                            ; 12/12/2023
7088                            push    cs
7089                            pop     ds
7090
7091                            push    ax              ; 32 bit division (data sectors / sector per cluster)
7092                            mov     ax, dx
7093                            xor     dx, dx
7094                            div     cx
7095                            mov     bx, ax
7096                            pop     ax
7097                            div     cx
7098                            or      bx, bx          ; 32 bit cluster count if bx > 0
7099                            jnz     short set_fbigbig_flag ; too big cluster number
7100                            cmp     ax, 0FFF6h
7101                            jb      short set_fbig_flag
7102                    set_fbigbig_flag:
7103                            or      byte [fbigfat], 20h ; FAT32 ; fbigbig
7104                            jmp     short set_fbig_flag_ok
7105                    ; ---------------------------------------------------------------------------
7106
7107                    set_fbig_flag:
7108                            cmp     ax, 0FF6h       ; 4096-10
7109                                                    ; is this 16-bit fat?
7110                            jb      short set_fbig_flag_ok ; no, small fat
7111                            or      byte [fbigfat], 40h ; FAT16 ; fbig
7112                    set_fbig_flag_ok:
7113                            ; 28/12/2023
7114                            ;pop    dx
7115                            pop     cx
7116                            pop     bx
7117                    boot_drv_fixed:
7118                            xor     di, di
7119
7120                            ; cx = sectors/cluster
7121                            ; si:bx = first cluster
7122                            ; di = 0
7123
7124                            ; ...........
7125                    loadit:
7126                            mov     ax, SYSINITSEG ; 46Dh
7127                            ;mov    ax, 544h        ; 11/12/2023 - PCDOS 7.1 IBMBIO.COM
7128                            ;;mov   ax, 46Dh        ; sysinit segment
7129                            mov     es, ax
7130                            mov     es, [es:CURRENTDOSLOCATION] ; 09/12/2022
7131                            ;mov    es, [es:271h]
7132
7133                            call    getclus         ; read cluster at ES:DI (DI is updated)
7134
7135                    ; ---------------------------------------------------------------------------
7136
7137                            ; 13/12/2023 - Retro DOS v5.0 (PCDOS 7.1) IBMBIO.COM
7138                            ; (PCDOS 7.1 IBMBIO.COM - BIOSDATA:2431h)
7139
7140                            ;test   byte [cs:fbigfat], 20h ; fbigbig ; FAT32 fs flag
7141                            test    byte [fbigfat], 20h ; fbigbig ; FAT32 fs flag
7142                            jz      short iseof
7143
7144                    eofbigbig:  ; si:bx = 32 bit cluster number
7145                            cmp     si, 0FFFh
7146                            jnz     short iseofx
7147                            cmp     bx, 0FFF7h
7148                            jmp     short iseofx
7149
7150                    ; ---------------------------------------------------------------------------
7151                            ; 13/12/2023
7152                    iseof:
7153                            ;;test  byte [cs:fbigfat], fbig
7154                            ;test   byte [cs:fbigfat], 40h ; fbig
7155                            ; 12/12/2023
7156                            ; ds = cs
7157                            test    byte [fbigfat], 40h ; fbig
7158                            jnz     short eofbig
7159                            cmp     bx, 0FF7h
7160                            jmp     short iseofx
7161                    ; ---------------------------------------------------------------------------
7162
7163                    eofbig:
7164                            cmp     bx, 0FFF7h
7165                    iseofx:
7166                            jb      short loadit    ; keep loading until cluster = eof
7167
7168                    ; ---------------------------------------------------------------------------
7169
7170                            ; 19/04/2024
7171                            ; 28/12/2023
7172                            pop     ax              ; (*) restore fbigfat flags
7173                                                    ; (after loading DOS kernel)
7174                            ; 06/04/2024
7175                            ;mov    [cs:fbigfat], al
7176                            mov     [fbigfat], al
7177                    %endif
7178                    ; ---------------------------------------------------------------------------
7179                    ; ---------------------------------------------------------------------------
7180
7181                            ; 19/04/2024
7182
7183 000020E5 E8FE04              call    setdrvparms
7184
7185                            ;;;jmp  far ptr 46Dh:267h ; jmp       SYSINIT_SEG:SYSINIT_START
7186                            ;;jmp   far 46Dh:267h
7187                            ; 12/12/2023
7188                            ;jmp    far 544h:269h  ; (PCDOS 7.1 IBMBIO.COM)
7189
7190 000020E8 EA[6902]D904     jmp     SYSINITSEG:SYSINITSTART
7191
7192                    ; =============== S U B   R O U T I N E =======================================
7193
7194                    ; Following are subroutines to support resident device driver initialization
7195                    ;
7196                    ;M011 -- note:  deleted setup_bdsms and reset_bdsms here
7197
7198                    ;   M035 -- begin changed section
7199
7200                    ;****************************************************************************
7201                    ; module name: remap
```

```
                                ;
                                ; 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
                                          ;mov   di, [cs:start_bds] ; get first bds
000020ED 8B3E[1901]                       mov   di, [start_bds]
                                ; search for 1st fixed disk physical drive num
                                drive_loop:
000020F1 807D0480                         cmp   byte [di+4], 80h ; [di+BDS.drivenum]
                                                                 ; first hard disk??
000020F5 7409                             jz    short fdrv_found ; yes,      continue
000020F7 8B3D                             mov   di, [di]         ; [di+BDS.link]
                                                                 ; get next bds,       assume segment
000020F9 83FFFF                           cmp   di, -1 ; 0FFFFh        ; last bds?
000020FC 75F3                             jnz   short drive_loop ; loop      if not
000020FE 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:
00002100 B002                             mov   al, 2            ; start with logical drv num=2
                                fdrv_loop:
00002102 884505                           mov   [di+5], al       ; [di+BDS.drivelet]
00002105 8B3D                             mov   di, [di]         ; [di+BDS.link]
                                                                 ; ds:di--> next       bds
                                          ;inc   al              ; set num for next drive
                                          ; 18/12/2022
00002107 40                               inc   ax
00002108 83FFFF                           cmp   di, 0FFFFh       ; last hard drive ?
0000210B 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
0000210D 8B3E[1901]                       mov   di, [start_bds]        ; get first bds
00002111 8B3D                             mov   di, [di]         ; [di+BDS.link]
                                                                 ; ds:di-->bds2
                                          ;mov   ah, [cs:dsktnum] ; get number of floppies to remap
00002113 8A26[2501]                       mov   ah, [dsktnum]
00002117 80EC02                           sub   ah, 2            ; adjust for a:       & b:
                                remap_loop1:
0000211A 8B3D                             mov   di, [di]         ; [di+BDS.link]
                                                                 ; set new num to next floppy
0000211C 884505                           mov   [di+5], al       ; [di+BDS.drivelet]
0000211F FEC0                             inc   al               ; new number for next floppy
00002121 FECC                             dec   ah               ; count down extra floppies
00002123 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]
00002125 A0[FC19]                         mov   al, [drvfat]
00002128 3C02                             cmp   al, 2            ; is it a: or b: ?
0000212A 721D                             jb    short rmap_exit
                                          ;sub   al, [cs:dsktnum]
0000212C 2A06[2501]                       sub   al, [dsktnum]  ; is it one of the other floppies?
00002130 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
00002132 0402                             add   al, 2            ; bootdrv -= (dsktnum-2)
00002134 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
00002136 0206[7500]                       add   al, [drvmax]
                                remap_change_boot_drv:
                                          ;mov   [cs:drvfat], al ; alter msdos.sys load drive
0000213A A2[FC19]                         mov   [drvfat], al
0000213D FEC0                             inc   al
0000213F 1E                               push  ds
00002140 BFD904                           mov   di, SYSINITSEG ; 46Dh
                                          ;mov   di, 544h        ; PCDOS 7.1 IBMBIO.COM
```

```
7326                                            ;;mov   di, 46Dh       ; SYSINIT segment
7327 00002143 8EDF                              mov     ds, di
7328 00002145 A2[9802]                          mov     [DEFAULTDRIVE], al
7329                                            ;mov    ds:296h, al    ; [SYSINIT+DEFAULT_DRIVE]
7330                                                                    ; pass it to sysinit as      well
7331 00002148 1F                                pop     ds ; ds = cs
7332                                    rmap_exit:
7333 00002149 C3                                retn
7334
7335                                    ; ============== S U B   R O U TI N E =======================================
7336
7337                                    ; 17/10/2022
7338                                    ; 02/10/2022 - Retro DOS v4.0 (MSDOS 5.0 -actual-)
7339                                    ; 28/12/2018 - Retro DOS v4.0 (MSDOS 6.21 -draft-)
7340                                    ; 02/06/2018 - Retro DOS v3.0 (MSDOS 3.3)
7341                                    ; 19/03/2018 - Retro DOS v2.0 (MSDOS 2.11)
7342                                    ;************************************************
7343                                    ; getboot - get the boot sector for a hard disk
7344                                    ;
7345                                    ; Reads the boot sector from a specified drive into
7346                                    ; a buffer at the top of memory.
7347                                    ;
7348                                    ; dl = int13 drive number to read boot sector for
7349                                    ;************************************************
7350
7351                                    ; 17/10/2022
7352                                    bootbias equ 200h
7353
7354                                    getboot:    ; proc near
7355
7356                                                ; 15/12/2023 - Retro DOS v5.0
7357                                                ;       (Modified PCDOS 7.1) IBMBIO.COM/IO.SYS
7358                                                ; ds = cs
7359
7360                                                ; 08/04/2018
7361                                                ; Retro DOS v2.0 (IBMBIO.COM, IBMDOS 2.1)
7362                                                ; 28/03/2018 - MSDOS 6.0 - MSINIT.ASM, 1991
7363                                                ; 02/10/2022 - Retro DOS v4.0
7364                                                ;            (disassembled IO.SYS code of MSDOS 5.0)
7365
7366                                                ;mov    ax, [cs:init_bootseg] ; 17/10/2022
7367                                                ; 15/12/2023
7368 0000214A A1[061A]                          mov     ax, [init_bootseg]
7369 0000214D 8EC0                              mov     es, ax
7370
7371                                                ; 17/10/2022
7372 0000214F BB0002                            mov     bx, bootbias ; 200h
7373                                            ;mov    bx, 200h       ; bootbias
7374                                                                    ; load BX, ES:BX is where sector goes
7375 00002152 B80102                            mov     ax, 201h
7376 00002155 30F6                              xor     dh, dh
7377 00002157 B90100                            mov     cx, 1
7378 0000215A CD13                              int     13h             ; DISK - READ SECTORS INTO MEMORY
7379                                                                    ; AL = number of sectors to read, CH = track, CL = sector
7380                                                                    ; DH = head, DL       = drive, ES:BX -> buffer to fill
7381                                                                    ; Return: CF set on error, AH =      status, AL = number of sectors read
7382 0000215C 7209                              jc      short erret
7383                                            ; 17/10/2022
7384 0000215E 26813EFE0355AA                    cmp     word [es:bootbias+1FEh], 0AA55h
7385                                            ;cmp    word ptr es:3FEh, 0AA55h ; [es:bootbias+1FEh]
7386                                                                    ; Dave Litton magic word?
7387 00002165 7401                              jz      short norm_ret ; yes
7388                                    erret:
7389 00002167 F9                                stc
7390                                    norm_ret:
7391 00002168 C3                                retn
7392
7393                                    ; ============== S U B   R O U TI N E =======================================
7394
7395                                    ; 28/12/2018 - Retro DOS v4.0
7396
7397                                    ;****************************************************************************
7398                                    ;   sethard - generate bpb for a variable sized hard file. ibm has a
7399                                    ;   partitioned hard file; we must read physical sector 0 to determine where
7400                                    ;   our own logical sectors start. we also read in our boot sector to
7401                                    ;   determine version number
7402                                    ;
7403                                    ;   inputs: dl is rom drive number (80...)
7404                                    ;           bh is partition number (0....)
7405                                    ;           ds:di points to bds
7406                                    ;   outputs:       carry clear -> bpb is filled in
7407                                    ;           carry set   -> bpb is left uninitialized due to error
7408                                    ;   trashes (at least) si, cx
7409                                    ;   MUST PRESERVE ES:!!!!
7410                                    ;****************************************************************************
7411
7412                                                ; 16/12/2023 - Retro DOS v5.0 (PCDOS 7.1) IBMBIO.COM
7413                                                ; (PCDOS 7.1 IBMBIO.COM - BIOSDATA:24E9h)
7414
7415                                    sethard:    ; proc near
7416                                                ; 12/08/2023
7417                                                ; ds = cs = BIOSDATA
7418 00002169 57                                push    di
7419 0000216A 53                                push    bx
7420                                            ;push   ds  ; ds = cs = BIOSDATA ; 12/08/2023
7421 0000216B 06                                push    es
7422 0000216C 885D05                            mov     [di+5], bl      ; [di+BDS.drivelet]
7423 0000216F 885504                            mov     [di+4], dl      ; [di+BDS.drivenum]
7424                                            ; 16/12/2023
7425 00002172 804D3F01                          or      byte [di+3Fh], 1 ; PCDOS 7.1
7426                                            ;or     byte [di+23h], 1 ; [di+BDS.flags]
7427                                                                    ; fnon_removable
7428 00002176 C6453E05                          mov     byte [di+3Eh], 5 ; PCDOS 7.1
7429                                            ;mov    byte [di+22h], 5 ; [di+BDS.formfactor]
7430                                                                    ; ffHardFile
7431 0000217A C606[081A]00                      mov     byte [fbigfat], 0 ; assume 12 bit FAT
7432 0000217F 88FE                              mov     dh, bh          ; partition number
7433 00002181 52                                push    dx
7434 00002182 B408                              mov     ah, 8
7435 00002184 CD13                              int     13h             ; DISK - DISK -       GET CURRENT DRIVE PARAMETERS (XT,AT,XT286,CONV,PS)
7436                                                                    ; DL = drive number
7437                                                                    ; Return: CF set on error, AH =      status code, BL       = drive type
7438                                                                    ; DL = number of consecutive drives
7439                                                                    ; DH = maximum value for head number, ES:DI -> drive parameter
7440                                            ;inc    dh
7441                                            ; 16/12/2023 - Retro DOS v5.0
7442 00002186 88F2                              mov     dl, dh
7443 00002188 B600                              mov     dh, 0
7444 0000218A 42                                inc     dx
7445                                            ;mov    [di+15h], dh   ; [di+BDS.heads] ; get number of heads
7446 0000218B 895515                            mov     [di+15h], dx
7447 0000218E 5A                                pop     dx
7448 0000218F 7253                              jc      short setret   ; error if no hard disk
7449                                            ; 16/12/2023
```

```
7450                                            ;jc     short setret_j
7451
7452 00002191 80E13F                           and     cl, 3Fh
7453 00002194 884D13                           mov     [di+13h], cl   ; [di+BDS.secpertrack]
7454 00002197 52                               push    dx             ; save partition number
7455 00002198 E8AFFF                           call    getboot
7456 0000219B 5A                               pop     dx             ; restore partition number
7457 0000219C 7246                             jc      short setret
7458                                           ; 16/12/2023
7459                                           ;jnc    short chk_act_part
7460                                    ;setret_j:
7461                                           ;jmp    setret
7462
7463                                           ; 16/12/2023 - Retro DOS v5.0 (PCDOS 7.1 IBMBIO.COM)
7464                                    chk_act_part:
7465 0000219E 31DB                             xor     bx, bx ; 0
7466                                           ;;mov   [cs:ep_start_sector], bx
7467                                           ;;mov   [cs:ep_start_sector+2], bx
7468                                           ;mov    [cs:ep_hidden_secs], bx
7469                                           ;mov    [cs:ep_hidden_secs+2], bx
7470                                           ; 16/12/2023
7471                                           ; ds = cs
7472                                           ; 20/12/2023
7473                                           ;mov    [ep_start_sector], bx
7474                                           ;mov    [ep_start_sector+2], bx
7475 000021A0 891E[0622]                       mov     [ep_hidden_secs], bx
7476 000021A4 891E[0822]                       mov     [ep_hidden_secs+2], bx
7477
7478 000021A8 BBC203                           mov     bx, 3C2h       ; 1C2h+bootbias
7479
7480                                    ; The first 'active' partition is 00, the second is 01....
7481                                    ;   then the remainder of the 'primary' but non-active partitions
7482
7483                                    act_part:
7484 000021AB 26F647FC80                       test    byte [es:bx-4], 80h ; is the partition active?
7485 000021B0 740B                             jz      short no_act   ; no
7486                                    ; 16/12/2023
7487                                    %if 0
7488                                           ; 16/12/2023
7489                                           ; reject if partitiontype != 1, 4, 6, 0Bh, 0Ch, 0Eh
7490                                           cmp     byte [es:bx], 1 ; FAT12
7491                                           jz      short got_good_act
7492                                           cmp     byte [es:bx], 4        ; FAT16 CHS (<= 32MB)
7493                                           jz      short got_good_act
7494
7495                                           ; 16/12/2023 - Retro DOS v5.0 (PCDOS 7.1 IBMBIO.COM)
7496                                           cmp     byte [es:bx], 0Bh ; FAT32 CHS
7497                                           jz      short got_good_act
7498                                           cmp     byte [es:bx], 0Ch ; FAT32 LBA
7499                                           jz      short got_good_act
7500                                           cmp     byte [es:bx], 0Eh ; FAT16 LBA
7501                                           jz      short got_good_act
7502
7503                                           cmp     byte [es:bx], 6        ; FAT16 BIG CHS (> 32MB)
7504                                           jnz     short no_act
7505                                    ;%else
7506                                           ; 16/12/2023
7507                                           mov     al, [es:bx]     ; partition type
7508
7509                                           ; reject if partitiontype != 1, 4, 6, 0Bh, 0Ch, 0Eh
7510                                           cmp     al, 1           ; FAT12
7511                                           je      short got_good_act
7512                                           cmp     al, 4           ; FAT16 CHS (<= 32MB)
7513                                           je      short got_good_act
7514
7515                                           ; 16/12/2023 - Retro DOS v5.0 (PCDOS 7.1 IBMBIO.COM)
7516                                           cmp     al, 0Bh         ; FAT32 CHS
7517                                           je      short got_good_act
7518                                           cmp     al, 0Ch         ; FAT32 LBA
7519                                           je      short got_good_act
7520                                           cmp     al, 0Eh         ; FAT16 LBA
7521                                           je      short got_good_act
7522
7523                                           cmp     al, 6           ; FAT16 BIG CHS (> 32MB)
7524                                           jne     short no_act
7525                                    %endif
7526                                           ; 16/12/2023 - Retro DOS v5.0 (Modified PCDOS 7.1)
7527                                           ; check if it is a primary dos partition
7528
7529 000021B2 E83300                           call    chk_partition_type
7530 000021B5 7506                             jne     short no_act
7531
7532                                    got_good_act:                          ; 11/08/2023
7533 000021B7 08F6                             or      dh, dh          ; is this our target partition #?
7534                                                                   ; (0 = first primary dos or active partition)
7535 000021B9 744F                             jz      short set2      ; WE GOT THE ONE WANTED!!
7536 000021BB FECE                             dec     dh              ; count down
7537                                    no_act:
7538 000021BD 83C310                           add     bx, 16
7539 000021C0 81FB0204                         cmp     bx, 402h        ; 202h+bootbias
7540                                                                   ; last entry done?
7541 000021C4 75E5                             jnz     short act_part ; no, process next entry
7542
7543 000021C6 BBC203                           mov     bx, 3C2h        ; 1C2h+bootbias
7544                                                                   ; restore original value of bx
7545
7546                                    ; Now scan the non-active partitions
7547
7548                                    get_primary:
7549 000021C9 26F647FC80                       test    byte [es:bx-4], 80h
7550 000021CE 750B                             jnz     short not_prim ; we've already scanned
7551                                                                   ; the ACTIVE ones
7552                                    ; 16/12/2023
7553                                    %if 0
7554                                           ; 16/12/2023
7555                                           ; see if partitiontype == 1, 4, 6, 0Bh, 0Ch, 0Eh
7556                                           cmp     byte [es:bx], 1        ; FAT12
7557                                           jz      short got_prim
7558                                           cmp     byte [es:bx], 4        ; FAT16 CHS (<= 32MB)
7559                                           jz      short got_prim
7560
7561                                           ; 16/12/2023 - Retro DOS v5.0 (PCDOS 7.1 IBMBIO.COM)
7562                                           cmp     byte [es:bx], 0Bh ; FAT32 CHS
7563                                           jz      short got_prim
7564                                           cmp     byte [es:bx], 0Ch ; FAT32 LBA
7565                                           jz      short got_prim
7566                                           cmp     byte [es:bx], 0Eh ; FAT16 LBA
7567                                           jz      short got_prim
7568
7569                                           cmp     byte [es:bx], 6        ; FAT16 BIG CHS (> 32MB)
7570                                           jnz     short not_prim
7571                                    ;%else
7572                                           ; 16/12/2023
7573                                           mov     al, [es:bx]     ; partition type
```

```
7574
7575                                    ; see if partitiontype == 1, 4, 6, 0Bh, 0Ch, 0Eh
7576                                    cmp     al, 1           ; FAT12
7577                                    je      short got_prim
7578                                    cmp     al, 4           ; FAT16 CHS (<= 32MB)
7579                                    je      short got_prim
7580
7581                                    ; 16/12/2023 - Retro DOS v5.0 (PCDOS 7.1 IBMBIO.COM)
7582                                    cmp     al, 0Bh         ; FAT32 CHS
7583                                    je      short got_prim
7584                                    cmp     al, 0Ch         ; FAT32 LBA
7585                                    je      short got_prim
7586                                    cmp     al, 0Eh         ; FAT16 LBA
7587                                    je      short got_prim
7588
7589                                    cmp     al, 6           ; FAT16 BIG CHS (> 32MB)
7590                                    jne     short not_prim
7591                                %endif
7592                                    ; 16/12/2023 - Retro DOS v5.0 (Modified PCDOS 7.1)
7593                                    ; check if it is a primary dos partition
7594
7595 000021D0 E81500                   call    chk_partition_type
7596 000021D3 7506                     jne     short not_prim
7597
7598                                got_prim:
7599 000021D5 08F6                     or      dh, dh          ; is this our target partition?
7600 000021D7 7431                     jz      short set2
7601 000021D9 FECE                     dec     dh
7602                                not_prim:
7603 000021DB 83C310                   add     bx, 16
7604 000021DE 81FB0204                 cmp     bx, 402h        ; 202h+bootbias
7605 000021E2 75E5                     jnz     short get_primary ; loop till we've gone through table
7606                                setret:
7607 000021E4 F9                       stc                     ; error return
7608 000021E5 E9C703                   jmp     ret_hard_err
7609
7610                                ; ---------------------------------------------------------------------------
7611                                    ; 16/12/2023 - Retro DOS v5.0 (Modified PCDOS 7.1)
7612
7613
7614                                chk_partition_type:
7615                                    ; 16/12/2023
7616 000021E8 268A07                   mov     al, [es:bx]     ; partition type
7617
7618                                    ; see if partitiontype == 1, 4, 6, 0Bh, 0Ch, 0Eh
7619 000021EB 3C01                     cmp     al, 1           ; FAT12
7620 000021ED 7412                     je      short chk_ptype_retn
7621 000021EF 3C04                     cmp     al, 4           ; FAT16 CHS (<= 32MB)
7622 000021F1 740E                     je      short chk_ptype_retn
7623
7624                                    ; 16/12/2023 - Retro DOS v5.0 (PCDOS 7.1 IBMBIO.COM)
7625 000021F3 3C0B                     cmp     al, 0Bh         ; FAT32 CHS
7626 000021F5 740A                     je      short chk_ptype_retn
7627 000021F7 3C0C                     cmp     al, 0Ch         ; FAT32 LBA
7628 000021F9 7406                     je      short chk_ptype_retn
7629 000021FB 3C0E                     cmp     al, 0Eh         ; FAT16 LBA
7630 000021FD 7402                     je      short chk_ptype_retn
7631
7632 000021FF 3C06                     cmp     al, 6           ; FAT16 BIG CHS (> 32MB)
7633                                chk_ptype_retn:
7634                                    ; zf = 1 -> primary DOS partition
7635                                    ; zf = 0 -> not a primary DOS partition
7636 00002201 C3                       retn
7637
7638                                ; ---------------------------------------------------------------------------
7639
7640                                    ; 16/12/2023 - Retro DOS v5.0 (Modified PCDOS 7.1)
7641                                    ; (PCDOS 7.1 IBMBIO.COM - BIOSDATA:25B6h)
7642                                ep_start_sector:
7643 00002202 00000000                 dd 0
7644 00002206 00000000             ep_hidden_secs:    dd 0
7645
7646                                ; ---------------------------------------------------------------------------
7647
7648                                ;  until we get the real logical boot record and get the bpb,
7649                                ;  BDS_BPB.BPB_BIGTOTALSECTORS will be used instead of BDS_BPB.BPB_TOTALSECTORS
7650                                ;  for the convenience of the computation.
7651                                ;
7652                                ;  at the end of this procedure, if a bpb information is gotten from
7653                                ;  the valid boot record, then we are going to use those bpb information
7654                                ;  without change.
7655                                ;
7656                                ;  otherwise, if (hidden sectors + total sectors) <= a word, then we will move
7657                                ;  BDS_BPB.BPB_BIGTOTALSECTORS (low) to BDS_BPB.BPB_TOTALSECTORS and zero out
7658                                ;  BDS_BPB.BPB_BIGTOTALSECTORS entry to make it a conventional bpb format.
7659
7660                                set2:
7661                                    ; 12/08/2023
7662                                    ; ds = cs = BIOSDATA segment (0070h)
7663 0000220A 8816[091A]               mov     [rom_drv_num], dl
7664                                    ;mov    [cs:rom_drv_num], dl
7665                                    ; save the rom bios drive number we are handling now.
7666 0000220E 268B4704                 mov     ax, [es:bx+4]  ; hidden sectors (start      sector)
7667 00002212 268B5706                 mov     dx, [es:bx+6]
7668
7669                                ; decrement the sector count by 1 to make it zero based. exactly 64k
7670                                ; sectors should be allowed
7671
7672 00002216 83E801                   sub     ax, 1
7673 00002219 83DA00                   sbb     dx, 0
7674 0000221C 26034708                 add     ax, [es:bx+8]  ; sectors in partition
7675 00002220 2613570A                 adc     dx, [es:bx+10]
7676 00002224 7305                     jnc     short okdrive
7677 00002226 800E[081A]80             or      byte [fbigfat], 80h ; ftoobig
7678
7679                                    ; 16/12/2023 - Retro DOS v5.0 (PCDOS 7.1) IBMBIO.COM
7680                                    ;;;
7681                                okdrive:
7682                                    ;add    ax, [cs:ep_hidden_secs]
7683                                    ;adc    dx, [cs:ep_hidden_secs+2]
7684                                    ; ds = cs
7685 0000222B 0306[0622]               add     ax, [ep_hidden_secs]
7686 0000222F 1316[0822]               adc     dx, [ep_hidden_secs+2]
7687 00002233 7305                     jnc     short okdrive_1
7688 00002235 800E[081A]80             or      byte [fbigfat], 80h ; ftoobig
7689                                okdrive_1:
7690 0000223A 26803F0C                 cmp     byte [es:bx], 0Ch ; FAT32 LBA partition ID
7691 0000223E 7418                     je      short set_lba_flag
7692 00002240 26803F0E                 cmp     byte [es:bx], 0Eh ; FAT16 LBA partition ID
7693 00002244 7412                     je      short set_lba_flag
7694 00002246 3B5513                   cmp     dx, [di+13h]   ; if dx > [di+BDS.secpertrack] then
7695 00002249 730D                     jnb     short set_lba_flag ; set LBA r/w flag
7696 0000224B F77513                   div     word [di+13h]
7697 0000224E 31D2                     xor     dx, dx
```

```
7698 00002250 F77515                            div     word [di+15h]
7699 00002253 3D0004                            cmp     ax, 400h        ; if ax (cylinder number) >= 1024
7700                                                                    ;  set LBA r/w flag
7701 00002256 7204                              jb      short set3
7702                                    set_lba_flag:
7703 00002258 804D4004                              or byte [di+40h], 4 ; fLBArw ; LBA r/w flag
7704                                            ;;;
7705                                    ;okdrive:
7706                                            ; 16/12/2023
7707                                    set3:
7708                                            ;mov    ax, [es:bx+4]
7709                                            ;mov    [di+17h], ax    ; [di+BDS.hiddensecs]
7710                                            ;                       ; BPB_HIDDENSECTORS = p->partitionbegin
7711                                            ;mov    ax, [es:bx+6]
7712                                            ;mov    [di+19h], ax    ; [di+BDS.hiddensecs+2]
7713
7714                                            ; 16/12/2023 - Retro DOS v5.0 (PCDOS 7.1) IBMBIO.COM
7715                                            ;;;
7716 0000225C 268B4704                          mov     ax, [es:bx+4]  ; start sector (LBA) of the partition
7717 00002260 268B5706                          mov     dx, [es:bx+6]
7718                                            ;add    ax, [cs:ep_hidden_secs]
7719                                            ;adc    dx, [cs:ep_hidden_secs+2]
7720                                            ; ds = cs
7721 00002264 0306[0622]                        add     ax, [ep_hidden_secs]
7722                                                                    ; + hidden secs of the extd dos partion
7723 00002268 1316[0822]                        adc     dx, [ep_hidden_secs+2]
7724 0000226C 894517                            mov     [di+17h], ax    ; [di+BDS.hiddensecs]
7725 0000226F 895519                            mov     [di+19h], dx    ; [di+BDS.hiddensecs+2]
7726 00002272 31C0                              xor     ax, ax ; 0
7727 00002274 89457B                            mov     [di+7Bh], ax    ; [di+BDS.bdsm_hidden_trks]
7728 00002277 89450E                            mov     [di+0Eh], ax    ; [di+BDS.totalsec16]
7729                                            ;;;
7730
7731 0000227A 268B570A                          mov     dx, [es:bx+10] ; # of sectors (high)
7732 0000227E 268B4708                          mov     ax, [es:bx+8]  ; # of sectors (low)
7733 00002282 89551D                            mov     [di+1Dh], dx    ; [di+BDS.totalsecs32+2]
7734 00002285 89451B                            mov     [di+1Bh], ax    ; [di+BDS.totalsecs32]
7735                                                                    ; bpb->maxsec =       p->partitionlength
7736                                            ;cmp    dx, 0
7737                                            ;ja     short okdrive_1
7738                                            ; 16/12/2023
7739 00002288 09D2                              or      dx, dx
7740 0000228A 7505                              jnz     short set3_read
7741 0000228C 83F840                            cmp     ax, 64          ; if (p->partitionlength < 64)
7742                                            ;jb     short setret    ; return -1;
7743 0000228F 7264                              jb      short set3_err
7744                                    ;okdrive_1:
7745                                            ; 16/12/2023
7746                                    set3_read:
7747 00002291 8B5519                            mov     dx, [di+19h]    ; [di+BDS.hiddensecs+2]
7748 00002294 8B4517                            mov     ax, [di+17h]    ; [di+BDS.hiddensecs]
7749 00002297 31DB                              xor     bx, bx          ; boot sector number - for mini      disk
7750                                                                    ; usually equal       to the # of sec/trk.
7751 00002299 8A5D13                            mov     bl, [di+13h]    ; [di+BDS.secpertrack]
7752 0000229C 50                                push    ax
7753 0000229D 89D0                              mov     ax, dx
7754 0000229F 31D2                              xor     dx, dx
7755 000022A1 F7F3                              div     bx              ; (sectors)dx:ax / (BDS.secpertrack)bx =
7756                                                                    ; (track)temp_h:ax + (sector)dx
7757                                    ; 16/12/2023
7758                                    %if 0
7759                                            ; 17/10/2022
7760                                            ;mov    [cs:temp_h], ax
7761                                            ; 12/08/2023 (ds=cs)
7762                                            mov     [temp_h], ax
7763                                            pop     ax
7764                                            div     bx
7765                                            mov     cl, dl
7766                                            inc     cl
7767                                            xor     bx, bx
7768                                            mov     bl, [di+15h]    ; [di+BDS.heads]
7769                                            push    ax
7770                                            xor     dx, dx
7771                                            ;mov    ax, [cs:temp_h]
7772                                            mov     ax, [temp_h] ; 12/08/2023
7773                                            div     bx
7774                                            ;mov    [cs:temp_h], ax
7775                                            mov     [temp_h], ax ; 12/08/2023
7776                                            pop     ax
7777                                            div     bx              ; dl is head, ax is cylinder
7778                                            ; 12/08/2023 (ds=cs)
7779                                            cmp     word [temp_h], 0
7780                                            ;cmp    word [cs:temp_h], 0
7781                                            ja      short setret_brdg ; exceeds the      limit of int 13h
7782                                            cmp     ax, 1024
7783                                            ja      short setret_brdg ; exceeds the      limit of int 13h
7784                                                    ; Retro DOS v3.2 note by Erdogan Tan - 28/07/2019
7785                                                    ; **MSDOS code accepts if ax = 1024 but it is nonsense here
7786                                                    ; ('ja' must be 'jnb')
7787                                    okdrive_2:
7788                                            ; 28/07/2019
7789                                    ; dl is head.
7790                                    ; ax is cylinder
7791                                    ; cl is sector number (assume less than 2**6 = 64 for int 13h)
7792
7793                                    ;*** for mini disks ***
7794
7795                                            cmp     word [di+47h], 1 ; [di+BDS.bdsm_ismini]
7796                                                                    ; check for mini disk
7797                                            jnz     short oknotmini       ; not mini disk.
7798                                            add     ax, [di+49h]    ; [di+BDS.bdsm_hidden_trks]
7799                                                                    ; set the physical track number
7800                                    oknotmini:
7801                                    %endif
7802                                            ; 16/12/2023 - Retro DOS v5.0 (PCDOS 7.1) IBMBIO.COM
7803                                            ;;;
7804                                            ;mov    [cs:saved_word], ax
7805 000022A3 A3[9E04]                          mov     [saved_word], ax
7806 000022A6 58                                pop     ax
7807 000022A7 F7F3                              div     bx
7808 000022A9 88D1                              mov     cl, dl
7809 000022AB FEC1                              inc     cl
7810 000022AD 8B5D15                            mov     bx, [di+15h]    ; [di+BDS.heads]
7811 000022B0 50                                push    ax
7812 000022B1 31D2                              xor     dx, dx
7813                                            ;mov    ax, [cs:saved_word]
7814 000022B3 A1[9E04]                          mov     ax, [saved_word]
7815 000022B6 F7F3                              div     bx
7816                                            ;mov    [cs:saved_word], ax
7817 000022B8 A3[9E04]                          mov     [saved_word], ax ; not necessary !? (ax must be 0)
7818 000022BB 58                                pop     ax
7819 000022BC F7F3                              div     bx              ; dl is head, ax is cylinder
7820                                            ; 16/12/2023
7821 000022BE 0E                                push    cs
```

```
                                        pop     es ; (*)
000022BF 07                             mov     bx, disksector ; (**)
000022C0 BB[5201]                       ;
                                        test    byte [di+40h], 4 ; fLBArw ; LBA read/write flag
000022C3 F6454004                       jz      short set3_chs_read
000022C7 742F                   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
000022C9 BE[591B]                       mov     si, dap_buffer ; empty_dap_buff
000022CC 56                             push    si
000022CD 87FE                           xchg    si, di
                                        ; si = BDS
                                        ; di = DAP buffer
000022CF B81000                         mov     ax, 16
000022D2 AB                             stosw           ; DAP size
000022D3 B001                           mov     al, 1
000022D5 AB                             stosw           ; sector (read) count
                                        ; buffer address
000022D6 89D8                           mov     ax, bx  ; offset disksector
000022D8 AB                             stosw
000022D9 8CC0                           mov     ax, es  ; es=ds=cs = BIOSDATA segment
000022DB AB                             stosw
                                        ; sector address (bits 0 to 31)
000022DC 8B4417                         mov     ax, [si+17h] ; [di+BDS.hiddensectors]
000022DF AB                             stosw
000022E0 8B4419                         mov     ax, [si+19h] ; [di+BDS.hiddensectors+2]
000022E3 AB                             stosw
                                        ; sector address bits 32 to 63 (0)
000022E4 31C0                           xor     ax, ax ; 0
000022E6 AB                             stosw
000022E7 AB                             stosw
                                        ;xchg   di, si
000022E8 89F7                           mov     di, si
                                        ; di = BDS
000022EA 5E                             pop     si ; DAP buffer address

000022EB 8A16[091A]                     mov     dl, [rom_drv_num] ; ds = cs
000022EF B442                           mov     ah, 42h
000022F1 CD13                           int     13h             ; DISK - IBM/MS Extension
                                                                ; EXTENDED READ (DL - drive, DS:SI - disk address packet)
                                        ;pop    si ; *
000022F3 7324                           jnc     short set3_read_ok
                                set3_err:
                                        ;jmp    setret
000022F5 E9B702                         jmp     ret_hard_err
                                %endif

                                set3_chs_read:
000022F8 837D7901                       cmp     word [di+79h], 1 ; [di+BDS.bdsm_ismini] ; check for mini disk
000022FC 7503                           jnz     short oknotmini
000022FE 03457B                         add     ax, [di+7Bh]   ; [di+BDS.bdsm_hidden_trks]
                                        ;;;

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

00002301 D0CC                           ror     ah, 1           ; move high two bits of cyl to high
00002303 D0CC                           ror     ah, 1           ; two bits of upper byte
00002305 80E4C0                         and     ah, 0C0h        ; turn off remainder of bits
00002308 08E1                           or      cl, ah          ; move two bits to correct spot
0000230A 88C5                           mov     ch, al          ; ch iscylinder (low 8 bits)
                                                                ; cl is sector + 2 high bits of cylinder
0000230C 88D6                           mov     dh, dl          ; dh is head

                                        ; 12/08/2023 (ds=cs)
0000230E 8A16[091A]                     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.
00002312 B80102                         mov     ax, 201h
00002315 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
00002317 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)
                                        ;;;
00002319 81BFFE0155AA                   cmp     word [bx+1FEh], 0AA55h
0000231F 7541                           jne     short invalid_boot_record
                                        ; 16/12/2023
                                        ; 12/08/2023
                                        ; ds = cs = BIOSDATA segment ('disksector:' is in BIOSDATA)
00002321 803FE9                         cmp     byte [bx], 0E9h         ; is it a near jump?
00002324 740B                           je      short check_1_ok ; yes
00002326 803FEB                         cmp     byte [bx], 0EBh         ; is it a short jump?
00002329 7537                           jne     short invalid_boot_record ; no
0000232B 807F0290                       cmp     byte [bx+2], 90h ; yes, is the next one a nop?
0000232F 7531                           jne     short invalid_boot_record ; no, invalid bs ; 11/08/2023
                                check_1_ok:
00002331 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
00002335 740E                           jz      short check_2  ; FAT32 bs

                                        ; FAT16 or FAT12 bs

                                        ;push   ds
                                        ;push   si  ; (not necessary)
00002337 57                             push    di
                                        ; es=ds=cs = BIOSDATA segment
                                        ;push   es
                                        ;pop    ds

                                        ;mov    cx, 28
00002338 B90E00                         mov     cx, 14 ; *
0000233B 8D7724                         lea     si, [bx+24h]    ; move offset 36 to 63
                                                                ;       to offset 64 (28 bytes)
0000233E 8D7F40                         lea     di, [bx+40h]    ; boot sector offset 64
00002341 FC                             cld     ; (not necessary, 'std' is not used before here)
                                        ;rep movsb
00002342 F3A5                           rep movsw ; *
00002344 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
00002345 8A4715                         mov     al, [bx+21]     ; [bx+EXT_BOOT.BPB+EBPB.MEDIADESCRIPTOR]
                                                                ; get the mediadescriptor byte
00002348 24F0                           and     al, 0F0h        ; mask off low nibble
0000234A 3CF0                           cmp     al, 0F0h        ; is high nibble = 0Fh?
```

```
8070 0000234C 7514                          jne     short invalid_boot_record ; no, invalid boot record
8071 0000234E 817F0B0002                    cmp     word [bx+11], 512 ; [bx+EXT_BOOT.BPB+EBPB.BYTESPERSECTOR]
8072 00002353 750D                          jne     short invalid_boot_record ; invalidate non 512 byte sectors
8073
8074                                    check2_ok:  ; yes, mediadescriptor ok.
8075 00002355 8A470D                        mov     al, [bx+13]     ; now make sure that
8076                                                                ; the sectorspercluster is
8077                                                                ; a power of 2
8078                                                                ;
8079                                                                ; [bx++EXT_BOOT.BPB+EBPB.SECTORSPERCLUSTER]
8080                                                                ; get the sectorspercluster
8081                                        ;;;
8082
8083 00002358 08C0                          or      al, al          ; is it zero?
8084 0000235A 7406                          jz      short invalid_boot_record ; yes, invalid boot record
8085
8086                                    ck_power_of_two:
8087 0000235C D0E8                          shr     al, 1           ; shift until first bit emerges
8088 0000235E 73FC                          jnc     short ck_power_of_two
8089 00002360 7406                          jz      short valid_boot_record
8090
8091                                    invalid_boot_record:
8092                                        ; 18/12/2023
8093                                        ;pop    ax
8094                                        ;pop    bx ; +
8095 00002362 E96001                        jmp     unknown         ; jump to invalid boot record
8096                                                                ; unformatted or illegal media.
8097                                    ; 16/12/2023
8098                                    ; ---------------------------------------------------------------------------
8099                                    ;   ; 12/08/2023
8100                                    ;setret_brdg:
8101                                    ;           jmp     setret
8102                                    ; ---------------------------------------------------------------------------
8103
8104                                    unknown3_0_j:
8105 00002365 E96101                        jmp     unknown3_0      ; legally formatted media,
8106                                                                ; although, content might be bad.
8107                                    ; ---------------------------------------------------------------------------
8108
8109                                    valid_boot_record:
8110                                        ; 18/12/2023
8111                                        ;pop    ax
8112                                        ;pop    bx ; +
8113
8114                                        ; 18/12/2023
8115                                        ; bx = offset disksector ; +
8116
8117                                    ; Signature found. Now check version.
8118
8119                                        ; 14/08/2023
8120 00002368 817F08322E                    cmp     word [bx+8], '2.'
8121                                        ;cmp    word [cs:bx+8], '2.' ; 03/10/2022 (NASM syntax)
8122                                        ;;cmp   word ptr cs:[bx+8], 2E32h ; '2.'
8123 0000236D 7506                          jne     short try5
8124 0000236F 807F0A30                      cmp     byte [bx+10], '0'
8125                                        ;cmp    byte [cs:bx+0Ah], '0' ; 03/10/2022 (NASM syntax)
8126                                        ;;cmp   byte ptr cs:[bx+0Ah], 30h ; '0'
8127                                        ; 12/08/2023
8128                                        ;jnz    short try5
8129                                        ;jmp    short copybpb
8130 00002373 7425                          je      short copybpb
8131
8132                                    ;; ---------------------------------------------------------------------------
8133                                    ;;  ; 12/08/2023
8134                                    ;;setret_brdg:
8135                                    ;;          jmp     setret
8136                                    ;; ---------------------------------------------------------------------------
8137                                    ;
8138                                    ;unknown3_0_j:
8139                                    ;           jmp     unknown3_0      ; legally formatted media,
8140                                    ;                                   ; although, content might be bad.
8141                                    ; ---------------------------------------------------------------------------
8142
8143                                    try5:
8144 00002375 E83B02                        call    cover_fdisk_bug
8145
8146                                    ; see if it is an os2 signature
8147
8148                                        ; 12/08/2023
8149                                        ; ds = cs = BIOSDATA segment
8150 00002378 817F08302E                    cmp     word [bx+8], '0.'
8151                                        ;cmp    word [cs:bx+8], '0.' ; 03/10/2022 (NASM syntax)
8152                                        ;;cmp   word ptr cs:[bx+8], 2E30h ; '0.'
8153 0000237D 750C                          jne     short no_os2
8154 0000237F 8A4707                        mov     al, [bx+7] ; 12/08/2023
8155                                        ;mov    al, [cs:bx+7]  ; 17/10/2022 (NASM syntax)
8156 00002382 2C31                          sub     al, '1'
8157                                        ;sub    al, 31h         ; '1'
8158 00002384 24FE                          and     al, 0FEh
8159 00002386 7412                          jz      short copybpb  ; accept either       '1' or '2'
8160 00002388 E93A01                        jmp     unknown
8161                                    ; ---------------------------------------------------------------------------
8162
8163                                    ; no os2 signature, this is to check for real dos versions
8164
8165                                    no_os2:
8166                                        ; 12/08/2023
8167                                        ; ds = cs = BIOSDATA
8168 0000238B 817F08332E                    cmp     word [bx+8], '3.'
8169                                        ;cmp    word [cs:bx+8], '3.' ; 03/10/2022 (NASM syntax)
8170                                        ;;cmp   word ptr cs:[bx+8], 2E33h ; '3.'
8171 00002390 72D3                          jb      short unknown3_0_j ; must be 2.1 boot record.
8172                                                                    ; do not trust it, but still legal.
8173 00002392 7506                          jnz     short copybpb  ; honor os2 boot record
8174                                                                ; or dos 4.0 version
8175 00002394 807F0A31                      cmp     byte [bx+10], '1' ; 12/08/2023
8176                                        ;cmp    byte [cs:bx+10], '1'
8177                                        ;;cmp   byte ptr cs:[bx+0Ah], 31h ; '1'
8178 00002398 72CB                          jb      short unknown3_0_j ; if version >= 3.1, then o.k.
8179                                    copybpb:
8180
8181                                    ; 03/10/2022
8182
8183                                    ; we have a valid boot sector. use the bpb in it to build the
8184                                    ; bpb in bios. it is assumed that only
8185                                    ;   BDS_BPB.BPB_SECTORSPERCLUSTER
8186                                    ;   BDS_BPB.BPB_ROOTENTRIES, and
8187                                    ;   BDS_BPB.BPB_SECTORSPERFAT
8188                                    ; need to be set (all other values in already). fbigfat is also set.
8189
8190                                    ; if it is non fat based system, then just copy the bpb from the boot sector
8191                                    ; into the bpb in bds table, and also set the boot serial number, volume id,
8192                                    ; and system id according to the boot record.
8193                                    ; for the non_fat system, don't need to set the other value. so just do goodret.
```

```
8194
8195                                    ; 17/12/2023 - Retro DOS v5.0 (PCDOS 7.1) IBMBIO.COM
8196                                    ; (PCDOS 7.1 IBMBIO.COM - BIOSDATA:2787h)
8197                                    ;;;
8198                                    ; 17/12/2023
8199 0000239A BE[5D01]                 mov     si, disksector+11
8200                                    ;sub    ch, ch ; ; (ch may be > 0)
8201 0000239D 29C9                     sub     cx, cx ; 0
8202                                    ;mov    cl, [disksector+16] ; BPB_NumFATs
8203 0000239F 8A4C05                   mov     cl, [si+5] ; number of FATs
8204
8205                                    ; NOTE: This check is not proper for FAT32 boot sector (standard spec)
8206                                    ; (after PCDOS 7.1). So, it is not existing in Windows ME IO.SYS
8207                                    ; Erdogan Tan - 01/09/2023 ((IBMBIO.COM 7.1 disassembly note))
8208
8209                                    ;;cmp   word ptr cs:disksector+4Dh, 0 ; ???
8210                                    ;cmp    word [disksector+4Dh], 0
8211                                    ;jnz    short check_3
8212
8213                                    ; 17/12/2023
8214                                    ; check extended boot signature (0x29)
8215                                    ;
8216                                    ; (***) NOTE: 28 bytes of FAT16/FAT12 boot sector from offset 36
8217                                    ; have been moved to offset 64 (see label 'check_1_ok:' above)
8218                                    ; ((now, BS_BootSig is at offset 66 even if it was at offset 38))
8219
8220                                    ;cmp    cs:disksector+42h, 29h; BS_BootSig (FAT32)
8221 000023A2 803E[9401]29               cmp     byte [disksector+42h], 29h ; BS_BootSig (***)
8222                                    ;jmp    short check_4
8223                                check_3:
8224                                    ;;cmp   cs:disksector+26h, 29h; BS_BootSig (FAT16/FAT12)
8225                                    ;cmp    byte [disksector+26h], 29h ; (***)
8226                                check_4:
8227 000023A7 7538                     jnz     short copybpb_fat     ; conventional fat system
8228
8229                                ; 17/12/2023
8230                                %if 0
8231                                    ; 10/12/2022
8232                                    ; (number of FATs optimization)
8233                                    mov     si, disksector+11 ; disksector+0Bh
8234                                    ;;mov   cl, [cs:disksector+10h] ; Number of FATs (may be 2 or 1)
8235                                    ;mov    cl, [cs:si+05h]
8236                                    ; 12/08/2023
8237                                    ; ds = cs = BIOSDATA segment (0070h)
8238                                    mov     cl, [si+05h] ; number of FATs
8239
8240                                    cmp     byte [si+1Bh], 29h ; 12/08/2023
8241                                    ;cmp    byte [cs:si+1Bh], 29h ; 10/12/2022
8242                                    ;;cmp   byte [cs:disksector+26h], 29h ; 17/10/2022
8243                                                    ; [disksector+EXT_BOOT.SIG]
8244                                                    ; EXT_BOOT_SIGNATURE
8245                                    jnz     short copybpb_fat ; conventional fat system
8246
8247                                    ; 03/10/2022
8248                                    ; 29/12/2018 - Retro DOS v4.0 modification note:
8249                                    ; Regarding 'fat_big_small' part of this (MSDOS 6.0) code
8250                                    ;           number of FATs must be 2 ; =*?=
8251                                    ; (Otherwise, '# of data sectors' would be calculated as wrong!!!)
8252                                    ;
8253                                    ;cmp    byte [disksector+EXT_BOOT.BPB+EBPB.NUMBEROFFATS], 2 ; =*?=
8254
8255                                    ; 10/12/2022
8256                                    ;cmp    byte [cs:disksector+10h], 0
8257                                                    ; [disksector+EXT_BOOT.BPB+EBPB.NUMBEROFFATS]
8258                                    ;jnz    short copybpb_fat ; a fat system.
8259                                    or      cl, cl   ; [cs:disksector+10h]
8260                                    jnz     short copybpb_fat ; a fat system.
8261                                %endif
8262
8263                                    ; 17/12/2023 - Retro DOS v5.0
8264                                    ;;cmp   byte [cs:disksector+10h], 0 ; BPB.fats
8265                                    ;cmp    byte [disksector+10h], 0 ; BPB_NUmFATs
8266                                    ;jnz    short copybpb_fat ; a fat system
8267                                    ; 17/12/2023
8268                                    ; cl = [disksector+10h]
8269 000023A9 20C9                     and     cl, cl ; 0 ?
8270 000023AB 7534                     jnz     short copybpb_fat ; a fat system
8271
8272                                ; non fat based    media.
8273
8274 000023AD 57                        push    di  ; BDS
8275                                    ; 12/08/2023
8276                                    ;push   ds  ; ds = cs = BIOSDATA segment
8277
8278                                    ; 17/12/2023
8279                                    ; es = ds = cs
8280                                    ;push   ds
8281                                    ;pop    es
8282
8283                                    ; 12/08/2023
8284                                    ; ds = cs
8285                                    ;push   cs
8286                                    ;pop    ds
8287
8288                                    ; 10/12/2022
8289                                    ; (number of FATs optimization)
8290                                    ; SI = disksector+11
8291                                    ; 17/10/2022
8292                                    ;;mov   si, 159h       ; disksector+EXT_BOOT.BPB
8293                                    ;mov    si, disksector+11
8294 000023AE 83C706                   add     di, 6          ; add di,BDS.BPB
8295
8296                                ; just for completeness, we'll make sure that total_sectors and
8297                                ; big_total_sectors aren't both zero. I've seen examples of
8298                                ; this on DOS 3.30 boot records. I don't know exactly how it
8299                                ; got that way. If it occurs, then use the values from the
8300                                ; partition table.
8301
8302                                    ; 17/12/2023
8303                                    ; cx = 0
8304                                    ; 18/12/2022
8305                                    ;sub    cx, cx
8306
8307                                    ;cmp    word [cs:si+8], 0     ; [cs:si+EBPB.TOTALSECTORS]
8308                                    ;jnz    short already_nonz
8309                                    ;                     ; how about big_total?
8310                                    ;cmp    word [cs:si+15h], 0   ; [cs:si+EBPB.BIGTOTALSECTORS]
8311                                    ;jnz    short already_nonz ; we're okay if any are != 0
8312                                    ;cmp    word [cs:si+17h], 0   ; [cs:si+EBPB.BIGTOTALSECTORS+2]
8313                                    ;jnz    short already_nonz
8314
8315                                    ; 12/08/2023
8316                                    ; ds = cs = BIOSDATA segment (0070h)
8317
```

```
                                        ; 17/12/2023
                                        ; 12/08/2023
000023B1 394C08                         cmp     [si+8], cx ; 0          ; [si+EBPB.TOTALSECTORS]
000023B4 751C                           jnz     short already_nonz
                                                ; how about big_total?
000023B6 394C15                         cmp     [si+15h], cx ; 0        ; [si+EBPB.BIGTOTALSECTORS]
000023B9 7517                           jnz     short already_nonz ; we're okay if any are != 0
000023BB 394C17                         cmp     [si+17h], cx ; 0        ; [si+EBPB.BIGTOTALSECTORS+2]
000023BE 7512                           jnz     short already_nonz

                                ; now let's copy the values from the partition table (now in the BDS)
                                ; into the BPB in the boot sector buffer, before they get copied back.

000023C0 8B4508                         mov     ax, [di+8]      ; [di+BDS.totalsecs16]
                                        ; 12/08/2023
                                        ;mov    [cs:si+8], ax   ; [cs:si+EBPB.TOTALSECTORS]
000023C3 894408                         mov     [si+8], ax
000023C6 8B4515                         mov     ax, [di+15h]    ; [di+BDS.totalsecs32]
                                        ;mov    [cs:si+15h], ax         ; [cs:si+EBPB.BIGTOTALSECTORS]
000023C9 894415                         mov     [si+15h], ax
000023CC 8B4517                         mov     ax, [di+17h]    ; [di+BDS.totalsecs32+2]
                                        ;mov    [cs:si+17h], ax         ; [cs:si+EBPB.BIGTOTALSECTORS+2]
000023CF 894417                         mov     [si+17h], ax

                                already_nonz:
                                        ; 18/12/2022
                                        ; cx = 0
                                        ;mov    cl, 25
                                        ;;mov   cx, 25          ; A_BPB.size - 6 ; Use SMALL version!
                                        ; 17/12/2023 - Retro DOS v5.0
000023D2 B135                           mov     cl, 53  ; PCDOS 7.1 IBMBIO.COM
                                                        ; BDS.BPB size (25 + 28 for FAT32 parms)
000023D4 F3A4                           rep movsb
                                        ;pop    ds
                                        ; 12/08/2023
                                        ; ds = cs
                                        ;pop    bp  ; ds (on top of stack) = BIOSDATA
000023D6 5F                             pop     di  ; BDS
                                        ;push   es
                                        ;push   ds
                                        ;pop    es
                                        ;push   cs
                                        ;pop    ds
                                        ; 12/08/2023
                                        ;mov    es, bp
                                        ; ds = cs = es

                                        ; 14/08/2023
000023D7 BD[4F08]                       mov     bp, MOVMEDIAIDS ; mov_media_ids
                                        ; 18/12/2022
                                        ;mov    bp, mov_media_ids
                                        ;;mov   bp, 751h        ; mov_media_ids
                                                                ; at 2C7h:751h = 70h:2CC1h
                                                                ; set volume id, systemid, serial.
000023DA 0E                             push    cs              ; simulate far call
000023DB E895F6                         call    call_bios_code
                                        ; 12/08/2023
                                        ; ds = cs = es
                                        ;push   es
                                        ;pop    ds
                                        ;pop    es
000023DE E9C701                         jmp     goodret

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

                                ; ****** cas ---
                                ; IBM DOS 3.30 doesn't seem to mind that the TOTAL_SECTORS and
                                ; BIG_TOTAL_SECTORS field in the boot sector are 0000. This
                                ; happens with some frequency -- perhaps through some OS/2 setup
                                ; program. We haven't actually been COPYING the TOTAL_SECTORS
                                ; from the boot sector into the DPB anyway, we've just been using
                                ; it for calculating the fat size. Pretty scary, huh? For now,
                                ; we'll go ahead and copy it into the DPB, except in the case
                                ; that it equals zero, in which case we just use the values in
                                ; the DPB from the partition table.

                                ; 17/10/2022
                                ;MOVMEDIAIDS equ mov_media_ids - DOSBIOSEG_2C7h ; (751h for MSDOS 5.0 IO.SYS)
                                ;CLEARIDS equ clear_ids - DOSBIOSEG_2C7h ; (5D9h for MSDOS 5.0 IO.SYS)
                                ; 09/12/2022
                                MOVMEDIAIDS equ mov_media_ids
                                CLEARIDS equ clear_ids
                                ; 11/09/2023
                                CLEARIDS_X equ clear_ids_x

                                copybpb_fat:
                                        ; 17/12/2023
                                        ; ch = 0, cl = number of FATs
                                        ; 10/12/2022
                                        ; (number of FATs optimization)
                                        ; SI = disksector+11
                                        ; 17/10/2022
                                        ;mov    si, disksector+11
                                        ;;mov   si, 159h        ; disksector+EXT_BOOT.BPB
                                                                ; cs:si -> bpb in boot
                                        ; 17/12/2023
                                        ; dx = 0
                                        ; 08/05/2024
                                        ; 18/04/2024 (BugFix)
000023E1 31D2                           xor     dx, dx

                                        ; 12/08/2023
                                        ; ds = cs = BIOSDATA segment (0070h)
000023E3 8B4408                         mov     ax, [si+8]
                                        ;mov    ax, [cs:si+8]  ; [cs:si+EBPB.TOTALSECTORS]
                                                                ; get totsec from boot sec
000023E6 09C0                           or      ax, ax
000023E8 7514                           jnz     short copy_totsec ; if non zero, use that
000023EA 8B4415                         mov     ax, [si+15h] ; 12/08/2023
                                        ;mov    ax, [cs:si+15h]         ; [cs:si+EBPB.BIGTOTALSECTORS]
                                                                ; get the big version
                                                                ; (32 bit total       sectors)
000023ED 8B5417                         mov     dx, [si+17h] ; 12/08/2023
                                        ;mov    dx, [cs:si+17h]         ; [cs:si+EBPB.BIGTOTALSECTORS+2]
                                        ; 10/12/2022
                                        ; (number of FATs optimization)
                                        ; CL = number of FATs (2 or 1)
000023F0 89D3                           mov     bx, dx          ; see if it is a big zero
000023F2 09C3                           or      bx, ax
000023F4 7508                           jnz     short copy_totsec
                                                ; screw it. it was bogus.
000023F6 8B451B                         mov     ax, [di+1Bh]    ; [di+BDS.totalsecs32]
000023F9 8B551D                         mov     dx, [di+1Dh]    ; [di+BDS.totalsecs32+2]
000023FC EB06                           jmp     short fat_big_small
```

```
                                        ;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:
000023FE 89451B                         mov     [di+1Bh], ax    ; [di+BDS.totalsecs32]
                                                                ; make DPB match boot sec
00002401 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)

00002404 8B5C03                         mov     bx, [si+3]
                                        ;mov    bx, [cs:si+3]   ; [cs:si+EBPB.RESERVEDSECTORS]
                                                                ; get #reserved_sectors       from BPB
00002407 895D09                         mov     [di+9],bx       ; [di+BDS.resectors]
                                                                ; update BDS field
0000240A 29D8                           sub     ax, bx
0000240C 83DA00                         sbb     dx, 0           ; update the count
                                        ; 12/08/2023
0000240F 8B5C0B                         mov     bx, [si+0Bh]
                                        ;mov    bx, [cs:si+0Bh]         ; [cs:si+EBPB.SECTORSPERFAT]
                                                                ; bx = sectors/fat
00002412 895D11                         mov     [di+11h], bx    ; [di+BDS.fatsecs]
                                                                ; set in bds bpb
                                        ; 17/12/2023 - Retro DOS v5.0
                                        ;              (PCDOS 7.1 IBMBIO.COM)
00002415 53                             push    bx ; FAT sectors
00002416 09DB                           or      bx, bx
00002418 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
0000241A 8B5C19                         mov     bx, [si+19h]    ; BPB.FATSz32
                                sub_fat32_size:
0000241D 29D8                           sub     ax, bx
0000241F 1B541B                         sbb     dx, [si+1Bh]    ; BPB.FATSz32+2
                                        ;dec    cl
00002422 49                             dec     cx
00002423 7FF8                           jg      short sub_fat32_size

00002425 895D1F                         mov     [di+1Fh], bx    ; [di+BDS.fatsecs32]
00002428 8B5C1B                         mov     bx, [si+1Bh]    ; BPB.FATSz32+2
0000242B 895D21                         mov     [di+21h], bx    ; [di+BDS.fatsecs32+2]

0000242E 8B5C1D                         mov     bx, [si+1Dh]    ; BPB.BPB_ExtFlags
00002431 895D23                         mov     [di+23h], bx    ; [di+BDS.extflags]
00002434 8B5C1F                         mov     bx, [si+1Fh]    ; BPB.FSVer
00002437 895D25                         mov     [di+25h], bx    ; [di+BDS.fsver]
0000243A 8B5C21                         mov     bx, [si+21h]    ; BPB.RootClus
0000243D 895D27                         mov     [di+27h], bx    ; [di+BDS.rootdirclust]
00002440 8B5C23                         mov     bx, [si+23h]    ; BPB.RootClus+2
00002443 895D29                         mov     [di+29h], bx    ; [di+BDS.rootdirclust+2]
00002446 8B5C25                         mov     bx, [si+25h]    ; BPB.FSInfo
00002449 895D2B                         mov     [di+2Bh], bx    ; [di+BDS.fsinfo]
0000244C 8B5C27                         mov     bx, [si+27h]    ; BPB.FSInfo+2
0000244F 895D2D                         mov     [di+2Dh], bx    ; [di+BDS.fsinfo+2]
00002452 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
00002454 49                             dec     cx ; *
00002455 D3E3                           shl     bx, cl
                                        ;shl    bx, 1   ; =*?=  ; always 2 fats

00002457 29D8                           sub     ax, bx          ; sub # fat sectors
00002459 83DA00                         sbb     dx, 0
                                fat_32bit:
                                        ; 17/12/2023
0000245C 8B5C06                         mov     bx, [si+6] ; 12/08/2023
```

```
                                        ;mov    bx, [cs:si+6]  ; [cs:si+EBPB.ROOTENTRIES]
                                                               ; # root entries
0000245F 895D0C                         mov     [di+0Ch], bx   ; [di+BDS.direntries]
                                                               ; set in bds bpb
00002462 B104                           mov     cl, 4
00002464 D3EB                           shr     bx, cl         ; div by 16 ents/sector
00002466 29D8                           sub     ax, bx         ; sub # dir sectors
00002468 83DA00                         sbb     dx, 0          ;
                                                               ; dx:ax now contains the
                                                               ; # of data sectors
                                        ; 17/12/2023
                                        ; ch = 0
                                        ;xor    cx, cx ; *
0000246B 8A4C02                         mov     cl, [si+2] ; 12/08/2023
                                        ;mov    cl, [cs:si+2]  ; [cs:si+EBPB.SECTORSPERCLUSTER]
                                                               ; sectors per cluster
0000246E 884D08                         mov     [di+8], cl     ; [di+BDS.secperclus]
                                                               ; set in bios bpb
00002471 50                             push    ax
00002472 89D0                           mov     ax, dx
00002474 31D2                           xor     dx, dx
00002476 F7F1                           div     cx             ; cx = sectors per cluster
                                        ; 12/08/2023 (ds=cs)
                                        ;mov    [temp_h], ax
                                        ;;mov   [cs:temp_h], ax       ; [temp_h]:ax now contains the
                                                               ; # clusters.
                                        ; 17/12/2023
00002478 A3[9E04]                       mov     [saved_word], ax ; hw of cluster number
0000247B 58                             pop     ax
0000247C F7F1                           div     cx
                                        ; 17/12/2023
                                        ;;cmp   word [cs:temp_h], 0
                                        ;cmp    word [temp_h], 0  ; 12/08/2023
                                        ;cmp    word [saved_word], 0 ; (*)
                                        ;ja     short toobig_ret ; too big cluster number

                                        ; 17/12/2023
                                        ;;;
0000247E 5B                             pop     bx ; FAT sectors (16 bit)
                                        ;and    bx, bx ; 0 ?
0000247F 09DB                           or      bx, bx ; 0 ?
00002481 751F                           jnz     short chk_clnum_hw
                                                        ; 16 bit fat sectors > 0 ; FAT12 or FAT16 fs

00002483 813E[9E04]FF0F                 cmp     word [saved_word], 0FFFh
00002489 7503                           jne     short fat32_clust_limit
0000248B 83F8F6                         cmp     ax, 0FFF6h     ; FAT32 cluster number limit: 0FFFFFF6h
                                fat32_clust_limit:
0000248E 772D                           ja      short short toobig_ret ; too big cluster number
00002490 391E[9E04]                     cmp     [saved_word], bx ; 0 ?
                                        ;jnz    short fat16_clust_limit
00002494 7505                           jnz     short set_fbigbig_flag ; 17/12/2023
                                fat16_clust_limit: ; 17/12/2023
00002496 83F8F6                         cmp     ax, 0FFF6h     ; FAT16 cluster number limit: 0FFF6h
                                ;fat16_clust_limit:
00002499 760E                           jna     short fat12_clust_limit ; jbe
                                set_fbigbig_flag:  ; 17/12/2023
0000249B 800E[081A]20                   or      byte [fbigfat], 20h ; fbigbig ; FAT32 fs
000024A0 EB11                           jmp     short copymediaid
                                chk_clnum_hw:
000024A2 833E[9E04]00                   cmp     word [saved_word], 0 ; (*)
000024A7 7714                           ja      short toobig_ret ; too big cluster number
                                        ;;;
                                fat12_clust_limit:
000024A9 3DF60F                         cmp     ax, 0FF6h      ; 4096-10
                                                               ; is this 16-bit fat?
000024AC 7205                           jb      short copymediaid ; no,       small fat
                                        ; 17/10/2022
000024AE 800E[081A]40                   or      byte [fbigfat], 40h ; fbig ; FAT16 fs
                                        ;or     ds:fbigfat, 40h       ; fbig
                                                               ; 16 bit fat
                                copymediaid:
                                        ; 17/12/2023
                                        ; es = ds = cs

                                        ;push   es
                                        ;push   ds
                                        ;pop    es

                                        ; 12/08/2023
                                        ; ds = cs = BIOSDATA
                                        ;push   cs
                                        ;pop    ds
                                        ; 17/10/2022
000024B3 BD[4F08]                       mov     bp, MOVMEDIAIDS
                                        ;mov    bp, 865h       ; (PCDOS 7.1 IBMBIO.COM)
                                        ;;mov   bp, 751h       ; mov_media_ids
                                                               ; at 2C7h:751h = 70h:2CC1h
                                                               ; copy filesys_id, volume label
000024B6 0E                             push    cs             ; simulate far call
000024B7 E8B9F5                         call    call_bios_code

                                        ; 12/08/2023
                                        ;push   es
                                        ;pop    ds
                                        ; 17/12/2023
                                        ;pop    es

000024BA E9CD00                         jmp     massage_bpb    ; now final check for bpb info
                                                               ; and return.
                                ; ---------------------------------------------------------------------------

                                toobig_ret:
                                        ; 12/08/2023 (ds=cs=BIOSDATA)
000024BD 800E[081A]80                   or      byte [fbigfat], 80h ; ftoobig
                                        ;or     byte [cs:fbigfat], 80h ; ftoobig
                                                               ; too big (32 bit clust #) for FAT16
000024C2 E9E300                         jmp     goodret        ; still drive letter is       assigned
                                                               ; but useless. to big for
                                                               ; current pc dos fat file system
                                ; ---------------------------------------------------------------------------

                                unknown:
                                        ; 17/12/2023 - Retro DOS v5.0 (PCDOS 7.1 IBMBIO.COM)
000024C5 804D4002                       or      byte [di+40h], 2 ; [di+BDS.flags+1]
                                                                 ; unformatted_media
                                        ; 12/12/2022
                                        ;or     byte [di+24h], 02h
                                        ;;or    word [di+23h], 200h ; [di+BDS.flags]
                                                               ; unformatted_media
                                                               ; Set unformatted media      flag.

                                ; the boot signature may not be   recognizable,
                                ; but we should    try and read it anyway.
```

```
8690
8691                                    unknown3_0:
8692 000024C9 8B551D                              mov     dx, [di+1Dh]   ; skip setting unformatted_media bit
8693                                                                      ; [di+BDS.totalsecs32+2]
8694 000024CC 8B451B                              mov     ax, [di+1Bh]   ; [di+BDS.totalsecs32]
8695 000024CF BE[181A]                            mov     si, disktable2
8696                                    scan:                                   ; 08/08/2023
8697                                              ;cmp    dx, [cs:si]    ; total sectors hw
8698                                              ; 12/08/2023 (ds=cs)
8699 000024D2 3B14                                cmp     dx, [si]
8700 000024D4 720C                                jb      short gotparm
8701 000024D6 7705                                ja      short scan_next
8702                                              ;cmp    ax, [cs:si+2]  ; total sectors lw
8703 000024D8 3B4402                              cmp     ax, [si+2]
8704 000024DB 7605                                jbe     short gotparm
8705                                    scan_next:
8706 000024DD 83C60A                              add     si, 10         ; 5*2
8707 000024E0 EBF0                                jmp     short scan     ; covers upto 512 mb media
8708                                    ; ---------------------------------------------------------------------------
8709
8710                                    gotparm:
8711 000024E2 8A4C08                              mov     cl, [si+8]     ; fat size for fbigfat flag
8712                                              ;or     ds:fbigfat, cl
8713                                              ; 17/10/2022
8714 000024E5 080E[081A]                          or      [fbigfat], cl  ; (fbig flag, 40h or 0) ; 08/08/2023
8715                                              ; 12/08/2023
8716                                              ; ds = cs = BIOSDATA
8717 000024E9 8B4C04                              mov     cx, [si+4]
8718                                              ;mov    cx, [cs:si+4]  ; ch = number of sectors per cluster
8719                                                                      ; cl = log base 2 of ch
8720 000024EC 8B5406                              mov     dx, [si+6]
8721                                              ;mov    dx, [cs:si+6]  ; dx = number of root dir entries
8722
8723                                    ; now calculate size of fat table
8724
8725 000024EF 89550C                              mov     [di+0Ch], dx   ; [di+BDS.direntries]
8726                                                                      ; save number of (root)      dir entries
8727 000024F2 8B551D                              mov     dx, [di+1Dh]   ; [di+BDS.totalsecs32+2]
8728 000024F5 8B451B                              mov     ax, [di+1Bh]   ; [di+BDS.totalsecs32]
8729 000024F8 886D08                              mov     [di+8], ch     ; [di+BDS.secperclus]
8730                                                                      ; save sectors per cluster
8731
8732                                              ; 17/12/2023 - Retro DOS v5.0 (PCDOS 7.1 IBMBIO.COM)
8733 000024FB F606[081A]60                        test    byte [fbigfat], 60h ; fbig+fbigbig ; FAT16 or FAT32
8734                                              ; 11/09/2023
8735                                              ; 17/10/2022
8736                                              ;test   byte [fbigfat], 40h
8737                                              ;;test  ds:fbigfat, 40h        ; fbig
8738                                                                      ; if (fbigfat)
8739 00002500 751E                                jnz     short dobig    ; goto dobig; (16 bit fat)
8740
8741                                    ; we don't need to change "small fat" logic since it is guaranteed
8742                                    ; that double word total sector will not use 12 bit fat (unless
8743                                    ; it's sectors/cluster >= 16 which will never be in this case.)
8744                                    ; so in this case we assume dx = 0 !!
8745
8746 00002502 31DB                                  xor     bx, bx         ; 12 bit fat (FAT12 fs)
8747 00002504 88EB                                mov     bl, ch
8748 00002506 4B                                  dec     bx
8749 00002507 01C3                                add     bx, ax         ; dx=0
8750 00002509 D3EB                                shr     bx, cl         ; bx = 1+(bpb->maxsec+BDS.secperclus-1)/
8751 0000250B 43                                  inc     bx             ; BDS.secperclus
8752 0000250C 80E3FE                              and     bl, 0FEh       ; bx &= ~1; (=number of clusters)
8753 0000250F 89DE                                mov     si, bx
8754 00002511 D1EB                                shr     bx, 1
8755 00002513 01F3                                add     bx, si         ; number of FAT bytes ; 08/08/2023
8756 00002515 81C3FF01                            add     bx, 511        ; bx += 511 + bx/2
8757 00002519 D0EF                                shr     bh, 1          ; bh >>= 1; (=bx/512)
8758 0000251B 887D11                              mov     [di+11h], bh   ; [di+BDS.fatsecs]
8759                                                                      ; save number of fat sectors
8760 0000251E EB6A                                jmp     short massage_bpb
8761                                    ; ---------------------------------------------------------------------------
8762
8763                                    ; for bigfat we do need to extend this logic to 32 bit sector calculation.
8764
8765                                    dobig:
8766 00002520 B104                                mov     cl, 4          ; 16 (2^4) directory entries per sector
8767 00002522 52                                  push    dx             ; save total sectors (high)
8768 00002523 8B550C                              mov     dx, [di+0Ch]   ; [di+BDS.direntries]
8769 00002526 D3EA                                shr     dx, cl         ; root dir sectors = BDS.direntries / 16;
8770 00002528 29D0                                sub     ax, dx
8771 0000252A 5A                                  pop     dx
8772 0000252B 83DA00                              sbb     dx, 0          ; dx:ax = total sectors - root dir sectors
8773 0000252E 83E801                              sub     ax, 1
8774 00002531 83DA00                              sbb     dx, 0          ; dx:ax = t - r - d
8775                                                                      ; total secs - reserved       secs - root dir       secs
8776 00002534 B302                                mov     bl, 2
8777 00002536 8A7D08                              mov     bh, [di+8]     ; [di+BDS.secperclus]
8778                                                                      ; bx = 256 * BDS.secperclus + 2
8779
8780                                    ; I don't understand why to add bx here!!!
8781
8782                                              ; 29/12/2018 - Erdogan Tan (Retro DOS v4.0)
8783                                              ; 27/09/2022
8784                                              ; (Microsoft FAT32 File      System Specification,
8785                                              ; December 2000, Page 21)
8786                                              ; TmpVal1 = DskSize - (BPB_ResvdSecCnt+RootrDirSectors)
8787                                              ; TmpVal2 = (256*BPB_SecPerClus)+BPB_NumFATs
8788                                              ; 8/8/2023 (Retro DOS v5.0)
8789                                              ; If(FATType == FAT32)
8790                                              ;   TmpVal2 = TmpVal2 / 2;
8791                                              ; FATsz = (TmpVal1+(TmpVal2-1))/TmpVal2
8792                                              ; 8/8/2023 (Retro DOS v5.0)
8793                                              ; If(FATType == FAT32) {
8794                                              ;   BPB_FATSz16 = 0;
8795                                              ;   BPB_FATSz32 = FATSz;
8796                                              ;} else {
8797                                              ;   BPB_FATSz16 = LOWORD(FATSz);
8798                                              ;/* there is no BPB_FATSz32 in a FAT16 BPB */
8799                                              ;}
8800                                                                      ; dx:ax = TmpVal1, bx = TmpVal2
8801 00002539 01D8                                add     ax, bx         ;
8802 0000253B 83D200                              adc     dx, 0          ; dx:ax = TmpVal1+TmpVal2
8803 0000253E 83E801                              sub     ax, 1
8804 00002541 83DA00                              sbb     dx, 0          ; dx:ax = TmpVal1+TmpVal2-1
8805
8806                                              ;;;
8807                                              ; 17/12/2023 - Retro DOS v5.0 (PCDOS 7.1 IBMBIO.COM)
8808 00002544 F606[081A]20                        test    byte [fbigfat], 20h ; fbigbig (FAT32) flag
8809 00002549 740D                                jz      short dobig1
8810
8811 0000254B D1EB                                shr     bx, 1           ; TmpVal2 = TmpVal2 / 2
8812                                                                      ; dx:ax = TmpVal1+(2*TmpVal2)-1
8813 0000254D 83E81F                              sub     ax, 31          ; reserved sectors = 32 (for FAT32 fs) /// 1+31 = 32
```

```
8814 00002550 83DA00                            sbb     dx, 0
8815 00002553 29D8                              sub     ax, bx
8816 00002555 83DA00                            sbb     dx, 0           ; dx:ax = TmpVal1+(2*TmpVal2)-TmpVal2-1
8817                                                                    ;       = TmpVal1+(TmpVal2-1)
8818                                    dobig1:
8819 00002558 50                                push    ax              ; save lw of dividend
8820 00002559 89D0                              mov     ax, dx          ; divide hw of dx:ax at first (as 1st stage)
8821 0000255B 31D2                              xor     dx, dx
8822 0000255D F7F3                              div     bx              ; 32 bit division, dx:ax/bx
8823                                                                    ; remainder in dx is hw of 2nd stage dividend
8824 0000255F 89C5                              mov     bp, ax          ; hw of quotient
8825 00002561 58                                pop     ax              ; restore lw of dividend (of 1st stage)
8826                                            ;;;
8827
8828                                    ; assuming dx in the table will never be bigger than bx.
8829
8830 00002562 F7F3                              div     bx              ; BDS.fatsecs =
8831                                                                    ; ceil((total-dir-res)/(256*BDS.secperclus+2))
8832 00002564 894511                            mov     [di+11h], ax    ; [di+BDS.fatsecs]
8833                                                                    ; number of fat       sectors
8834                                            ;;;
8835
8836                                            ; 18/12/2023 - Retro DOS v5.0 (PCDOS 7.1 IBMBIO.COM)
8837 00002567 8A1E[081A]                        mov     bl, [fbigfat]
8838 0000256B 885D3B                            mov     [di+3Bh], bl    ; [di+BDS.fatsiz] ; fat size flag
8839
8840 0000256E F6C320                            test    bl, 20h         ; fbigbig (FAT32) flag
8841 00002571 7410                              jz      short dobig2    ; not FAT32
8842
8843 00002573 89451F                            mov     [di+1Fh], ax    ; [di+BDS.fatsecs32]
8844 00002576 896D21                            mov     [di+21h], bp    ; [di+BDS.fatsecs32+2]
8845 00002579 C745110000                        mov     word [di+11h], 0 ; [di+BDS.fatsecs] = 0
8846                                                                    ; clear 16 bit FAT size field
8847 0000257E C745092000                        mov     word [di+9], 32         ; [di+BDS.resectors]
8848                                                                    ; set reserved sectors to 32 (FAT32 de facto)
8849                                    dobig2:
8850                                            ;;;
8851
8852                                    ; now, set the default filesys_id, volume label, serial number
8853
8854                                            ; 05/08/2023
8855                                            ; [di+1Fh] = [fbigfat]
8856                                            ;
8857                                            ;;mov   bl, ds:fbigfat
8858                                            ;; 17/10/2022
8859                                            ;mov    bl, [fbigfat]
8860                                            ;mov    [di+1Fh], bl    ; [di+BDS.fatsiz] ; fat       size flag
8861
8862                                            ; 12/08/2023
8863                                            ;push   ds ; ds = cs = BIOSDATA
8864
8865                                            ; 17/12/2023
8866                                            ; es = ds = cs
8867                                            ;push   ds
8868                                            ;pop    es
8869
8870                                            ; 12/08/2023
8871                                            ; ds = cs = BIOSDATA
8872                                            ;push   cs
8873                                            ;pop    ds
8874
8875                                            ; 18/12/2023 - Retro DOS v5.0
8876                                            ; bl = [fbigfat] (clear_ids_x uses bl value here)
8877                                            ; 11/09/2023
8878                                            ;mov    al, [fbigfat]
8879 00002583 BD[A106]                          mov     bp, CLEARIDS_X ; clear_ids_x (uses AL value here)
8880                                            ; 17/10/2022
8881                                            ;mov    bp, CLEARIDS
8882                                            ;;mov   bp, 5D9h        ; clear_ids
8883                                                                    ; at 2C7h:5D9h = 70h:2B49h
8884                                                                    ; at BIOSCODE:06ABh
8885                                                                    ;       in PCDOS 7.1 IBMBIO.COM
8886 00002586 0E                                push    cs
8887 00002587 E8E9F4                            call    call_bios_code
8888
8889                                            ; 12/08/2023
8890                                            ;pop    ds ; ds = cs = BIOSDATA
8891
8892                                    ; at this point, in bpb of bds table, BDS_BPB.BPB_BIGTOTALSECTORS which is
8893                                    ; set according to the partition information. we are going to
8894                                    ; see if (hidden sectors + total sectors) > a word. if it is true,
8895                                    ; then no change. otherwise, BDS_BPB.BPB_BIGTOTALSECTORS will be moved
8896                                    ; to BDS_BPB.BPB_TOTALSECTORS and BDS_BPB.BPB_BIGTOTALSECTORS will be set to 0.
8897                                    ; we don't do this for the bpb information from the boot record. we
8898                                    ; are not going to change the bpb information from the boot record.
8899
8900                                    massage_bpb:
8901                                            ; 05/08/2023
8902                                            ; [di+1Fh] = [fbigfat]
8903                                            ;
8904                                            ;; 12/12/2022
8905                                            ;mov    bl, [fbigfat]
8906                                            ;mov    [di+1Fh], bl    ; [di+BDS.fatsiz]
8907                                            ;                       ; set size of fat on media
8908                                            ;
8909 0000258A 8B551D                            mov     dx, [di+1Dh]    ; [di+BDS.totalsecs32+2]
8910 0000258D 8B451B                            mov     ax, [di+1Bh]    ; [di+BDS.totalsecs32]
8911                                            ; 11/09/2023
8912 00002590 09D2                              or      dx, dx
8913 00002592 7514                              jnz     short goodret
8914                                            ;cmp    dx, 0           ; double word total sectors?
8915                                            ;;ja    short goodret   ; don't have to change it.
8916                                            ;; 12/12/2022
8917                                            ;ja     short short goodret2
8918                                            ;cmp    word [di+19h], 0 ; [di+BDS.hiddensecs+2]
8919                                            ;ja     short goodret   ; don't have to change it.
8920                                            ; 12/12/2022
8921 00002594 395519                            cmp     [di+19h], dx ; 0
8922                                            ;ja     short goodret2
8923 00002597 770F                              ja      short goodret   ; 11/09/2023
8924 00002599 034517                            add     ax, [di+17h]    ; [di+BDS.hiddensecs]
8925                                            ;jb     short goodret
8926                                            ; 12/12/2022
8927                                            ;jc     short goodret
8928 0000259C 7209                              jc      short goodret_clc ; 11/09/2023
8929 0000259E 8B451B                            mov     ax, [di+1Bh]    ; [di+BDS.totalsecs32]
8930 000025A1 89450E                            mov     [di+0Eh], ax    ; [di+BDS.totalsecs16]
8931                                            ;mov    word [di+1Bh], 0 ; [di+BDS.totalsecs32]
8932                                            ; 12/12/2022
8933 000025A4 89551B                            mov     [di+1Bh], dx ; 0
8934                                    goodret_clc:
8935                                            ; 11/09/2023
8936 000025A7 F8                                clc
8937                                    goodret:
```

```
                                        ;mov    bl, ds:fbigfat
                                        ; 11/09/2023
                                        ; 12/12/2022
                                        ; 17/10/2022
000025A8 8A1E[081A]                     mov     bl, [fbigfat]
                                        ; 17/12/2023 - Retro DOS v5.0 (PCDOS 7.1 IBMBIO.COM)
000025AC 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:
000025AF 07                             pop     es
                                        ;pop    ds      ; ds = cs = BIOSDATA ; 14/08/2023
000025B0 5B                             pop     bx
000025B1 5F                             pop     di
000025B2 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
000025B3 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
000025B7 7426                           je      short cfb_retit       ; if extended bpb, then       >= pc dos 4.00

000025B9 817F073130                     cmp     word [bx+7], 3031h
                                        ;cmp    word [cs:bx+7], 3031h ; '10' ; os2 1.0 = ibm 10.0
000025BE 7506                           jne     short cfb_chk_totalsecs ; 11/08/2023
000025C0 807F0A30                       cmp     byte [bx+10], '0'
                                        ;cmp    byte [cs:bx+10], '0'
000025C4 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
000025C6 8B4713                         mov     ax, [bx+19]    ; [bx+EBPB.TOTALSECTORS]
000025C9 21C0                           and     ax, ax ; 0 ?
000025CB 7412                           jz      short cfb_retit

                                        ;mov    ax, [bx+19]
000025CD 03471C                         add     ax, [bx+28]    ; [bx+EBPB.HIDDENSECTORS]
000025D0 730D                           jnc     short cfb_retit
000025D2 750B                           jnz     short cfb_retit
                                        ; ax = 0               ; 0 -> 0FFFFh
000025D4 FF4F13                         dec     word [bx+19]   ; then decrease       BPB_TOTALSECTORS by 1

000025D7 836D1B01                       sub     word [di+1Bh], 1 ; [di+BDS.totalsecs32]
000025DB 835D1D00                       sbb     word [di+1Dh], 0 ; [di+BDS.totalsecs32+2]
                                cfb_retit:
                                        ; 18/12/2023
                                        ;pop    si
                                        ;pop    dx
                                        ;pop    ax

000025DF C3                             retn

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

000025E0 0200                   word2:              dw 2
000025E2 0300                   word3:              dw 3
000025E4 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.
                                
                                        ; 18/12/2023 - Retro DOS v5.0
                                        ; (PCDOS 7.1 IBMBIO.COM - BIOSDATA:2A43h)
                                setdrvparms:
                                        ; 12/12/2023
                                        ; ds = cs
000025E6 31DB                           xor     bx, bx
                                        ; 18/10/2022
000025E8 C43E[1901]                     les     di, [start_bds] ; get first bds in list
                                _next_bds:
000025EC 06                             push    es
000025ED 57                             push    di
                                
                                        ; 18/12/2023 - Retro DOS v5.0 (PCDOS 7.1 IBMBIO.COM)
000025EE 268A5D3E                       mov     bl, [es:di+3Eh]         ; [es:di+BDS.formfactor]
                                        ;mov    bl, [es:di+22h]         ; [es:di+BDS.formfactor]
                                
000025F2 80FB05                         cmp     bl, 5           ; ffHardFile
000025F5 753A                           jnz     short nothardff
000025F7 31D2                           xor     dx, dx
000025F9 268B450E                       mov     ax, [es:di+0Eh]         ; [es:di+BDS.totalsecs16]
000025FD 09C0                           or      ax, ax
000025FF 7508                           jnz     short get_ccyl
00002601 268B551D                       mov     dx, [es:di+1Dh]         ; [es:di+BDS.totalsecs32+2]
00002605 268B451B                       mov     ax, [es:di+1Bh]         ; [es:di+BDS.totalsecs32]
                                get_ccyl:
00002609 52                             push    dx
0000260A 50                             push    ax
0000260B 268B4515                       mov     ax, [es:di+15h]         ; [es:di+BDS.heads]
0000260F 26F76513                       mul     word [es:di+13h] ; [es:di+BDS.secpertrack]
                                                                        ; assume sectors per cyl. < 64k.
00002613 89C1                           mov     cx, ax          ; cx has # sectors per cylinder
00002615 58                             pop     ax
00002616 5A                             pop     dx              ; dx:ax = total sectors
00002617 50                             push    ax
00002618 89D0                           mov     ax, dx
0000261A 31D2                           xor     dx, dx
0000261C F7F1                           div     cx
                                        ; 12/12/2023  ; !*!
                                        ; (data segment may not be same with code segment here)
                                        ;mov    [cs:temp_h], ax         ; ax be 0 here.
                                        ; 18/12/2023 - Retro DOS v5.0
                                        ;mov    [cs:saved_word], ax
0000261E 58                             pop     ax
0000261F F7F1                           div     cx              ; div #sec by sec/cyl to get # cyl.
00002621 09D2                           or      dx, dx
00002623 7401                           jz      short no_cyl_rnd ; came out even
00002625 40                             inc     ax              ; round up
                                no_cyl_rnd:
                                        ; 18/12/2023 - Retro DOS v5.0
00002626 26894541                       mov     [es:di+41h], ax         ; [es:di+BDS.cylinders]
                                        ;mov    [es:di+25h], ax         ; [es:di+BDS.cylinders]
                                
0000262A 06                             push    es
0000262B 1F                             pop     ds  ; !*! ; 12/12/2023
                                
0000262C 8D7506                         lea     si, [di+6]      ; [di+BDS.bytespersec]
                                                                ; ds:si -> bpb for hard         file
0000262F EB55                           jmp     short set_recbpb
                                ; ---------------------------------------------------------------------------
                                
                                nothardff:
00002631 0E                             push    cs
00002632 1F                             pop     ds
                                
                                ; if fake floppy drive variable is set then we don't have to handle this bds.
                                ; we can just go and deal with the next bds at label go_to_next_bds.
                                
                                        ; 10/12/2022
                                        ; ds = cs
                                        ; 17/10/2022 (ds=cs)
00002633 803E[131A]01                   cmp     byte [fakefloppydrv], 1
                                        ;cmp    byte [cs:fakefloppydrv], 1
00002638 7454                           jz      short go_to_next_bds
0000263A 80FB07                         cmp     bl, 7           ; ffother
                                                                ; special case "other" type of medium
0000263D 753D                           jnz     short not_process_other
                                process_other:
0000263F 31D2                           xor     dx, dx
                                
                                        ;mov    ax, [di+25h]    ; [di+BDS.cylinders]
                                        ;mul    word [di+36h]   ; [di+BDS.rheads]
                                        ;mul    word [di+34h]   ; [di+BDS.rsecpertrack]
                                        ;mov    [di+2Fh], ax    ; [di+BDS.rtotalsecs16]
                                        ;                       ; have the total number of sectors
                                        ; 18/12/2023 - Retro DOS v5.0
00002641 8B4541                         mov     ax, [di+41h]    ; [di+BDS.cylinders]
00002644 F76552                         mul     word [di+52h]   ; [di+BDS.rheads]
00002647 F76550                         mul     word [di+50h]   ; [di+BDS.rsecpertrack]
0000264A 89454B                         mov     [di+4Bh], ax    ; [di+BDS.rtotalsecs16]
                                                                ; have the total number of sectors
0000264D 48                             dec     ax
0000264E B201                           mov     dl, 1
                                _again:
00002650 3DF60F                         cmp     ax, 0FF6h       ; 4096-10
00002653 7206                           jb      short _@@
00002655 D1E8                           shr     ax, 1
00002657 D0E2                           shl     dl, 1
00002659 EBF5                           jmp     short _again
                                ; ---------------------------------------------------------------------------
                                
                                _@@:
0000265B 80FA01                         cmp     dl, 1           ; is it a small disk ?
0000265E 7405                           jz      short __@@      ; yes, 224 root         entries is enuf
                                
                                        ; 18/12/2023 - Retro DOS v5.0
00002660 C74549F000                     mov     word [di+49h], 240 ; [di+BDS.rdirentries]
                                        ;mov    word [di+2Dh], 240 ; [di+BDS.rdirentries]
                                __@@:
                                        ; 18/12/2023 - Retro DOS v5.0
00002665 885545                         mov     [di+45h], dl    ; [di+BDS.rsecperclus]
                                        ;mov    [di+29h], dl    ; [di+BDS.rsecperclus]
                                
                                ; logic to get the sectors/fat area.
                                ; fat entry is assumed to be 1.5 bytes!!!
                                
                                        ; 10/12/2022
                                        ; ds = cs
                                        ; 17/10/2022 (ds=cs)
00002668 F726[E225]                     mul     word [word3]    ; * 3
0000266C F736[E025]                     div     word [word2]    ; / 2
00002670 31D2                           xor     dx, dx
00002672 F736[E425]                     div     word [word512] ; / 512
```

```
                                        ;
                                        ; 10/12/2022
                                        ;mul    word [cs:word3]       ; * 3
                                        ;div    word [cs:word2]       ; / 2
                                        ;xor    dx, dx
                                        ;div    word [cs:word512] ; / 512
                                        ;
00002676 40                             inc     ax              ; + 1
                                no_round_up:
                                        ; 18/12/2023 - Retro DOS v5.0
00002677 89454E                         mov     [di+4Eh], ax    ; [di+BDS.rfatsecs]
                                        ;mov    [di+32h], ax    ; [di+BDS.rfatsecs]

0000267A EB12                           jmp     short go_to_next_bds
                                ; ---------------------------------------------------------------------------

                                not_process_other:
0000267C 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
0000267E 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
00002682 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
00002686 8D7D43                         lea     di, [di+43h]    ; [di+BDS.R_BPB]
                                                                ; es:di -> recbpb
00002689 B93500                         mov     cx, 53          ; bpbx.size
0000268C F3A4                           rep movsb               ; move (size bpbx) byte
                                go_to_next_bds:
0000268E 5F                             pop     di
0000268F 07                             pop     es              ; restore pointer to bds
00002690 26C43D                         les     di, [es:di]     ; [es:di+BDS.link]
00002693 83FFFF                         cmp     di, 0FFFFh      ; -1
00002696 740A                           jz      short got_end_of_bds_chain
00002698 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:
0000269B 98                             cbw
0000269C 89C2                           mov     dx, ax
0000269E B401                           mov     ah, 1
000026A0 CD17                           int     17h             ; PRINTER - INITIALIZE
                                                                ; DX = printer port (0-3)
                                                                ; Return: AH = status
                                got_end_of_bds_chain:     ; 18/12/2022
000026A2 C3                             retn

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

                                ; al = device number

                                aux_init:
000026A3 98                             cbw
000026A4 89C2                           mov     dx, ax
                                        ;mov    al, 0A3h        ; RSINIT ; 0A3h
                                                                ; 2400,n,1,8 (msequ.inc)
                                        ;mov    ah, 0
                                        ; 10/12/2022
000026A6 B8A300                         mov     ax, 00A3h
000026A9 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
000026AB 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:
000026AC 8A36[5F1A]                         mov     dh, [hnum]      ; get number of hardfiles
                                            ; 10/12/2022
000026B0 20F6                               and     dh, dh
                                            ;cmp    dh, 0
000026B2 743C                               jz      short domininet       ; no hard file?       then exit.
000026B4 B280                               mov     dl, 80h         ; start with hardfile 80h
                                domini_loop:
                                            ; 18/12/2023 - Retro DOS v5.0
000026B6 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
000026B8 A3[0222]                           mov     [ep_start_sector], ax
000026BB A3[0422]                           mov     [ep_start_sector+2], ax
000026BE A3[0622]                           mov     [ep_hidden_secs], ax
000026C1 A3[0822]                           mov     [ep_hidden_secs+2], ax
                                            ;
000026C4 52                                 push    dx
000026C5 8816[5E1A]                         mov     [rom_minidisk_num], dl
000026C9 B408                               mov     ah, 8
000026CB 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
000026CD 31C0                               xor     ax, ax
000026CF 88F0                               mov     al, dh  ; <= 255
000026D1 40                                 inc     ax      ; (0FFh -> 100h)
000026D2 A3[641A]                           mov     [mini_hdlim], ax ; # of heads
                                            ;and    cl, 3Fh
                                            ;mov    al, cl
                                            ; 08/08/2023
000026D5 88C8                               mov     al, cl
000026D7 83E03F                             and     ax, 3Fh
000026DA A3[661A]                           mov     [mini_seclim], ax ; # of sectors/track

                                            ; 18/12/2023
                                            ;push   es ; * ; not necessary
000026DD 8A16[5E1A]                         mov     dl, [rom_minidisk_num]
000026E1 E866FA                             call    getboot         ; read master boot record into
                                                                    ; initbootsegment:bootbias
000026E4 7203                               jc      short domininext
000026E6 E80800                             call    find_mini_partition
                                domininext:
                                            ;pop    es ; *
000026E9 5A                                 pop     dx
000026EA FEC2                               inc     dl              ; next hard file
000026EC FECE                               dec     dh
000026EE 75C6                               jnz     short domini_loop
                                dominiret:
000026F0 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:
000026F1 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)
```

```
9434                                            ;              ; (64*4 = 256 pte's, 256*16 = 4096, + 26 = 4122)
9435                                            ;jg     short fmpnextfound
9436
9437 000026F5 26803F05                         cmp     byte [es:bx], 5 ; 05h = extended partition id.
9438 000026F9 7410                             je      short fmpgot ; Extended DOS CHS
9439
9440                                            ; 19/12/2023 - Retro DOS v5.0
9441 000026FB 26803F0F                         cmp     byte [es:bx], 0Fh ; Extended DOS LBA
9442 000026FF 740A                             je      short fmpgot
9443
9444 00002701 83C310                           add     bx, 16
9445 00002704 81FB0204                         cmp     bx, 402h        ; 202h+bootbias
9446 00002708 75EB                             jnz     short fmpnext
9447                                            ;jmp    short fmpnextfound ; extended partition     not found
9448                                            ; 18/12/2022
9449                                    fmpnextfound:
9450 0000270A C3                               retn
9451
9452                                    ;       ; 30/07/2019 - Retro DOS v3.2
9453                                    ;       jb      short fmpnext
9454                                    ;fmpret:
9455                                    ;       retn    ; 29/05/2019
9456
9457                                    ; ---------------------------------------------------------------------------
9458
9459                                            ; 19/10/2022
9460                                    fmpgot:                                 ; found my partition.
9461 0000270B E82B01                           call    dmax_check      ; check for drvmax already 26
9462 0000270E 73FA                             jnb     short fmpnextfound ; done if too many
9463
9464 00002710 8B3E[621A]                       mov     di, [end_of_bdss] ; get next free bds
9465
9466                                            ; 19/12/2023
9467                                            ;mov    word [di+47h], 1 ; [di+BDS.bdsm_ismini]
9468                                            ;; 10/12/2022
9469                                            ;or     byte [di+23h], 1
9470                                            ;;or    word [di+23h], 1 ; [di+BDS.flags]
9471                                            ;                       ; fNon_Removable
9472                                            ;mov    byte [di+22h], 5 ; [di+BDS.formfactor]
9473                                            ;                       ; ffHardFile
9474                                            ; 19/12/2023 - Retro DOS v5.0
9475 00002714 C745790100                       mov     word [di+79h], 1 ; [di+BDS.bdsm_ismini]
9476 00002719 804D3F01                         or      byte [di+3Fh], 1 ; [di+BDS.flags], fNon_Removable
9477 0000271D C6453E05                         mov     byte [di+3Eh], 5 ; [di+BDS.formfactor], ffHardFile
9478
9479 00002721 C606[081A]00                     mov     byte [fbigfat], 0 ; assume 12 bit fat.
9480 00002726 A1[641A]                         mov     ax, [mini_hdlim]
9481 00002729 894515                           mov     [di+15h], ax    ; [di+BDS.heads]
9482 0000272C A1[661A]                         mov     ax, [mini_seclim]
9483 0000272F 894513                           mov     [di+13h], ax    ; [di+BDS.secpertrack]
9484 00002732 A0[5E1A]                         mov     al, [rom_minidisk_num]
9485 00002735 884504                           mov     [di+4],al       ; [di+BDS.drivenum]
9486                                                                   ; set physical number
9487 00002738 A0[7500]                         mov     al, [drvmax]
9488 0000273B 884505                           mov     [di+5],al       ; [di+BDS.drivelet]
9489                                                                   ; set logical number
9490 0000273E 26837F0A00                       cmp     word [es:bx+10], 0
9491                                            ;ja     short fmpgot_cont
9492 00002743 7707                             ja      short fmpgot1  ; 19/12/2023
9493 00002745 26837F0840                       cmp     word [es:bx+8], 64 ; with current bpb,
9494                                                                   ; only lower word is meaningful.
9495 0000274A 72BE                             jb      short fmpnextfound
9496                                                                   ; should be bigger than 64 sectors at least
9497                                    fmpgot1:    ; 19/12/2023
9498                                    ;fmpgot_cont:
9499 0000274C 83EB04                           sub     bx, 4           ; let bx point to the start of the entry
9500 0000274F 268A7702                         mov     dh, [es:bx+2]  ; cylinder
9501 00002753 80E6C0                           and     dh, 0C0h        ; get higher bits of cyl
9502 00002756 D0C6                             rol     dh, 1
9503 00002758 D0C6                             rol     dh, 1
9504 0000275A 268A5703                         mov     dl, [es:bx+3]  ; cyl byte
9505                                            ; 19/12/2023 - Retro DOS v5.0
9506 0000275E 89557B                           mov     [di+7Bh], dx    ; [di+BDS.bdsm_hidden_trks]
9507                                            ;mov    [di+49h], dx    ; [di+BDS.bdsm_hidden_trks]
9508                                                                   ; set hidden trks
9509                                            ; 19/12/2023
9510                                            ;push   bx ; * ; PCDOS 7.1
9511                                            ;;;
9512 00002761 268B4F08                         mov     cx, [es:bx+8]  ; partition size, lw
9513 00002765 268B470A                         mov     ax, [es:bx+10] ; partition size, hw
9514 00002769 030E[0222]                       add     cx, [ep_start_sector]
9515 0000276D 1306[0422]                       adc     ax, [ep_start_sector+2]
9516 00002771 31D2                             xor     dx, dx ; 19/12/2023
9517 00002773 3916[0222]                       cmp     [ep_start_sector], dx ; 0
9518                                            ;cmp    word [ep_start_sector], 0
9519 00002777 750D                             jnz     short fmpgot2
9520 00002779 3916[0422]                       cmp     [ep_start_sector+2], dx ; 0
9521                                            ;cmp    word [ep_start_sector+2], 0
9522 0000277D 7507                             jnz     short fmpgot2
9523 0000277F 890E[0222]                       mov     [ep_start_sector], cx
9524 00002783 A3[0422]                         mov     [ep_start_sector+2], ax
9525                                    fmpgot2:
9526 00002786 890E[0622]                       mov     [ep_hidden_secs], cx
9527 0000278A A3[0822]                         mov     [ep_hidden_secs+2], ax
9528
9529                                            ; convert start sector address to CHS
9530
9531                                            ; 19/12/2023
9532                                            ; dx = 0
9533                                            ;push   bx ; * ; not necessary
9534
9535                                            ;mov    bx, [di+13h]   ; [di+BDS.secpertrack]
9536 0000278D 8B7513                           mov     si, [di+13h]   ; [di+BDS.secpertrack]
9537                                            ;xor    dx, dx  ; dx = 0
9538                                            ;div    bx
9539 00002790 F7F6                             div     si
9540 00002792 91                               xchg    ax, cx
9541                                            ;div    bx
9542 00002793 F7F6                             div     si
9543                                            ;mov    bx, [di+15h]   ; [di+BDS.heads]
9544                                            ; 07/05/2024
9545                                            ; 17/04/2024 (BugFix)
9546 00002795 8B7515                           mov     si, [di+15h]   ; [di+BDS.heads]
9547 00002798 91                               xchg    ax, cx
9548 00002799 31D2                             xor     dx, dx
9549                                            ;div    bx
9550 0000279B F7F6                             div     si
9551 0000279D 91                               xchg    ax, cx
9552                                            ;div    bx
9553 0000279E F7F6                             div     si
9554
9555                                            ;pop    bx ; *
9556
9557 000027A0 09C9                             or      cx, cx
```

```
9558 000027A2 7505                              jnz     short fmpgot_lba_rd
9559 000027A4 3D0004                            cmp     ax, 1024        ; cylinder number < 1024, CHS read is proper
9560 000027A7 7235                              jb      short fmpgot_chs_rd
9561                                    fmpgot_lba_rd:
9562 000027A9 804D4004                          or      byte [di+40h], 4 ; set fLBArw flag ; LBA read/write ok/ready
9563 000027AD 8A16[5E1A]                        mov     dl, [rom_minidisk_num]
9564 000027B1 1E                                push    ds
9565                                            ; 19/12/2023
9566                                            ;push   si ; ** ; not necessary
9567 000027B2 31C0                              xor     ax, ax          ; push bp
9568                                                                    ; mov bp, sp ; (*)
9569 000027B4 50                                push    ax ; 0
9570 000027B5 50                                push    ax ; 0
9571 000027B6 FF36[0822]                        push    word [ep_hidden_secs+2]
9572 000027BA FF36[0622]                        push    word [ep_hidden_secs]
9573 000027BE B80002                            mov     ax, bootbias ; 200h
9574                                            ;mov    ax, 200h        ; bootbias (buffer offset)
9575 000027C1 06                                push    es              ; buffer segment
9576 000027C2 50                                push    ax
9577 000027C3 B80100                            mov     ax, 1
9578 000027C6 50                                push    ax              ; read count
9579 000027C7 B81000                            mov     ax, 10h         ; DAP size = 16
9580 000027CA 50                                push    ax
9581 000027CB 8CD0                              mov     ax, ss
9582 000027CD 8ED8                              mov     ds, ax
9583 000027CF 89E6                              mov     si, sp          ; ds:si = Disk Address Packet
9584
9585 000027D1 B442                              mov     ah, 42h         ; LBA read
9586 000027D3 CD13                              int     13h             ; DISK - IBM/MS Extension
9587                                                                    ; EXTENDED READ (DL - drive, DS:SI - disk address packet)
9588                                            ; 19/12/2023
9589                                            ;pushf          ; PCDOS 7.1 IBMBIO.COM BUG! Erdogan Tan - 08/08/2023
9590                                            ;add    sp, 16
9591                                            ;popf           ; BUG!
9592                                                                    ; mov sp, bp ; (*)
9593                                                                    ; pop bp
9594                                            ; 19/12/2023
9595 000027D5 9F                                lahf            ; load status flags into AH
9596 000027D6 83C410                            add     sp, 16
9597 000027D9 9E                                sahf            ; store AH into flags
9598
9599                                            ;pop    si ; ** ; 19/12/2023
9600 000027DA 1F                                pop     ds
9601 000027DB 7317                              jnc     short fmpgot3
9602                                    fmpnotfound:        ; 19/12/2023
9603 000027DD C3                                retn
9604                                            ;jmp    short fmpgot3
9605                                            ;;;
9606
9607                                            ; 19/12/2023
9608                                    fmpgot_chs_rd:
9609 000027DE 268B4F02                          mov     cx, [es:bx+2]  ; cylinder,cylinder/sector
9610 000027E2 268A7701                          mov     dh, [es:bx+1]  ; head
9611 000027E6 8A16[5E1A]                        mov     dl, [rom_minidisk_num]
9612 000027EA BB0002                            mov     bx, 200h        ; bootbias
9613 000027ED B80102                            mov     ax, 201h
9614 000027F0 CD13                              int     13h             ; DISK - READ SECTORS INTO MEMORY
9615                                                                    ; AL = number of sectors to read, CH = track, CL = sector
9616                                                                    ; DH = head, DL       = drive, ES:BX -> buffer to fill
9617                                                                    ; Return: CF set on error, AH =      status, AL = number of sectors read
9618                                    ;fmpgot3:   ; 19/12/2023
9619                                            ;jc     short fmpnextfound
9620 000027F2 72E9                              jc      short fmpnotfound
9621                                    fmpgot3:
9622 000027F4 BBC203                            mov     bx, 3C2h        ; 1C2h+bootbias
9623
9624                                            ; 08/08/2023 - Retro DOS v4.2 IO.SYS (optimization)
9625                                            ; PCDOS 7.1 IBMBIO.COM - BIOSDATA:2C7Ch
9626 000027F7 26817F3C55AA                      cmp     word [es:bx+3Ch], 0AA55h ; 03C2h+03Ch = 3FEh
9627                                            ;jne    short fmpnextfound ; not a valid boot sector !
9628                                            ; 19/12/2023
9629 000027FD 75DE                              jne     short fmpnotfound ; not a valid boot sector !
9630
9631                                            ; 13/08/2023
9632                                            ;push   es
9633 000027FF E80800                            call    setmini         ; install a mini disk.
9634                                                                    ; bx value saved.
9635                                            ;pop    es  ; 13/08/2023
9636 00002802 7203                              jc      short fmpnextchain
9637 00002804 E84700                            call    xinstall_bds    ; -- install the bdsm into table
9638                                    fmpnextchain:
9639 00002807 E9EBFE                            jmp     fmpnext         ; let's find out
9640                                                                    ; if we have any chained partition
9641                                    ; ---------------------------------------------------------------------------
9642
9643                                            ; 18/12/2022
9644                                    ;fmpnextfound:
9645                                            ;retn
9646
9647                                    ; =============== S U B   R O U T I N E =======================================
9648
9649                                    ; 15/10/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 IO.SYS)
9650                                    ; 28/12/2018 - Retro DOS v4.0 (MSDOS 6.21)
9651
9652                                    ; 19/12/2022 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMBIO.COM)
9653                                    ; (PCDOS 7.1 IBMBIO.COM - BIOSDATA:2C92h)
9654
9655                                    setmini:    ; 'setmini' is called from 'find_mini_partition' procedure
9656
9657 0000280A 57                                  push    di
9658 0000280B 53                                push    bx
9659                                            ; 12/08/2023 - Retro DOS v4.2 IO.SYS (optimization)
9660                                            ; ds = cs = BIOSDATA segment
9661                                            ;push   ds
9662 0000280C 06                                push    es
9663                                    setmini_1:
9664                                            ;cmp    byte [es:bx], 1 ; FAT12 partition
9665                                            ;je     short setmini_2
9666                                            ;cmp    byte [es:bx], 4 ; FAT16 (CHS) partition
9667                                            ;je     short setmini_2
9668                                            ;cmp    byte [es:bx], 6 ; FAT16 BIG (CHS) partition
9669                                            ;je     short setmini_2
9670                                            ;
9671                                            ; 19/12/2023 - Retro DOS v5.0
9672                                            ;cmp    byte [es:bx], 0Bh ; FAT32 (CHS) partition
9673                                            ;je     short setmini_2
9674                                            ;cmp    byte [es:bx], 0Ch ; FAT32 (LBA) partition
9675                                            ;je     short setmini_2
9676                                            ;cmp    byte [es:bx], 0Eh ; FAT16 (LBA) partition
9677                                            ;je     short setmini_2
9678
9679                                            ; 19/12/2023
9680 0000280D 268A07                            mov     al, [es:bx]
9681 00002810 3C01                              cmp     al, 1           ; FAT12 partition
```

```
9682 00002812 7422                              je      short setmini_2
9683 00002814 3C04                              cmp     al, 4           ; FAT16 (CHS) partition
9684 00002816 741E                              je      short setmini_2
9685 00002818 3C06                              cmp     al, 6           ; FAT16 BIG (CHS) partition
9686 0000281A 741A                              je      short setmini_2
9687 0000281C 3C0B                              cmp     al, 0Bh         ; FAT32 (CHS) partition
9688 0000281E 7416                              je      short setmini_2
9689 00002820 3C0C                              cmp     al, 0Ch         ; FAT32 (LBA) partition
9690 00002822 7412                              je      short setmini_2
9691 00002824 3C0E                              cmp     al, 0Eh         ; FAT16 (LBA) partition
9692 00002826 740E                              je      short setmini_2
9693
9694 00002828 83C310                            add     bx, 16
9695 0000282B 81FB0204                          cmp     bx, 402h        ; 202h+bootbias
9696                                            ;jne    short setmini_1
9697 0000282F 72DC                              jb      short setmini_1 ; 19/12/2023
9698 00002831 F9                                stc
9699 00002832 07                                pop     es
9700                                            ; 12/08/2023
9701                                            ;pop    ds
9702 00002833 5B                                pop     bx
9703 00002834 5F                                pop     di
9704 00002835 C3                                retn
9705
9706                                    ; ---------------------------------------------------------------------------
9707                                    setmini_2:
9708 00002836 E9D1F9                            jmp     set2            ; branch into middle of sethard
9709
9710                                    ; =============== S U B   R O U TI N E =======================================
9711
9712                                    ; 30/12/2022 - Retro DOS v4.2
9713                                    ; (SYSINITSEG is 473h for MSDOS 6.21 IO.SYS)
9714
9715                                    ; 15/10/2022
9716                                    ; 28/12/2018 - Retro DOS v4.0
9717                                    ;
9718                                    ; dmax_check -- call this when we want to install a new drive.
9719                                    ;           it checks for drvmax < 26 to see if there is
9720                                    ;           a drive letter left.
9721                                    ;
9722                                    ;   drvmax < 26 : carry SET!
9723                                    ;   drvmax >=26 : carry RESET!, error flag set for message later
9724                                    ;                 trash ax
9725
9726                                            ; 19/12/2023 - Retro DOS v5.0
9727                                    dmax_check:
9728 00002839 803E[7500]1A                      cmp     byte [drvmax], 26 ; checks for drvmax < 26
9729 0000283E 720D                              jb      short dmax_ok  ; return with carry if okay
9730 00002840 06                                push    es
9731                                            ;;mov   ax, 46Dh        ; SYSINIT_SEG (SYSINIT segment)
9732                                            ;mov    ax, 544h        ; 19/12/2023 (PCDOS 7.1)
9733 00002841 B8D904                            mov     ax, SYSINITSEG ; 17/10/2022
9734 00002844 8EC0                              mov     es, ax
9735                                            ; 18/10/2022
9736 00002846 26C606[8803]01                    mov     byte [es:TOOMANYDRIVESFLAG], 1 ; 09/12/2022
9737                                            ;mov    byte ptr es:3FFh, 1 ; [es:toomanydrivesflag]
9738                                                                    ; set message flag
9739                                                                    ; [SYSINIT+toomanydrivesflag]
9740 0000284C 07                                pop     es
9741
9742                                            ;;push  es
9743                                            ;;mov   ax,SYSINIT_SEG
9744                                            ;;mov   es,ax
9745                                            ;;mov   byte [es:toomanydrivesflag],1
9746                                                                    ; set message flag
9747                                            ;;pop   es
9748                                            ;
9749                                            ;mov    byte [SYSINIT+toomanydrivesflag],1
9750                                    dmax_ok:
9751 0000284D C3                                retn
9752
9753                                    ; =============== S U B   R O U TI N E =======================================
9754
9755                                    ; 18/10/2022
9756                                    ; 15/10/2022
9757                                    ; 28/12/2018 - Retro DOS v4.0
9758                                    ;
9759                                    ;   link next bds (at ds:di) into the chain. assume that the
9760                                    ;     chain is entirely within ds == datagrp. also update drvmax,
9761                                    ;     dskdrv_table, and end_of_bdss.
9762
9763                                            ; 19/12/2023 - Retro DOS v5.0
9764                                            ; (PCDOS 7.1 IBMBIO.COM - BIOSDATA:2CE1h)
9765                                    xinstall_bds:
9766 0000284E 56                                push    si
9767 0000284F 53                                push    bx
9768 00002850 8B36[1901]                        mov     si, [start_bds]       ; get first bds
9769                                    xinstall_bds_1:
9770 00002854 833CFF                            cmp     word [si], 0FFFFh ; is this the last one?
9771 00002857 7404                              jz      short xinstall_bds_2 ;skip ahead if so
9772                                            ;mov    si, [si+BDS.link]
9773 00002859 8B34                              mov     si, [si]        ; chain through list
9774 0000285B EBF7                              jmp     short xinstall_bds_1
9775
9776                                    xinstall_bds_2:
9777                                            ;mov    [si+BDS.link], di
9778 0000285D 893C                              mov     [si], di
9779                                            ;mov    [si+BDS.link+2], ds
9780 0000285F 8C5C02                            mov     [si+2], ds
9781                                            ;mov    word [di+BDS.link], -1
9782 00002862 C705FFFF                          mov     word [di], 0FFFFh ; make sure it is a null ptr.
9783                                            ;mov    [di+BDS.link+2], ds
9784 00002866 8C5D02                            mov     [di+2], ds ; might as well plug segment
9785                                            ; 20/03/2019 - Retro DOS v4.0
9786                                            ;lea    bx, [di+BDS.BPB]
9787 00002869 8D5D06                            lea     bx, [di+6]
9788 0000286C 8B36[601A]                        mov     si, [last_dskdrv_table]
9789 00002870 891C                              mov     [si], bx
9790 00002872 8306[601A]02                      add     word [last_dskdrv_table], 2
9791 00002877 FE06[7500]                        inc     byte [drvmax]
9792                                            ;add    word [end_of_bdss], 100 ; BDS.size = 100
9793                                            ; 19/12/2023 - Retro DOS v5.0
9794 0000287B 8106[621A]9600                    add     word [end_of_bdss], 150 ; BDS.size = 150
9795 00002881 5B                                pop     bx
9796 00002882 5E                                pop     si
9797 00002883 C3                                retn
9798
9799                                    ; =============== S U B   R O U TI N E =======================================
9800
9801                                    ; 17/10/2022
9802                                    ; 15/10/2022
9803                                    ; 28/12/2018 - Retro DOS v4.0
9804                                    ; 03/06/2018 - Retro DOS v3.0
9805
```

```
                                        ; 19/12/2023 - Retro DOS v5.0
                                cmos_clock_read:
00002884 50                             push    ax
00002885 51                             push    cx
00002886 52                             push    dx
00002887 55                             push    bp
00002888 31ED                           xor     bp, bp
                                loop_clock:
0000288A 31C9                           xor     cx, cx
0000288C 31D2                           xor     dx, dx
0000288E B402                           mov     ah, 2
00002890 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
                                        ; 19/12/2023
                                        ;cmp    cx, 0
00002892 21C9                           and     cx, cx
00002894 750F                           jnz     short clock_present
                                        ;cmp    dx, 0
00002896 09D2                           or      dx, dx
00002898 750B                           jnz     short clock_present
                                        ;cmp    bp, 1           ; read again after a slight delay, in case clock
                                        ;je     short no_readdate ; was       at zero setting.
0000289A 21ED                           and     bp, bp
0000289C 751A                           jnz     short no_readdate
0000289E 45                             inc     bp              ; only perform delay once.
                                        ;mov    cx, 4000h       ; 16384
                                        ; 19/12/2023
0000289F B540                           mov     ch, 40h ; cx = 4000h ; 16384
                                delay:
000028A1 E2FE                           loop    delay
000028A3 EBE5                           jmp     short loop_clock
                                ; ---------------------------------------------------------------------------

                                clock_present:
                                        ;mov    byte [cs:havecmosclock], 1 ; set the flag for cmos clock
                                        ; 19/12/2023
                                        ; ds = cs
000028A5 C606[8C04]01                   mov     byte [havecmosclock], 1 ; set the flag for cmos clock

000028AA E81000                         call    cmosck          ; reset cmos clock rate       that may be
                                                                ; possibly destroyed by       cp dos and
                                                                ; post routine did not restore that.
000028AD 56                             push    si
000028AE E8BAEF                         call    read_real_date ; read real-time clock for date
000028B1 FA                             cli
                                        ;mov    ds:daycnt, si  ; set system date
000028B2 8936[8904]                     mov     [daycnt], si
000028B6 FB                             sti
000028B7 5E                             pop     si
                                no_readdate:
000028B8 5D                             pop     bp
000028B9 5A                             pop     dx
000028BA 59                             pop     cx
000028BB 58                             pop     ax

                                cmosck9:    ; 19/12/2023
000028BC C3                             retn

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

                                ; the following code is written by jack gulley in engineering group.
                                ; cp dos (CP/DOS, OS/2) is changing cmos clock rate for its own purposes
                                ; and if the use cold boot the system to use pc dos while running cp dos,
                                ; the cmos clock rate are still slow which slow down disk operations
                                ; of pc dos which uses cmos clock. pc dos is put this code in msinit
                                ; to fix this problem at the request of cp dos.
                                ;
                                ; the program is modified to be run on msinit. equates are defined
                                ; in cmosequ.inc. this program will be called by cmos_clock_read procedure.
                                ;
                                ;  the following code cmosck is used to insure that the cmos has not
                                ;   had its rate controls left in an invalid state on older at's.
                                ;
                                ;   it checks for an at model byte "fc" with a submodel type of
                                ;   00, 01, 02, 03 or 06 and resets the periodic interrupt rate
                                ;   bits in case post has not done it. this initilization routine
                                ;   is only needed once when dos loads. it should be run as soon
                                ;   as possible to prevent slow diskette access.
                                ;
                                ;   this code exposes one to dos clearing cmos setup done by a
                                ;   resident program that hides and re-boots the system.

                                cmosck:                                 ; check and reset rtc rate bits

                                ;model byte and submodel byte were already determined in msinit.

                                    ; 16/06/2018 - Retro DOS v3.0
                                    ; 19/03/2018 (Model: 0FCh, Sub Model: 01h, REF: AMIBIOS Prog. Guide)

                                    ; 19/12/2023 - Retro DOS v5.0

                                        ; 19/12/2023
                                        ; ds = cs
                                        ;push  ax ; not necessary ; 19/12/2023
                                        ;
000028BD 803E[AF05]FC                   cmp     byte [model_byte], 0FCh
                                        ;cmp    byte [cs:model_byte], 0FCh
000028C2 75F8                           jnz     short cmosck9  ; Exit if not an AT model
000028C4 803E[B005]06                   cmp     byte [secondary_model_byte], 6 ; 21/04/2024
                                        ;cmp    byte [cs:secondary_model_byte], 6
                                                                ; Is it 06 for the industral AT ?
000028C9 7407                           jz      short cmosck4  ; Go reset CMOS periodic rate if 06
000028CB 803E[B005]04                   cmp     byte [secondary_model_byte], 4
                                        ;cmp    byte [cs:secondary_model_byte], 4
                                                                ; Is it 00, 01, 02, or 03 ?
000028D0 73EA                           jnb     short cmosck9  ; EXIT if problem fixed by POST
                                                                ; Also,Secondary_model_byte = 0
                                                                ;   when AH=0C0h, int 15h failed.
                                                                ;       RESET THE CMOS PERIODIC RATE
                                                                ;  Model=FC submodel=00,01,02,03 or 06
                                cmosck4:
000028D2 B08A                           mov     al, 8Ah         ; cmos_reg_a|nmi
                                                                ; NMI disabled on return
000028D4 B426                           mov     ah, 26h         ; 00100110b
                                                                ; Set divider & rate selection
000028D6 E80B00                         call    cmos_write
000028D9 B08B                           mov     al, 8Bh         ; cmos_reg_b|nmi
                                                                ; NMI disabled on return
000028DB E82000                         call    cmos_read
000028DE 2407                           and     al, 7           ; 00000111b
                                                                ; clear SET,PIE,AIE,UIE,SQWE
000028E0 88C4                           mov     ah, al
```

```
000028E2 B00B                           mov     al, 0Bh         ; cmos_reg_b
                                                                ; 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)
000028E4 9C                             pushf                   ;
000028E5 50                             push    ax              ; save work register values
000028E6 FA                             cli
000028E7 50                             push    ax              ; save user nmi       state
000028E8 0C80                           or      al, 80h         ; disable nmi for us
000028EA E670                           out     70h, al         ; CMOS Memory/RTC Index       Register:
                                                                ; RTC Seconds
000028EC 90                             nop
000028ED 88E0                           mov     al, ah
000028EF E671                           out     71h, al         ; CMOS Memory/RTC Data Register
000028F1 58                             pop     ax              ; get user nmi
000028F2 2480                           and     al, 80h
000028F4 0C0F                           or      al, 0Fh
000028F6 E670                           out     70h, al         ; CMOS Memory/RTC Index       Register:
                                                                ; RTC Seconds
000028F8 90                             nop
000028F9 E471                           in      al, 71h         ; CMOS Memory/RTC Data Register
000028FB 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
000028FC 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)
000028FE 9C                             pushf
000028FF FA                             cli
00002900 53                             push    bx
                                        ;push   ax ; *          ; AL = cmos table address to be read
                                        ; 19/12/2023
00002901 89C3                           mov     bx, ax ; * ; input
00002903 0C80                           or      al, 80h
00002905 E670                           out     70h, al         ; CMOS Memory/RTC Index       Register:
                                                                ; RTC Seconds
00002907 90                             nop                     ; (undocumented delay needed)
00002908 E471                           in      al, 71h         ; CMOS Memory/RTC Data Register

                                        ;mov    bx, ax ; output
                                        ;pop    ax ; * ; input

                                        ; 19/12/2023
                                        ; al = output, bl = input
0000290A 93                             xchg    ax, bx ; *
                                        ; bl = output, al = input

0000290B 2480                           and     al, 80h
0000290D 0C0F                           or      al, 0Fh
0000290F E670                           out     70h, al         ; CMOS Memory/RTC Index       Register:
                                                                ; RTC Seconds
00002911 90                             nop
00002912 E471                           in      al, 71h         ; CMOS Memory/RTC Data Register
                                        ;mov    ax, bx ; * ; output
                                        ; 19/12/2023
00002914 93                             xchg    ax, bx
00002915 5B                             pop     bx

                                        ; 19/12/2023
                                cmos_rw_popf:
00002916 0E                             push    cs              ; *place code segment in stack and
00002917 E80100                         call    cmos_popf       ; *handle popf for b- level 80286
0000291A C3                             retn                    ; return with flags restored

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

                                cmos_popf:
0000291B CF                             iret                    ; popf for level b- parts
                                                                ; return far and restore flags

                                ; 21/12/2022
                                ; -------------------------------------------------------------------------------
                                ; -------------------------------------------------------------------------------
                                %if 0

                                ; -------------------------------------------------------------------------------
                                ; MSINIT.ASM (MSDOS 6.0, 1991)
                                ; -------------------------------------------------------------------------------
```

```
10054                                   ; The following routines provide support for reading in the file MSDOS.SYS.
10055                                   ; ---------------------------------------------------------------------------
10056                                   
10057                                   ; 15/10/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 IO.SYS)
10058                                   ;
10059                                   ; (For Retro DOS, 'IO.SYS' and 'MSDOS.SYS' are already loaded together
10060                                   ;  at once -as single kernel file- by the Retro DOS boot sector code.
10061                                   ;  So, following disk reads -MSDOS.SYS loading- is not needed!
10062                                   ;  Only needing is to move MSDOS Kernel to it's final memory location.)
10063                                   
10064                                   ; ============== S U B   R O U T I N E =======================================
10065                                   
10066                                   ; GetClus, read in a cluster at a specified address
10067                                   ;
10068                                   ;  bx = cluster to read
10069                                   ;  cx = sectors per cluster
10070                                   ;  es:di = load location
10071                                   
10072                                   ; 17/10/2022
10073                                   ;DISKRD equ diskrd - DOSBIOSEG_2C7h      ; (8E5h for MSDOS 5.0 IO.SYS)
10074                                   ; 09/12/2022
10075                                   DISKRD equ diskrd
10076                                   
10077                                               ; 29/12/2023
10078                                               ; 20/12/2023 - Retro DOS v5.0
10079                                               ; (PCDOS 7.1 IBMBIO.COM - BIOSDATA:2DC4h)
10080                                   
10081                                               ; si:bx = (32 bit) cluster to read
10082                                               ; cx = sectors per cluster
10083                                               ; es:di = load location
10084                                   
10085                                               ; 17/10/2022
10086                                   getclus:
10087                                               ; 12/12/2023
10088                                               ; ds = cs
10089                                   
10090                                               push    cx ; 1*
10091                                               push    di ; 2*
10092                                               ;mov    [cs:doscnt], cx
10093                                               mov     [doscnt], cx ; 12/12/2023
10094                                   
10095                                               ; 20/12/2023
10096                                               ;;mov   [cs:ClusterH], si ; high word of cluster number
10097                                               ;mov    [ClusterH], si ; high word of cluster number
10098                                               mov     bp, si
10099                                   
10100                                               mov     ax, bx
10101                                   
10102                                               ;dec    ax
10103                                               ;dec    ax
10104                                               ; 20/12/2023
10105                                               sub     ax, 2
10106                                   
10107                                               ;;sbb   [cs:ClusterH], 0
10108                                               ;sbb    [ClusterH], 0
10109                                               sbb     bp, 0
10110                                   
10111                                               ; 20/12/2023
10112                                               ;;xchg  ax, [cs:ClusterH]
10113                                               ;xchg   ax, [ClusterH]
10114                                               xchg    ax, bp
10115                                   
10116                                               mul     cx
10117                                   
10118                                               ;;xchg  ax, [cs:ClusterH]
10119                                               ;xchg   ax, [ClusterH]
10120                                               xchg    ax, bp ; (+)
10121                                               ;
10122                                               mul     cx              ;; convert to logical sector
10123                                                                       ;; dx:ax = matching logical sector number
10124                                                                       ;;       starting from the data sector
10125                                               ;;add   ax, [cs:bios_l]
10126                                               ;;adc   dx, [cs:bios_h]       ; dx:ax = first logical sector to read
10127                                               ; 12/12/2023
10128                                               ;add    ax, [bios_l]
10129                                               ;adc    dx, [bios_h]   ; dx:ax = first logical sector to read
10130                                   
10131                                               ; 20/12/2023
10132                                               ;;add   dx, [cs:ClusterH]
10133                                               ;add    ax, [cs:First_Data_Sector]
10134                                               ;adc    dx, [cs:First_Data_Sector+2]
10135                                               add     dx, bp ; (+)
10136                                               ;add    dx, [ClusterH] ; convert to logical sector
10137                                                                       ; dx:ax = matching logical sector number
10138                                                                       ;       starting from the data sector
10139                                               add     ax, [First_Data_Sector]
10140                                               adc     dx, [First_Data_Sector+2]
10141                                                                       ; dx:ax = first logical sector to read
10142                                   unpack:
10143                                               ; 20/12/2023
10144                                               push    ds ; 3* ; ds = cs ; 12/12/2023
10145                                               push    dx ; 4* ; * ; 12/12/2023
10146                                               push    ax ; 5*
10147                                               ; 29/12/2023
10148                                               push    si ; 6*
10149                                               push    bx ; 7*
10150                                   
10151                                               ;;mov   si, [cs:fatloc]
10152                                               ;mov    si, [fatloc] ; 12/12/2023
10153                                               ;mov    ds, si
10154                                               ; 20/12/2023
10155                                               ;mov    ax, [fatloc]
10156                                               ;mov    ds, ax
10157                                               push    bx ; 8*
10158                                               push    word [fatloc] ; 9*
10159                                   
10160                                               ;test   byte [cs:fbigfat], 20h
10161                                               test    byte [fbigfat], 20h   ; fbigbig FAT32 ?
10162                                               pop     ds ; 9* ; ds = [fatloc]
10163                                               jz      short not_32bit_cluster ; no
10164                                   unpack32:
10165                                               ;push   dx
10166                                               mov     dx, si
10167                                               ;mov    si, bx
10168                                               pop     si ; 8* ; si = bx
10169                                               add     si, si
10170                                               adc     dx, dx
10171                                               add     si, si
10172                                               adc     dx, dx
10173                                                       ; dx:si = 4*(si:bx) ; clust num offset from FAT entry 0
10174                                               call    get_fat_sector
10175                                               mov     si, [bx+2]      ; high word of the FAT32 cluster number
10176                                               mov     bx, [bx]        ; low word of the FAT32 cluster number
10177                                               ;pop    dx
```

```
10178                                   jmp     short getcl1
10179
10180                           not_32bit_cluster:
10181                                   ;mov    si, bx          ; next cluster
10182                                   pop     si ; 8* ; si = bx
10183                                   test    byte [cs:fbigfat], 40h; fbig
10184                                                           ; 16 bit fat?
10185                                   jnz     short unpack16 ; yes
10186                           unpack12:
10187                                   shr     si, 1           ; 12 bit fat. si = si/2
10188                                                           ; si = clus + clus/2
10189                                   add     si, bx          ;
10190                                                           ; (si = byte offset of the cluster in the FAT)
10191                                   ;push   dx ; 12/12/2023
10192                                   xor     dx, dx
10193                                   ; 12/12/2023
10194                                   ; ds = FAT buffer segment
10195                                   call    get_fat_sector
10196                                   ;pop    dx ; 12/12/2023
10197
10198                                   mov     ax, [bx]        ; save it into ax
10199                                   jnz     short even_odd ; if not a splitted fat, check even-odd.
10200                                   ; 25/06/2023
10201                                   ;mov    al, [bx]        ; splitted fat
10202
10203                                   ; 12/12/2023
10204                                   ;mov    [cs:temp_cluster], al
10205                                   push    ax ; ** ; al = low 8 bits of 12 bits cluster number
10206
10207                                   inc     si              ; (next byte)
10208
10209                                   ;push   dx ; 12/12/2023
10210                                   xor     dx, dx
10211                                   call    get_fat_sector
10212                                   ;pop    dx ; 12/12/2023
10213
10214                                   ;mov    al, ds:0
10215                                   ; 12/12/2023
10216                                   ; ds = FAT buffer segment
10217                                   ;mov    al, [0] ; 19/10/2022
10218                                   ;mov    [cs:temp_cluster+1], al
10219                                   ;mov    ax, [cs:temp_cluster]
10220                                   ; 12/12/2023
10221                                   ;mov    al, [cs:temp_cluster]
10222                                   pop     ax  ; ** ; al = low 8 bits of 12 bits cluster number
10223                                   mov     ah, [0] ; high 4 bits (bits 7 to 11) of 12 bits cluster num
10224                           even_odd:
10225                                   ; 29/12/2023
10226                                   pop     bx ; 7*         ; restore old fat entry       value
10227                                   push    bx              ; save it right       away.
10228                                   shr     bx, 1           ; was it even or odd?
10229                                   jnc     short havclus  ; it was even.
10230                                   shr     ax, 1           ; odd. massage fat value and keep
10231                                                           ; the highest 12 bits.
10232                                   shr     ax, 1
10233                                   shr     ax, 1
10234                                   shr     ax, 1
10235                           havclus:
10236                                   mov     bx, ax          ; now bx = new fat entry.
10237                                   and     bx, 0FFFh       ; keep low 12 bits.
10238                                   jmp     short unpackx
10239                           ; ---------------------------------------------------------------------------
10240
10241                           unpack16:
10242                                   ;push   dx      ; 12/12/2023
10243                                   xor     dx, dx ; 0
10244                                   shl     si, 1           ; extend to 32 bit offset
10245                                   ;adc    dx, 0
10246                                   ; 12/12/2023
10247                                   rcl     dx, 1
10248
10249                                   ; 12/12/2023
10250                                   ; ds = FAT buffer segment
10251                                   call    get_fat_sector
10252                                   ;pop    dx      ; 12/12/2023
10253                                   mov     bx, [bx]        ;
10254                                                           ; bx = new fat entry.
10255                           unpackx:
10256                                   ; 20/12/2023
10257                                   xor     si, si          ; high word of cluster number = 0
10258                                                           ; (FAT12 or FAT16)
10259                           getcl1:
10260                                   ; 29/12/2023
10261                                   pop     ax      ; 7* - cluster number lw
10262                                   ;pop    word [cs:ClusterH]
10263                                   pop     dx      ; 6* - cluster number hw
10264
10265                                   ; 20/12/2023 - Retro DOS v5.0 (PCDOS 7.1 IBMBIO.COM)
10266                                   ; (this is a fast kernel loading method by the MSDOS programmer)
10267                                   ; ((consequtive clusters --> consequtive sectors))
10268
10269                                   sub     ax, bx  ; previous - current (or current - new)
10270                                   ;sbb    [cs:ClusterH], si
10271                                   sbb     dx, si
10272                                   ;;cmp   [cs:ClusterH], -1 ; one apart? (current = previous+1)
10273                                   ;cmp    dx, -1
10274                                   ; 29/12/2023
10275                                   inc     dx ; -1 -> 0
10276                                   jnz     short not_consequental
10277                                   ;cmp    ax, -1          ; 0FFFFh ; is [ClusterH]:ax = -1 ?
10278                                   inc     ax ; -1 -> 0
10279                           not_consequental:
10280                                   pop     ax ; 5*         ; restore logical sector (low)
10281                                   pop     dx ; 4* ; * ; 12/12/2023
10282                                   pop     ds ; 3*
10283
10284                                   ;; 12/12/2023
10285                                   ;; (this is a fast kernel loading method by the MSDOS programmer)
10286                                   ;; ((consequtive clusters --> consequtive sectors))
10287                                   ;; ds = cs
10288                                   ;sub    si, bx
10289                                   ;cmp    si, -1          ; one apart? (consequtive?)
10290                                   ;                       ; (current = previous+1)
10291
10292                                   jnz     short getcl2   ; no, read [doscnt] sectors
10293
10294                                   ;add    [cs:doscnt], cx ; (cx = sectors per cluster)
10295                                   add     [doscnt], cx ; 12/12/2023 ; add to read count
10296                                   jmp     unpack
10297                           ; ---------------------------------------------------------------------------
10298
10299                           getcl2:
10300                                   push    si ; 20/12/2023
10301                                   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
0000291C 00<rep 4h>             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 $-$$
```

```
10550                                   ;--- End of DOSBIOS data segment ----------------------------------------
10551                                   ; ------------------------------------------------------------------------
10552                                             ;db 4 dup(0)
10553                                   ; 09/12/2022
10554                                   ;         times 4 db 0   ; 19/10/2022
10555                                   ; ------------------------------------------------------------------------
10556
10557                                   ;========================================================================
10558                                   ; DOS BIOS (IO.SYS) CODE SEGMENT
10559                                   ;========================================================================
10560                                   ; 09/12/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 IO.SYS)
10561                                   ; 30/12/2022 - Retro DOS v4.2 (Modified MSDOS 6.21 IO.SYS)
10562
10563                                   section .BIOSCODE vstart=0
10564
10565                                   ; 30/12/2022
10566                                   ; (BIOSCODE SEGMENT is 2CCh for MSDOS 6.21 IO.SYS)  -- ((25C0h+700h)>>4) --
10567
10568                                   BCode_start:        ; 09/12/2022
10569
10570                                   ; 02/10/2022
10571
10572                                   ;--- DOSBIOS code segment -----------------------------------------------
10573                                   ;------------------------------------------------------------------------
10574                                   ; MSBIO1.ASM (MSDOS 6.0, 1991)
10575                                   ;------------------------------------------------------------------------
10576
10577                                   DOSBIOSEG_2C7h:    ;db 30h dup(0)        ; SEGMENT 2C7h (2C70h-700h=2570h)
10578 00000000 00<rep 30h>                        times 48 db 0         ; 19/10/2022
10579 00000030 7000                    BiosDataWord:      dw 70h
10580
10581                                   ; 15/10/2022
10582                                   ;BIOSDATAWORD      equ BiosDataWord - DOSBIOSEG_2C7h
10583                                   ; 09/12/2022
10584                                   BIOSDATAWORD       equ BiosDataWord
10585
10586                                   ; ------------------------------------------------------------------------
10587
10588                                   ; 15/10/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 IO.SYS)
10589                                   ; 20/12/2023 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMBIO.COM)
10590
10591                                   ;************************************************************************
10592                                   ;*                                                                      *
10593                                   ;*  seg_reinit is called with ax = our new code segment value,          *
10594                                   ;*    trashes di, cx, es                                                *
10595                                   ;*                                                                      *
10596                                   ;*  cas -- should be made disposable!                                   *
10597                                   ;*                                                                      *
10598                                   ;************************************************************************
10599
10600                                       ; 20/09/2023
10601                                       ; 10/08/2023 - Retro DOS v4.2 IO.SYS (optimization)
10602                                       ; PCDOS 7.1 IBMBIO.COM - BIOSCODE:0032h
10603
10604                                   _seg_reinit:
10605 00000032 2E8E06[3000]                        mov     es, [cs:BIOSDATAWORD]
10606                                                                   ; at 2C7h:30h or 70h:25A0h
10607                                             ;mov    di, (offset cdev+2)
10608 00000037 BF[0406]                           mov     di, cdev+2     ; 19/10/2022
10609                                             ;mov    cx, 4          ; (end_BC_entries - cdev)/4
10610                                             ; 10/08/2023
10611 0000003A B90300                             mov     cx, 3 ; (PCDOS 7.1)
10612                                   _seg_reinit_1:
10613 0000003D AB                                 stosw                  ; modify Bios_Code entry points
10614 0000003E 47                                 inc     di
10615 0000003F 47                                 inc     di
10616 00000040 E2FB                               loop    _seg_reinit_1
10617                                             ; 10/08/2023 (PCDOS 7.1)
10618                                             ; (direct jump to i2f_handler from BIOSDATA:bios_i2f)
10619                                             ; (instead of 'bcode_i2f: dw i2f_handler, IOSYSCODESEG')
10620 00000042 26A3[0800]                         mov     [es:bios_i2f_seg], ax ; actual BIOSCODE segment
10621
10622 00000046 CB                                 retf
10623
10624                                   ; ------------------------------------------------------------------------
10625
10626                                   ; 15/10/2022
10627
10628                                   ;************************************************************************
10629                                   ;*                                                                      *
10630                                   ;*  chardev_entry - main device driver dispatch routine                 *
10631                                   ;*     called with a dummy parameter block on the stack                 *
10632                                   ;*     dw dispatch_table, dw prn/aux numbers (optional)                 *
10633                                   ;*                                                                      *
10634                                   ;*  will eventually take care of doing the transitions in               *
10635                                   ;*     out of Bios_Code                                                 *
10636                                   ;*                                                                      *
10637                                   ;************************************************************************
10638
10639                                             ; 20/09/2023
10640                                   chardev_entry:                         ; 0070h:25B3h =       02C7h:0043h
10641 00000047 56                                 push    si
10642 00000048 50                                 push    ax
10643 00000049 51                                 push    cx
10644 0000004A 52                                 push    dx
10645 0000004B 57                                 push    di
10646 0000004C 55                                 push    bp
10647 0000004D 1E                                 push    ds
10648 0000004E 06                                 push    es
10649 0000004F 53                                 push    bx
10650 00000050 89E5                               mov     bp, sp
10651 00000052 8B7612                             mov     si, [bp+18]    ; get return address (dispatch table)
10652                                             ;;mov   ds, word [cs:0030h]
10653                                             ;mov    ds, word ptr cs:BIOSDATAWORD ; [cs:0030h]
10654 00000055 2E8E1E[3000]                       mov     ds, [cs:BIOSDATAWORD] ; 17/10/2022
10655                                             ; 20/09/2023 (PCDOS 7.1 - IBMBIO.COM - BIOSCODE:005Ah)
10656 0000005A C434                               les     si, [si]
10657                                             ;mov    ax, [si+2]     ; get the device number       if present
10658 0000005C 8CC0                               mov     ax, es
10659 0000005E A2[2100]                           mov     [auxnum], al
10660 00000061 8826[8004]                         mov     [printdev], ah
10661                                             ;mov    si, [si]       ; point to the device dispatch table
10662 00000065 C41E[1200]                         les     bx, [ptrsav]   ; get pointer to i/o packet
10663 00000069 268A4701                           mov     al, [es:bx+1]  ; [es:bx+unit] ; al = unit code
10664 0000006D 268A670D                           mov     ah, [es:bx+13] ; [es:bx+media]       ; ah = media descrip
10665 00000071 268B4F12                           mov     cx, [es:bx+18] ; [es:bx+count]       ; cx = count
10666 00000075 268B5714                           mov     dx, [es:bx+20] ; [es:bx+start]       ; dx = start sector
10667                                             ; 17/10/2022
10668 00000079 81FE[6F05]                         cmp     si, DSKTBL
10669                                             ;;cmp   si, 579h       ; (PCDOS 7.1 IBMBIO.COM)
10670                                             ;cmp    si, 4A2h       ; dsktbl
10671                                                                    ; at 2C7h:4A2h = 70h:2A12h
10672 0000007D 7517                               jnz     short no_sector32_mapping
10673
```

```
                                ; Special case for 32-bit start sector number:
                                ;   if (si==dsktbl) /* if this is a disk device call */
                                ;      set high 16 bits of secnum to 0
                                ;      if (secnum == 0xffff) fetch 32 bit sector number
                                ;
                                ; pass high word of sector number in start_sec_h, low word in dx
                                ;
                                ; note: start_l and start_h are the offsets within the io_request packet
                                ;     which contain the low and hi words of the 32 bit start sector if
                                ;     it has been used.
                                ;
                                ; note: remember not to destroy the registers which have been set up before

                                        ; 20/09/2023
                                        ;mov   ds:start_sec_h,       0 ; initialize to 0
0000007F C706[9C04]0000                 mov     word [start_sec_h], 0
00000085 83FAFF                         cmp     dx, 0FFFFh
00000088 750C                           jnz     short no_sector32_mapping
0000008A 268B571C                       mov     dx, [es:bx+28] ; [es:bx+start_h]
                                                                ; 32 bits dsk req
                                        ;mov   ds:start_sec_h,       dx ; start_sec_h = packet.start_h
0000008E 8916[9C04]                     mov     [start_sec_h], dx
00000092 268B571A                       mov     dx, [es:bx+26] ; [es:bx+start_l]
                                                                ; dx = packet.start_l
                                no_sector32_mapping:
00000096 97                             xchg    ax, di
00000097 268A4702                       mov     al, [es:bx+2]  ; [es:bx+cmd]
0000009B 2E3A04                         cmp     al, [cs:si]
0000009E 732B                           jnb     short command_error
000000A0 98                             cbw                     ; note that al <= 15 means ok
000000A1 D1E0                           shl     ax, 1
000000A3 01C6                           add     si, ax
000000A5 97                             xchg    ax, di
000000A6 26C47F0E                       les     di, [es:bx+14] ; [es:bx+trans]
000000AA FC                             cld
                                        ; 17/10/2022
000000AB 2EFF5401                       call    near [cs:si+1]
                                        ;call  word ptr cs:si+1
000000AF 7202                           jb      short already_got_ah_status
000000B1 B401                           mov     ah, 1
                                already_got_ah_status:
                                        ;;mov  ds, [cs:0030h] ; 15/10/2022
                                        ;mov   ds, word ptr cs:BIOSDATAWORD ; [cs:0030h]
                                                                ; cas note: shouldn't be needed!
000000B3 2E8E1E[3000]                   mov     ds, [cs:BIOSDATAWORD] ; 17/10/2022
                                        ;lds   bx, ds:ptrsav
000000B8 C51E[1200]                     lds     bx, [ptrsav]
000000BC 894703                         mov     [bx+3], ax      ; [bx+status]
                                                                ; mark operation complete
000000BF 5B                             pop     bx
000000C0 07                             pop     es
000000C1 1F                             pop     ds
000000C2 5D                             pop     bp
000000C3 5F                             pop     di
000000C4 5A                             pop     dx
000000C5 59                             pop     cx
000000C6 58                             pop     ax
000000C7 5E                             pop     si
                                        ;add   sp, 2           ; get rid of fake return address
                                        ; 20/09/2023 (PCDOS 7.1       - IBMBIO.COM - BIOSCODE:00C8h)
000000C8 44                             inc     sp
000000C9 44                             inc     sp

                                        ; fall through into bc_retf
                                ; ---------------------------------------------------------------------------
                                bc_retf:
000000CA CB                             retf
                                ; ---------------------------------------------------------------------------

                                command_error:
000000CB E80700                         call    bc_cmderr
000000CE EBE3                           jmp     short already_got_ah_status
                                ; 15/10/2022
                                ; 01/05/2019

                                ;---------------------------------------------------------------------------
                                ; The following piece of hack is for supporting CP/M compatibility
                                ; Basically at offset 5 we have a far call into 0:c0. But this does not call
                                ; 0:c0 directly instead it call f01d:fef0, because it needs to support 'lhld 6'
                                ; The following hack has to reside at ffff:d0 (= f01d:fef0) if BIOS is loaded
                                ; high.
                                ;---------------------------------------------------------------------------

                                        ;db 7 dup(0)

                                        ; 20/09/2023 - Retro DOS v4.2 IO.SYS (Optimization)
                                        ; (PCDOS 7.1 - IBMBIO.COM - BIOSCODE:0D0h)
                                        ; 15/10/2022
                                        ;dw 0           ; pad to bring offset to 0D0h

000000D0 00<rep 5h>             off_d0:     times 5 db 0   ; 5 bytes from 0:c0 will be copied onto here
                                            ;  which is the CP/M call 5 entry point


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

                                ;   exit - all routines return through this path

                                bc_cmderr:
000000D5 B003                           mov     al, 3           ; 2C7h:D5h = 70h:2645h
                                                                ; unknown command error

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

                                ;   now zero the count field by subtracting its current value,
                                ;     which is still in cx, from itself.

                                ;   subtract the number of i/o's NOT YET COMPLETED from total
                                ;     in order to return the number actually complete

                                bc_err_cnt:
                                        ;les    bx, ds:ptrsav
                                        ; 19/10/2022
000000D7 C41E[1200]                     les     bx, [ptrsav]
000000DB 26294F12                       sub     [es:bx+18], cx ; [es:bx+count]
                                                                ; # of successful i/o's
000000DF B481                           mov     ah, 81h         ; mark error return
000000E1 F9                             stc                     ; indicate abnormal end
000000E2 C3                             retn

                                ; 15/10/2022

                                ;Bios_Code ends
```

```
10798                                   ;--------------------------------------------------------------------------
10799                                   ; MSCHAR.ASM - MSDOS 6.0 - 1991
10800                                   ;--------------------------------------------------------------------------
10801                                   ; 15/10/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 IO.SYS)
10802                                   ; 10/01/2019 - Retro DOS v4.0
10803
10804                                   ; 30/04/2019
10805
10806                                   ;title     mschar - character and clock devices
10807
10808                                   ;MODE_CTRLBRK      equ     0FFh
10809
10810                                   ; BIOSCODE:00E4h (MSDOS 6.21, IO.SYS)
10811
10812                                   ;**************************************************************************
10813                                   ;*                                                                        *
10814                                   ;*  device driver dispatch tables                                         *
10815                                   ;*                                                                        *
10816                                   ;*  each table starts with a byte which lists the number of           *
10817                                   ;*  legal functions, followed by that number of words. Each           *
10818                                   ;*  word represents an offset of a routine in Bios_Code which  *
10819                                   ;*  handles the function. The functions are terminated with           *
10820                                   ;*  a near return. If carry is reset, a 'done' code is returned       *
10821                                   ;*  to the caller. If carry is set, the ah/al registers are           *
10822                                   ;*  returned as abnormal completion status. Notice that ds            *
10823                                   ;*  is assumed to point to the Bios_Data segment throughout.   *
10824                                   ;*                                                                        *
10825                                   ;**************************************************************************
10826
10827                                               ; 20/09/2023
10828                                               ; (PCDOS 7.1 - IBMBIO.COM - BIOSCODE:00E3h)
10829                                               ; 13/12/2022
10830 000000E3 00                                   db 0
10831
10832                                               ; 13/12/2022
10833 000000E4 0B                       con_table:  db ((con_table_end - con_table)-1)/2 ; 11
10834                                                                   ; 2C7h:0E4h = 70h:2654h
10835 000000E5 [FA01]                               dw bc_exvec  ; 1FBh   ; bc_exvec at 2C7h:1FBh      = 70h:276Bh
10836                                                                   ; 00 init
10837 000000E7 [FA01]                               dw bc_exvec  ; 1FBh   ; 01
10838 000000E9 [FA01]                               dw bc_exvec  ; 1FBh   ; 02
10839 000000EB [D500]                               dw bc_cmderr ; 0D5h   ; bc_exvec at 2C7h:D5h = 70h:2645h
10840                                                                   ; 03
10841 000000ED [5C01]                               dw con_read  ; 15Ch   ; con_read at 2C7h:15Ch      = 70h:26CCh
10842                                                                   ; 04
10843 000000EF [9F01]                               dw con_rdnd  ; 19Fh   ; con_rdnd at 2C7h:19Fh      = 70h:270Fh
10844                                                                   ; 05
10845 000000F1 [FA01]                               dw bc_exvec  ; 1FBh   ; 06
10846 000000F3 [0802]                               dw con_flush ; 209h   ; con_flush at 2C7h:209h = 70h:2779h
10847                                                                   ; 07
10848 000000F5 [FC01]                               dw con_writ  ; 1FDh   ; con_writ at 2C7h:1FDh      = 70h:276Dh
10849                                                                   ; 08
10850 000000F7 [FC01]                               dw con_writ  ; 1FDh   ; 09
10851 000000F9 [FA01]                               dw bc_exvec  ; 1FBh   ; 0A
10852                                   con_table_end:
10853 000000FB 1A                       prn_table:  db ((prn_table_end - prn_table)-1)/2 ; 26
10854                                                                   ; 2C7h:0FBh = 70h:266Bh
10855 000000FC [FA01]                               dw bc_exvec   ; 1FBh  ; bc_exvec
10856 000000FE [FA01]                               dw bc_exvec   ; 1FBh  ; 01
10857 00000100 [FA01]                               dw bc_exvec   ; 1FBh  ; 02
10858 00000102 [D500]                               dw bc_cmderr  ;       0D5h    ; bc_cmderr
10859 00000104 [1902]                               dw prn_input  ;       21Ah    ; prn_input
10860                                                                   ; 04 indicate zero chars read
10861 00000106 [C701]                               dw z_bus_exit ; 1C8h  ; z_bus_exit
10862                                                                   ; 05 read non-destructive
10863 00000108 [FA01]                               dw bc_exvec   ; 1FBh  ; 06
10864 0000010A [FA01]                               dw bc_exvec   ; 1FBh  ; 07
10865 0000010C [1E02]                               dw prn_writ   ;       21Fh    ; prn_writ
10866 0000010E [1E02]                               dw prn_writ   ; 21Fh  ; 09
10867 00000110 [4F02]                               dw prn_stat   ; 251h  ; prn_stat
10868 00000112 [FA01]                               dw bc_exvec   ; 1FBh  ; 0B
10869 00000114 [FA01]                               dw bc_exvec   ; 1FBh  ; 0C
10870 00000116 [FA01]                               dw bc_exvec   ; 1FBh  ; 0D
10871 00000118 [FA01]                               dw bc_exvec   ; 1FBh  ; 0E
10872 0000011A [FA01]                               dw bc_exvec   ; 1FBh  ; 0F
10873 0000011C [9402]                               dw prn_tilbusy ; 28Bh ; prn_tilbusy
10874 0000011E [FA01]                               dw bc_exvec   ; 1FBh  ; 11
10875 00000120 [FA01]                               dw bc_exvec   ; 1FBh  ; 12
10876 00000122 [C202]                               dw prn_genioctl ; 2BAh; prn_genioctl
10877 00000124 [FA01]                               dw bc_exvec   ; 1FBh  ; 14
10878 00000126 [FA01]                               dw bc_exvec   ; 1FBh  ; 15
10879 00000128 [FA01]                               dw bc_exvec   ; 1FBh  ; 16
10880 0000012A [FA01]                               dw bc_exvec   ; 1FBh  ; 17
10881 0000012C [FA01]                               dw bc_exvec   ; 1FBh  ; 18
10882 0000012E [F702]                               dw prn_ioctl_query ; 2F0h ; prn_ioctl_query
10883                                   prn_table_end:
10884 00000130 0B                       aux_table:  db ((aux_table_end - aux_table)-1)/2 ; 11
10885                                                                   ; 2C7h:130h = 70h:26A0h
10886 00000131 [FA01]                               dw bc_exvec   ; 1FBh  ; 00 - init
10887 00000133 [FA01]                               dw bc_exvec   ; 1FBh  ; 01
10888 00000135 [FA01]                               dw bc_exvec   ; 1FBh  ; 02
10889 00000137 [D500]                               dw bc_cmderr  ;       0D5h    ; 03
10890 00000139 [1203]                               dw aux_read   ; 30Dh  ; aux_read ; 04       - read
10891 0000013B [3703]                               dw aux_rdnd   ; 335h  ; aux_rdnd - 05       - read non-destructive
10892 0000013D [FA01]                               dw bc_exvec   ; 1FBh  ; 06
10893 0000013F [7803]                               dw aux_flsh   ;       36Ch    ; aux_flsh
10894 00000141 [7F03]                               dw aux_writ   ;       374h    ; aux_writ
10895 00000143 [7F03]                               dw aux_writ   ;       374h    ; 09
10896 00000145 [5703]                               dw aux_wrst   ;       355h    ; aux_wrst
10897                                   aux_table_end:
10898 00000147 0A                       tim_table   db ((tim_table_end - tim_table)-1)/2 ; 10
10899                                                                   ; 2C7h:147h = 70h:26B7h
10900 00000148 [FA01]                               dw bc_exvec   ; 1FBh  ; 00
10901 0000014A [FA01]                               dw bc_exvec   ; 1FBh  ; 01
10902 0000014C [FA01]                               dw bc_exvec   ; 1FBh  ; 02
10903 0000014E [D500]                               dw bc_cmderr  ;       0D5h    ; 03
10904 00000150 [E404]                               dw tim_read   ;       435h    ; tim_read
10905 00000152 [C701]                               dw z_bus_exit ; 1C8h  ; z_bus_exit
10906 00000154 [FA01]                               dw bc_exvec   ; 1FBh  ; 06
10907 00000156 [FA01]                               dw bc_exvec   ; 1FBh  ; 07
10908 00000158 [E503]                               dw tim_writ   ; 3DBh  ; tim_writ
10909 0000015A [E503]                               dw tim_writ   ; 3DBh  ; 09
10910                                   tim_table_end:
10911
10912                                   ; --------------------------------------------------------------------------
10913
10914                                   ;**************************************************************************
10915                                   ;*                                                                        *
10916                                   ;*  con_read - read cx bytes from keyboard into buffer at es:di       *
10917                                   ;*                                                                        *
10918                                   ;**************************************************************************
10919
10920                                   con_read:                       ; 2C7h:15Ch = 70h:26CCh
10921                                               ;jcxz   short con_exit ; read cx bytes       from keyboard into buffer
```

```
10922 0000015C E306                             jcxz    con_exit        ; 19/10/2022
10923                                   con_loop:
10924 0000015E E80500                           call    chrin           ; get char in al
10925 00000161 AA                               stosb                   ; store char at es:di
10926 00000162 E2FA                             loop    con_loop
10927                                   con_exit:
10928 00000164 F8                               clc
10929 00000165 C3                               retn
10930
10931                                   ; =============== S U B   R O U T I N E =======================================
10932
10933                                   ;****************************************************************************
10934                                   ;*                                                                          *
10935                                   ;*  chrin - input single char from keyboard into al                         *
10936                                   ;*                                                                          *
10937                                   ;*    we are going to issue extended keyboard function, if                  *
10938                                   ;*    supported. the returning value of the extended keystroke              *
10939                                   ;*    of the extended keyboard function uses 0E0h in al                     *
10940                                   ;*    instead of 00h as in the conventional keyboard function.              *
10941                                   ;*    this creates a conflict when the user entered real                    *
10942                                   ;*    greek alpha charater (= 0E0h) to  distinguish the extended            *
10943                                   ;*    keystroke and the greek alpha. this case will be handled              *
10944                                   ;*    in the following manner:                                              *
10945                                   ;*                                                                          *
10946                                   ;*        ah = 16h                                                          *
10947                                   ;*        int 16h                                                           *
10948                                   ;*        if al == 0, then extended code (in ah)                            *
10949                                   ;*        else if al == 0E0h, then                                          *
10950                                   ;*        if ah <> 0, then extended code (in ah)                            *
10951                                   ;*          else greek_alpha character.                                     *
10952                                   ;*                                                                          *
10953                                   ;*  also, for compatibility reason, if an extended code is                  *
10954                                   ;*    detected, then we are going to change the value in al                 *
10955                                   ;*    from 0E0h to 00h.                                                     *
10956                                   ;*                                                                          *
10957                                   ;****************************************************************************
10958                                   ;
10959                                           ; 19/10/2022
10960                                   chrin:
10961 00000166 8A26[7E04]                       mov     ah, [keyrd_func] ; set by msinit. 0 or 10h
10962 0000016A 30C0                             xor     al, al
10963 0000016C 8606[0C00]                       xchg    al, [altah]     ; get character       & zero altah
10964 00000170 08C0                             or      al, al
10965 00000172 752A                             jnz     short keyret
10966 00000174 CD16                             int     16h             ; KEYBOARD -
10967 00000176 09C0                             or      ax, ax
10968 00000178 74EC                             jz      short chrin
10969 0000017A 3D0072                           cmp     ax, 7200h       ; check for ctrl-prtsc
10970 0000017D 7504                             jnz     short alt_ext_chk
10971 0000017F B010                             mov     al, 10h
10972 00000181 EB1B                             jmp     short keyret
10973                                   ; ---------------------------------------------------------------------------
10974
10975                                   ;  if operation was extended function (i.e. keyrd_func != 0) then
10976                                   ;    if character read was 0E0h then
10977                                   ;      if extended byte was zero (i.e. ah == 0) then
10978                                   ;    goto keyret
10979                                   ;      else
10980                                   ;    set al to zero
10981                                   ;    goto alt_save
10982                                   ;      endif
10983                                   ;    endif
10984                                   ;  endif
10985
10986                                   alt_ext_chk:
10987 00000183 803E[7E04]00                     cmp     byte [keyrd_func], 0
10988 00000188 740C                             jz      short not_ext
10989 0000018A 3CE0                             cmp     al, 0E0h
10990 0000018C 7508                             jnz     short not_ext
10991 0000018E 08E4                             or      ah, ah
10992 00000190 740C                             jz      short keyret
10993 00000192 30C0                             xor     al, al
10994 00000194 EB04                             jmp     short alt_save
10995                                   ; ---------------------------------------------------------------------------
10996
10997                                   not_ext:
10998 00000196 08C0                             or      al, al          ; special case?
10999 00000198 7504                             jnz     short keyret
11000                                   alt_save:
11001 0000019A 8826[0C00]                       mov     [altah], ah     ; store special key
11002                                   keyret:
11003 0000019E C3                               retn
11004
11005                                   ; ---------------------------------------------------------------------------
11006
11007                                   ;****************************************************************************
11008                                   ;*                                                                          *
11009                                   ;*  con_rdnd - keyboard non destructive read, no wait                       *
11010                                   ;*                                                                          *
11011                                   ;*  pc-convertible-type machine: if bit 10 is set by the dos                *
11012                                   ;*  in the status word of the request packet, and there is no               *
11013                                   ;*  character in the input buffer, the driver issues a system               *
11014                                   ;*  wait request to the rom. on return from the rom, it returns             *
11015                                   ;*  a 'char-not-found' to the dos.                                          *
11016                                   ;*                                                                          *
11017                                   ;****************************************************************************
11018
11019                                           ; 19/10/2022
11020                                   con_rdnd:
11021 0000019F A0[0C00]                         mov     al, [altah]
11022 000001A2 08C0                             or      al, al
11023 000001A4 754C                             jnz     short rdexit
11024 000001A6 8A26[7F04]                       mov     ah, [keysts_func]
11025 000001AA CD16                             int     16h             ; KEYBOARD -
11026 000001AC 751D                             jnz     short gotchr
11027 000001AE 803E[7900]00                     cmp     byte [fhavek09], 0
11028 000001B3 7412                             jz      short z_bus_exit
11029 000001B5 C41E[1200]                       les     bx, [ptrsav]
11030                                           ; 12/12/2022
11031 000001B9 26F6470404                       test    byte [es:bx+4], 04h
11032                                           ;test   word [es:bx+3], 400h ; [es:bx+status]
11033 000001BE 7407                             jz      short z_bus_exit
11034 000001C0 B80041                           mov     ax, 4100h
11035 000001C3 30DB                             xor     bl, bl
11036 000001C5 CD15                             int     15h             ; SYSTEM - WAIT       ON EXTERNAL EVENT (CONVERTIBLE)
11037                                                                   ; AL = condition type, BH = condition compare or mask value
11038                                                                   ; BL = timeout value times 55 milliseconds, 00h     means no timeout
11039                                                                   ; DX = I/O port       address if AL bit 4 set
11040                                   z_bus_exit:
11041 000001C7 F9                               stc                     ; 2C7h:1C8h = 70h:2738h
11042 000001C8 B403                             mov     ah, 3           ; indicate busy       status
11043 000001CA C3                               retn
11044                                   ; ---------------------------------------------------------------------------
11045
```

```
11046                                   gotchr:
11047 000001CB 09C0                               or      ax, ax
11048 000001CD 7508                               jnz     short notbrk   ; check for null after break
11049 000001CF 8A26[7E04]                         mov     ah, [keyrd_func] ; issue keyboard read function
11050 000001D3 CD16                               int     16h            ; KEYBOARD -
11051 000001D5 EBC8                               jmp     short con_rdnd ; get a real status
11052                                   ; ---------------------------------------------------------------------------
11053
11054                                   notbrk:
11055 000001D7 3D0072                             cmp     ax, 7200h      ; check for ctrl-prtsc
11056 000001DA 7504                               jnz     short rd_ext_chk
11057 000001DC B010                               mov     al, 10h        ; ('P' & 1Fh) ; return control p
11058 000001DE EB12                               jmp     short rdexit
11059                                   ; ---------------------------------------------------------------------------
11060
11061                                   rd_ext_chk:
11062 000001E0 803E[7E04]00                       cmp     byte [keyrd_func], 0 ; extended keyboard function?
11063 000001E5 740B                               jz      short rdexit
11064 000001E7 3CE0                               cmp     al, 0E0h       ; extended key value or      greek alpha?
11065 000001E9 7507                               jnz     short rdexit
11066 000001EB 80FC00                             cmp     ah, 0          ; scan code exist?
11067 000001EE 7402                               jz      short rdexit   ; yes. greek alpha char.
11068 000001F0 B000                               mov     al, 0          ; no. extended key stroke.
11069                                                                    ; change it for       compatibility
11070                                   rdexit:
11071 000001F2 C41E[1200]                         les     bx, [ptrsav]
11072 000001F6 2688470D                           mov     [es:bx+13], al ; [es:bx+media]
11073                                                                    ; return keyboard character here
11074                                   bc_exvec:
11075 000001FA F8                                 clc                    ; bc_exvec at 2C7h:1FBh      = 70h:276Bh
11076                                                                    ; indicate normal termination
11077 000001FB C3                                 retn
11078                                   ; ---------------------------------------------------------------------------
11079
11080                                   ;****************************************************************************
11081                                   ;*                                                                          *
11082                                   ;*  con_write - console write routine                                       *
11083                                   ;*                                                                          *
11084                                   ;*  entry:  es:di -> buffer                                                 *
11085                                   ;*          cx    =  count                                                  *
11086                                   ;*                                                                          *
11087                                   ;****************************************************************************
11088
11089                                   con_writ:
11090                                             ;jcxz   short bc_exvec
11091 000001FC E3FC                               jcxz    bc_exvec       ; 19/10/2022
11092                                             ; 12/12/2022
11093                                             ;jcxz   cc_ret
11094                                   con_lp:
11095 000001FE 268A05                             mov     al, [es:di]
11096 00000201 47                                 inc     di
11097 00000202 CD29                               int     29h            ; DOS 2+ internal - FAST PUTCHAR
11098                                                                    ; AL = character to display
11099 00000204 E2F8                               loop    con_lp
11100                                   cc_ret:
11101 00000206 F8                                 clc
11102 00000207 C3                                 retn
11103
11104                                   ; =============== S U B   R O U T I N E =======================================
11105
11106                                   ;****************************************************************************
11107                                   ;*                                                                          *
11108                                   ;*  con_flush - flush out keyboard queue                                    *
11109                                   ;*                                                                          *
11110                                   ;****************************************************************************
11111
11112                                   con_flush:
11113 00000208 C606[0C00]00                       mov     byte [altah], 0        ; clear out holding buffer
11114                                   flloop:                                  ; while (charavail()) charread();
11115 0000020D B401                               mov     ah, 1
11116 0000020F CD16                               int     16h            ; KEYBOARD - CHECK BUFFER, DO NOT CLEAR
11117                                                                    ; Return: ZF clear if character      in buffer
11118                                                                    ; AH = scan code, AL = character
11119                                                                    ; ZF set if no character in buffer
11120 00000211 74F3                               jz      short cc_ret
11121 00000213 30E4                               xor     ah, ah
11122 00000215 CD16                               int     16h            ; KEYBOARD - READ CHAR FROM BUFFER, WAIT IF EMPTY
11123                                                                    ; Return: AH = scan code, AL = character
11124 00000217 EBF4                               jmp     short flloop
11125                                   ; ---------------------------------------------------------------------------
11126
11127                                   ; 15/10/2022
11128
11129                                   ;****************************************************************************
11130                                   ;*                                                                          *
11131                                   ;*  some equates for rom bios printer i/o                                   *
11132                                   ;*                                                                          *
11133                                   ;****************************************************************************
11134
11135                                   ; ibm rom status bits (i don't trust them, neither should you)
11136                                   ; warning!!! the ibm rom does not return just one bit. it returns a
11137                                   ; whole slew of bits, only one of which is correct.
11138
11139                                   ;notbusystatus     equ 10000000b         ; not busy
11140                                   ;nopaperstatus     equ 00100000b         ; no more paper
11141                                   ;prnselected       equ 00010000b         ; printer selected
11142                                   ;ioerrstatus       equ 00001000b         ; some kinda error
11143                                   ;timeoutstatus     equ 00000001b         ; time out.
11144                                   ;
11145                                   ;noprinter  equ 00110000b         ; no printer attached
11146
11147                                   ; 18/03/2019 - Retro DOS v4.0
11148                                   ;error_I24_out_of_paper   equ 9 ; MSDOS 6.0, ERR.INC, 1991
11149
11150                                   ; ---------------------------------------------------------------------------
11151
11152                                   ;****************************************************************************
11153                                   ;*                                                                          *
11154                                   ;*  prn_input - return with no error but zero chars read                    *
11155                                   ;*                                                                          *
11156                                   ;*  enter with cx = number of characters requested                          *
11157                                   ;*                                                                          *
11158                                   ;****************************************************************************
11159
11160                                   prn_input:                       ; 2C7h:21Ah = 70h:278Ah
11161 00000219 E8BBFE                             call    bc_err_cnt     ; reset count to zero
11162                                                                    ; (sub reqpkt.count,cx)
11163                                             ; 12/12/2022
11164                                   prn_done:
11165 0000021C F8                                 clc                    ; but return with carry      reset for no error
11166 0000021D C3                                 retn
11167                                   ; ---------------------------------------------------------------------------
11168
11169
```

```
11170                                   ;****************************************************************************
11171                                   ;*                                                                          *
11172                                   ;*  prn_writ - write cx bytes from es:di to printer device                  *
11173                                   ;*                                                                          *
11174                                   ;*  auxnum has printer number                                               *
11175                                   ;*                                                                          *
11176                                   ;****************************************************************************
11177
11178                                   prn_writ:                       ; 2C7h:21Fh = 70h:278Fh
11179                                           ;jcxz   short prn_done ; no chars to output
11180 0000021E E3FC                       jcxz    prn_done        ; 19/10/2022
11181                                   prn_loop:
11182 00000220 BB0200                   mov     bx, 2           ; retry count
11183                                   prn_out:
11184 00000223 E83600                   call    prnstat         ; get status
11185 00000226 751D                     jnz     short TestPrnError
11186 00000228 268A05                   mov     al, [es:di]     ; get character       to print
11187 0000022B 30E4                     xor     ah, ah
11188 0000022D E82E00                   call    prnop           ; print to printer
11189 00000230 7419                     jz      short prn_con   ; no error - continue
11190 00000232 80FCFF                   cmp     ah, 0FFh        ; MODE_CTRLBRK
11191 00000235 7509                     jnz     short _prnwf
11192 00000237 B00C                     mov     al, 0Ch         ; error_I24_gen_failure
11193 00000239 C606[0C00]00             mov     byte [altah], 0
11194 0000023E EB08                     jmp     short pmessg
11195                                   ; ---------------------------------------------------------------------------
11196
11197                                   _prnwf:
11198 00000240 F6C401                   test    ah, 1           ; timeoutstatus
11199 00000243 7406                     jz      short prn_con
11200                                   TestPrnError:
11201 00000245 4B                       dec     bx              ; retry until count is exhausted.
11202 00000246 75DB                     jnz     short prn_out
11203                                   pmessg:
11204 00000248 E98CFE                   jmp     bc_err_cnt
11205                                   ; ---------------------------------------------------------------------------
11206
11207                                   prn_con:
11208 0000024B 47                       inc     di              ; point to next char and continue
11209 0000024C E2D2                     loop    prn_loop
11210                                   ;prn_done:
11211                                           ; 12/12/2022
11212                                   prn_done2:
11213                                           ;clc
11214                                           ; cf=0
11215 0000024E C3                       retn
11216                                   ; ---------------------------------------------------------------------------
11217
11218                                   ;****************************************************************************
11219                                   ;*                                                                          *
11220                                   ;*  prn_stat - device driver entry to return printer status                 *
11221                                   ;*                                                                          *
11222                                   ;****************************************************************************
11223
11224                                   prn_stat:                       ; 2C7h:251h = 70h:27C1h
11225 0000024F E80A00                   call    prnstat         ; device in dx
11226 00000252 75F4                     jnz     short pmessg
11227 00000254 F6C480                   test    ah, 80h         ; notbusystatus
11228                                           ;jnz    short prn_done
11229                                           ; 12/12/2022
11230 00000257 75F5                     jnz     short prn_done2 ; cf=0
11231 00000259 E96BFF                   jmp     z_bus_exit
11232                                   ; ---------------------------------------------------------------------------
11233
11234                                   ;****************************************************************************
11235                                   ;*                                                                          *
11236                                   ;*  prnstat - utility function to call ROM BIOS to check                    *
11237                                   ;*           printer status. Return meaningful error code                   *
11238                                   ;*                                                                          *
11239                                   ;****************************************************************************
11240
11241                                   prnstat:
11242 0000025C B402                     mov     ah, 2           ; set command for get status
11243                                                           ; PRINTER - GET       STATUS
11244                                                           ; DX = printer port (0-3)
11245                                                           ; Return: AH = status
11246
11247                                   ; ============== S U B   R O U T I N E =======================================
11248
11249                                   ;****************************************************************************
11250                                   ;*                                                                          *
11251                                   ;*  prnop - call ROM BIOS printer function in ah                            *
11252                                   ;*          return zero true if no error                                    *
11253                                   ;*          return zero false if error, al = error code                     *
11254                                   ;*                                                                          *
11255                                   ;****************************************************************************
11256
11257                                   prnop:
11258                                           ; 20/12/2023 - Retro DOS v5.0
11259                                           ; PCDOS 7.1 IBMBIO.COM
11260
11261                                           ;mov    dx, [auxnum]    ; get printer number
11262                                           ;int    17h
11263
11264 0000025E 1E                               push    ds
11265 0000025F FF36[2100]                       push    word [auxnum]
11266 00000263 31D2                             xor     dx, dx ; 0
11267 00000265 8EDA                             mov     ds, dx
11268 00000267 5A                               pop     dx
11269 00000268 9C                               pushf                   ; simulate int 17h
11270 00000269 FA                               cli
11271                                           ;call   dword ptr ds:5ghghCh
11272 0000026A FF1E5C00                         call    far [17h*4]     ; 0:5Ch = INT 17h vector
11273 0000026E 1F                               pop     ds
11274
11275                                       ; This check was added to see if this is a case of no
11276                                       ; printer being installed. This tests checks to be sure
11277                                       ; the error is noprinter (30h)
11278
11279 0000026F 50                               push    ax
11280 00000270 80E430                           and     ah, 30h
11281 00000273 80FC30                           cmp     ah, 30h         ; noprinter
11282 00000276 58                               pop     ax
11283 00000277 7506                             jnz     short NextTest
11284 00000279 80E4DF                           and     ah, 0DFh        ; ~nopaperstatus
11285 0000027C 80CC08                           or      ah, 8           ; ioerrstatus
11286
11287                                   ; examine the status bits to see if an error occurred. unfortunately, several
11288                                   ; of the bits are set so we have to pick and choose. we must be extremely
11289                                   ; careful about breaking basic.
11290
11291                                   NextTest:
11292 0000027F F6C428                           test    ah, 28h         ; (ioerrstatus+nopaperstatus)
11293                                                                   ; i/o error?
```

```
11294 00000282 740A                             jz      short checknotready ; no, try not ready
11295
11296                                   ; at this point, we know we have an error. the converse is not true
11297
11298 00000284 B009                             mov     al, 9           ; error_I24_out_of_paper
11299                                                                   ; first, assume       out of paper
11300 00000286 F6C420                           test    ah, 20h         ; out of paper set?
11301 00000289 7502                             jnz     short ret1      ; yes, error is       set
11302 0000028B FEC0                             inc     al              ; return al=10 (i/o error)
11303                                   ret1:
11304 0000028D C3                               retn
11305                                   ; ---------------------------------------------------------------------------
11306
11307                                   checknotready:
11308 0000028E B002                             mov     al, 2           ; assume not-ready
11309 00000290 F6C401                           test    ah, 1
11310 00000293 C3                               retn
11311
11312                                   ; ---------------------------------------------------------------------------
11313
11314                                   ;****************************************************************************
11315                                   ;*                                                                          *
11316                                   ;*  prn_tilbusy - output until busy. Used by print spooler.                 *
11317                                   ;*                this entry point should never block waiting for       *
11318                                   ;*                device to come ready.                                 *
11319                                   ;*                                                                      *
11320                                   ;*  inputs: cx = count, es:di -> buffer                                   *
11321                                   ;*  outputs: set the number of bytes transferred in the                   *
11322                                   ;*           device driver request packet                                 *
11323                                   ;*                                                                      *
11324                                   ;****************************************************************************
11325
11326                                             ; 19/10/2022
11327                                   prn_tilbusy:                          ; 2C7h:28Bh = 70h:27FBh
11328 00000294 89FE                             mov     si, di          ; everything is       set for lodsb
11329                                   prn_tilbloop:
11330 00000296 51                               push    cx
11331 00000297 53                               push    bx
11332 00000298 30FF                             xor     bh, bh
11333 0000029A 8A1E[8004]                       mov     bl, [printdev]
11334 0000029E D1E3                             shl     bx, 1
11335                                           ;mov    cx, ds:wait_count[bx] ;       wait count times to come ready
11336 000002A0 8B8F[8104]                       mov     cx, [wait_count+bx]
11337 000002A4 5B                               pop     bx
11338                                   prn_getstat:
11339 000002A5 E8B4FF                           call    prnstat         ; get status
11340 000002A8 7514                             jnz     short prn_bperr       ; error
11341 000002AA F6C480                           test    ah, 80h         ; ready yet?
11342 000002AD E1F6                             loope   prn_getstat     ; no, go for more
11343 000002AF 59                               pop     cx              ; get original count
11344 000002B0 740D                             jz      short prn_berr ; still not ready => done
11345 000002B2 26                               es
11346 000002B3 AC                               lodsb
11347                                           ;lods   byte ptr es:[si] ; es
11348                                                                   ; lodsb
11349 000002B4 30E4                             xor     ah, ah
11350 000002B6 E8A5FF                           call    prnop
11351 000002B9 7504                             jnz     short prn_berr ; error
11352 000002BB E2D9                             loop    prn_tilbloop
11353                                           ; 12/12/2022
11354                                           ; cf=0 (prnop)
11355                                           ;clc                    ; normal no-error return
11356 000002BD C3                               retn                    ;   from device driver
11357
11358                                   ; ---------------------------------------------------------------------------
11359
11360                                   prn_bperr:
11361 000002BE 59                               pop     cx              ; restore transfer count from stack
11362                                   prn_berr:
11363 000002BF E915FE                           jmp     bc_err_cnt
11364                                   ; ---------------------------------------------------------------------------
11365
11366                                   ; 15/10/2022
11367
11368                                   ;****************************************************************************
11369                                   ;*                                                                          *
11370                                   ;*  prn_genioctl - get/set printer retry count                              *
11371                                   ;*                                                                          *
11372                                   ;****************************************************************************
11373
11374                                   ; IOCTL.INC (MSDOS 6.0, 1991)
11375                                   ; 11/01/2019
11376
11377                                   ;*********************************;*
11378                                   ; CHARACTER DEVICES (PRINTERS)    ;*
11379                                   ;*********************************;*
11380
11381                                   ;;RAWIO SUB-FUNCTIONS
11382                                   ;;get_retry_count equ 65h
11383                                   ;;set_retry_count equ 45h
11384
11385                                   ;;struc A_RETRYCOUNT
11386                                   ;;.rc_count: resw 1
11387                                   ;;endstruc
11388
11389                                   ;ioc_pc equ 5
11390
11391                                   ; ---------------------------------------------------------------------------
11392
11393                                             ; 19/10/2022
11394                                   prn_genioctl:                         ; 2C7h:2BAh = 70h:282Ah
11395 000002C2 C43E[1200]                       les     di, [ptrsav]
11396 000002C6 26807D0D05                       cmp     byte [es:di+13], 5 ; [es:di+IOCTL_REQ.MAJORFUNCTION]
11397                                                                   ; ioc_pc
11398 000002CB 7403                             jz      short prnfunc_ok
11399
11400                                   prnfuncerr:
11401 000002CD E905FE                           jmp     bc_cmderr
11402                                   ; ---------------------------------------------------------------------------
11403
11404                                   prnfunc_ok:
11405 000002D0 268A450E                         mov     al, [es:di+14] ; [es:di+IOCTL_REQ.MINORFUNCTION]
11406 000002D4 26C47D13                         les     di, [es:di+19] ; [es:di+IOCTL_REQ.GENERICIOCTL_PACKET]
11407 000002D8 30FF                             xor     bh, bh
11408                                           ;mov    bl, ds:printdev       ; get index into retry counts
11409 000002DA 8A1E[8004]                       mov     bl, [printdev]
11410 000002DE D1E3                             shl     bx, 1
11411                                           ;mov    cx, ds:wait_count[bx] ;       pull out retry count for device
11412 000002E0 8B8F[8104]                       mov     cx, [wait_count+bx]
11413 000002E4 3C65                             cmp     al, 65h         ; get_retry_count
11414 000002E6 7407                             jz      short prngetcount
11415 000002E8 3C45                             cmp     al, 45h         ; set_retry_count
11416 000002EA 75E1                             jnz     short prnfuncerr
11417 000002EC 268B0D                           mov     cx, [es:di]
```

```
                                prngetcount:
                                        ;mov   ds:wait_count[bx], cx
000002EF 898F[8104]                     mov    [wait_count+bx], cx
000002F3 26890D                         mov    [es:di], cx    ; [es:di+A_RETRYCOUNT.RC_COUNT]
                                                               ; return current retry count
                                        ; 12/12/2022
                                        ; cf=0
                                        ;clc
000002F6 C3                             retn
                                ; ---------------------------------------------------------------------------

                                ;****************************************************************************
                                ;*                                                                          *
                                ;*  prn_ioctl_query                                                         *
                                ;*                                                                          *
                                ;*  Added for 5.00                                                          *
                                ;****************************************************************************
                                ;
                                prn_ioctl_query:                ; 2C7h:2F0h = 70h:2860h
000002F7 C43E[1200]                     les    di, [ptrsav]
000002FB 26807D0D05                     cmp    byte [es:di+13], 5 ; [es:di+IOCTL_REQ.MAJORFUNCTION]
                                                               ; ioc_pc
00000300 750D                           jnz    short prn_query_err
00000302 268A450E                       mov    al, [es:di+14] ; [es:di+IOCTL_REQ.MINORFUNCTION]
00000306 3C65                           cmp    al, 65h        ; GET_RETRY_COUNT
00000308 7404                           jz     short IOCtlSupported
0000030A 3C45                           cmp    al, 45h        ; SET_RETRY_COUNT
0000030C 7501                           jnz    short prn_query_err
                                IOCtlSupported:
                                        ; 12/12/2022
                                        ; cf=0
                                        ;clc
0000030E C3                             retn
                                ; ---------------------------------------------------------------------------

                                prn_query_err:
                                        ; 12/12/2022
                                        ;stc
0000030F E9C3FD                         jmp    bc_cmderr ; (bc_cmderr sets cf to 1)
                                ; ---------------------------------------------------------------------------

                                ;****************************************************************************
                                ;*                                                                          *
                                ;*  aux port driver code -- "aux" == "com1"                                 *
                                ;*                                                                          *
                                ;*  the device driver entry/dispatch code sets up auxnum to                 *
                                ;*  give the com port number to use (0=com1, 1=com2, 2=com3...)             *
                                ;*                                                                          *
                                ;****************************************************************************

                                ;   values in ah, requesting function of int 14h in rom bios

                                ;auxfunc_send       equ   1       ;transmit
                                ;auxfunc_receive equ      2       ;read
                                ;auxfunc_status     equ   3       ;request status

                                ;   error flags, reported by int 14h, reported in ah:

                                ;flag_data_ready equ      01h     ;data ready
                                ;flag_overrun       equ   02h     ;overrun error
                                ;flag_parity        equ   04h     ;parity error
                                ;flag_frame  equ   08h     ;framing error
                                ;flag_break  equ   10h     ;break detect
                                ;flag_tranhol_emp equ     20h     ;transmit holding register empty
                                ;flag_timeout       equ   80h     ;timeout

                                ;   these flags reported in al:

                                ;flag_cts    equ   10h     ;clear to send
                                ;flag_dsr    equ   20h     ;data set ready
                                ;flag_rec_sig       equ   80h     ;receive line signal detect

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

                                ;****************************************************************************
                                ;*                                                                          *
                                ;*  aux_read - read cx bytes from [auxnum] aux port to buffer               *
                                ;*             at es:di                                                     *
                                ;*                                                                          *
                                ;****************************************************************************
                                ;
                                aux_read:                       ; 2C7h:30Dh = 70h:287Dh
                                        ;jcxz  short exvec2
00000312 E311                           jcxz   exvec2          ; 19/10/2022
00000314 E88000                         call   getbx           ; put address of auxbuf      in bx
00000317 30C0                           xor    al, al
00000319 8607                           xchg   al, [bx]
0000031B 08C0                           or     al, al
0000031D 7503                           jnz    short aux2
                                aux1:
0000031F E80500                         call   auxin           ; get character       from port
                                                               ; won't return if error
                                aux2:
00000322 AA                             stosb
00000323 E2FA                           loop   aux1            ; if more characters, go around      again
                                exvec2:
00000325 F8                             clc                    ; all done, successful exit
                                auxin_retn: ; 18/12/2022
00000326 C3                             retn
                                ; ---------------------------------------------------------------------------

                                ;****************************************************************************
                                ;*                                                                          *
                                ;*  auxin - call rom bios to read character from aux port                   *
                                ;*          if error occurs, map the error and return one                   *
                                ;*          level up to device driver exit code, setting                    *
                                ;*          the number of bytes transferred appropriately                   *
                                ;*                                                                          *
                                ;****************************************************************************
                                ;
                                auxin:
00000327 B402                           mov    ah, 2           ; auxfunc_receive
00000329 E83A00                         call   auxop
0000032C F6C40E                         test   ah, 0Eh         ; flag_frame|flag_parity|flag_overrun
                                        ;jnz   short arbad     ; skip if any error bits set
                                        ;retn
                                        ; 25/06/2023 (BugFix)
0000032F 74F5                           jz     short auxin_retn
                                ; ---------------------------------------------------------------------------

                                arbad:
00000331 58                             pop    ax             ; remove return       address (near call)
                                        ;xor   al, al
                                        ;or    al, 0B0h       ; flag_rec_sig|       flag_dsr|flag_cts
```

```
                                        ; 11/08/2023
00000332 B0B0                           mov     al, 0B0h        ; PCDOS 7.1 IBMBIO.COM - BIOSCODE:0334h
00000334 E9A0FD                         jmp     bc_err_cnt

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

                                ;*****************************************************************************
                                ;*                                                                           *
                                ;*  aux_rdnd - non-destructive aux port read                                 *
                                ;*                                                                           *
                                ;*****************************************************************************

                                aux_rdnd:                       ; 2C7h:335h = 70h:28A5h
00000337 E85D00                         call    getbx
0000033A 8A07                           mov     al, [bx]        ; have bx point       to auxbuf
0000033C 08C0                           or      al, al          ; if al is non-zero (char in buffer)
0000033E 7511                           jnz     short auxdrx    ; then return character
00000340 E82100                         call    auxstat         ; if not, get status of      aux device
00000343 F6C401                         test    ah, 1           ; flag_data_ready - test data ready
00000346 740C                           jz      short auxbus    ; then device is busy (not ready)
00000348 A820                           test    al, 20h         ; flag_dsr - test data set ready
0000034A 7408                           jz      short auxbus    ; then device is busy (not ready)
0000034C E8D8FF                         call    auxin           ; else aux is ready, get character
0000034F 8807                           mov     [bx], al
                                auxdrx:
00000351 E99EFE                         jmp     rdexit          ; return busy status
                                ; ---------------------------------------------------------------------------

                                auxbus:
00000354 E970FE                         jmp     z_bus_exit
                                ; ---------------------------------------------------------------------------

                                ;*****************************************************************************
                                ;*                                                                           *
                                ;*  aux_wrst - return aux port write status                                      *
                                ;*                                                                           *
                                ;*****************************************************************************

                                aux_wrst:                       ; 2C7h:355h = 70h:28C5h
00000357 E80A00                         call    auxstat         ; get status of       aux in ax
0000035A A820                           test    al, 20h         ; test data set       ready
0000035C 74F6                           jz      short auxbus    ; then device is busy (not ready)
0000035E F6C420                         test    ah, 20h         ; flag_tranhol_emp - test transmit hold     reg empty
00000361 74F1                           jz      short auxbus    ; then device is busy (not ready)
                                        ; 12/12/2022
                                        ; cf=0  ; (test instruction resets cf)
                                        ;clc
00000363 C3                             retn
                                ; ---------------------------------------------------------------------------

                                ;*****************************************************************************
                                ;*                                                                           *
                                ;*  auxstat - call rom bios to determine aux port status                         *
                                ;*                                                                           *
                                ;*  exit:   ax = status                                                      *
                                ;*          dx = [auxnum]                                                    *
                                ;*                                                                           *
                                ;*****************************************************************************

                                auxstat:
00000364 B403                           mov     ah, 3           ; auxfunc_status

                                        ; fall into auxop

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

                                ;*****************************************************************************
                                ;*                                                                           *
                                ;*  auxop - perform rom-biox aux port interrupt                              *
                                ;*                                                                           *
                                ;*  entry:  ah = int 14h function number                                     *
                                ;*  exit:   ax = results                                                     *
                                ;*          dx = [auxnum]                                                    *
                                ;*                                                                           *
                                ;*****************************************************************************

                                auxop:                  ; proc near
                                        ; 20/12/2023 - Retro DOS v5.0
                                        ;mov    dx, [auxnum]    ; ah=function code
                                        ;                       ; 0=init, 1=send, 2=receive, 3=status
                                        ;                       ; get port number
                                        ;
                                        ;int    14h             ; SERIAL I/O - GET USART STATUS
                                        ;                       ; DX = port number (0-3)
                                        ;                       ; Return: AX = port status code
                                        ; (PCDOS 7.1 IBMBIO.COM)
00000366 1E                             push    ds
00000367 FF36[2100]                     push    word [auxnum]
0000036B 31D2                           xor     dx, dx ; 0
0000036D 8EDA                           mov     ds, dx
0000036F 5A                             pop     dx
00000370 9C                             pushf                   ; simulate INT 14h
00000371 FA                             cli
                                        ;call   dword ptr ds:50h
00000372 FF1E5000                       call    far [14h*4]     ; INT 14h vector (14h*4 = 50h)
00000376 1F                             pop     ds
00000377 C3                             retn

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

                                ;*****************************************************************************
                                ;*                                                                           *
                                ;*  aux_flsh - flush aux input buffer - set contents of                      *
                                ;*             auxbuf [auxnum] to zero                                       *
                                ;*                                                                           *
                                ;*  cas - shouldn't this code call the rom bios input function               *
                                ;*        repeatedly until it isn't ready?  to flush out any                 *
                                ;*        pending serial input queue if there's a tsr like MODE              *
                                ;*        which is providing interrupt-buffering of aux port?                *
                                ;*                                                                           *
                                ;*****************************************************************************

                                aux_flsh:                       ; 2C7h:36Ch = 70h:28DCh
00000378 E81C00                         call    getbx           ; flush aux input buffer
0000037B C60700                         mov     byte [bx], 0    ; get bx to point to auxbuf
                                                                ; zero out buffer
                                        ;clc                    ; all done, successful return
                                        ; 12/12/2022
                                        ; cf=0 ('add' instruction in 'getbx')
0000037E C3                             retn
                                ; ---------------------------------------------------------------------------

                                ;*****************************************************************************
                                ;*                                                                           *
```

```
                        ;*  aux_writ - write to aux device                          *
                        ;*                                                          *
                        ;**************************************************************************

                        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 T I 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 87CA                   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

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

```
11790
11791                                   ; 17/10/2022
11792                                   ; 15/10/2022
11793
11794                                   ;--------------------------------------------------------------------
11795                                   ;
11796                                   ; tim_writ sets the current time
11797                                   ;
11798                                   ; on entry es:[di] has the current time:
11799                                   ;
11800                                   ;   number of days since 1-1-80  (word)
11801                                   ;   minutes (0-59)               (byte)
11802                                   ;   hours (0-23)                 (byte)
11803                                   ;   hundredths of seconds (0-99)  (byte)
11804                                   ;   seconds (0-59)               (byte)
11805                                   ;
11806                                   ; each number has been checked for the correct range.
11807                                   ;
11808                                   ;   NOTE: Any changes in this routine probably require corresponding
11809                                   ;   changes in the version that is built with the power manager driver.
11810                                   ;   See ptime.asm.
11811                                   ;
11812                                   ;--------------------------------------------------------------------
11813
11814                                       ; 08/08/2023 - Retro DOS v4.2 IO.SYS (optimization)
11815                                       ; PCDOS 7.1 IBMBIO.COM - BIOSCODE:03EAh
11816                                   tim_writ:                       ; 2C7h:3DBh = 70h:294Bh
11817 000003E5 268B05                            mov     ax, [es:di]
11818 000003E8 50                                 push    ax              ; daycnt. we need to set this at the very
11819                                                                     ; end to avoid tick windows.
11820 000003E9 803E[8C04]00                       cmp     byte [havecmosclock], 0
11821                                             ;cmp    ds:havecmosclock, 0
11822 000003EE 7423                               jz      short no_cmos_1
11823 000003F0 268A4503                           mov     al, [es:di+3]   ; near indirect       calls
11824                                                                     ; get binary hours
11825                                                                     ; convert to bcd
11826                                             ;call   far [bintobcd]
11827                                             ;;call  ds:bintobcd     ; call far [bintobcd]
11828                                             ; 08/08/2023
11829 000003F4 E8E800                             call    bintobcd
11830 000003F7 88C5                               mov     ch, al          ; ch = bcd hours
11831 000003F9 268A4502                           mov     al, [es:di+2]   ; get binary minutes
11832                                             ;call   far [bintobcd]
11833                                             ;;call  ds:bintobcd     ; convert to bcd
11834 000003FD E8DF00                             call    bintobcd
11835 00000400 88C1                               mov     cl, al          ; cl = bcd minutes
11836 00000402 268A4505                           mov     al, [es:di+5]   ; get binary seconds
11837                                             ;call   far [bintobcd]
11838                                             ;;call  ds:bintobcd
11839 00000406 E8D600                             call    bintobcd
11840
11841 00000409 88C6                               mov     dh, al          ; dh = bcd seconds
11842 0000040B B200                               mov     dl, 0           ; dl = 0 (st) or 1 (dst)
11843 0000040D FA                                 cli
11844 0000040E B403                               mov     ah, 3
11845 00000410 CD1A                               int     1Ah             ; CLOCK - SET REAL TIME       CLOCK (AT,XT286,CONV,PS)
11846                                                                     ; CH = hours in       BCD, CL = minutes in BCD
11847                                                                     ;  DH = seconds in BCD,DL = 01h       if daylight savings, 00h if standard
11848                                                                     ; Return: CMOS clock set
11849 00000412 FB                                 sti
11850                                   no_cmos_1:
11851 00000413 268B4D02                           mov     cx, [es:di+2]
11852 00000417 268B5504                           mov     dx, [es:di+4]
11853                                             ; 17/10/2022
11854 0000041B FF1E[0606]                         call    far [ttticks]
11855                                             ;call   dword ptr ds:ttticks ; call far      [ttticks]
11856                                                                     ; convert time to ticks
11857                                                                     ; cx:dx now has time in ticks
11858 0000041F FA                                 cli                     ; turn off timer
11859 00000420 B401                               mov     ah, 1
11860 00000422 CD1A                               int     1Ah             ; CLOCK - SET TIME OF DAY
11861                                                                     ; CX:DX = clock count
11862                                                                     ; Return: time of day set
11863                                             ;pop    ds:daycnt
11864 00000424 8F06[8904]                         pop     word [daycnt]
11865 00000428 FB                                 sti
11866                                             ;cmp    ds:havecmosclock, 0
11867 00000429 803E[8C04]00                       cmp     byte [havecmosclock], 0
11868 0000042E 7409                               jz      short no_cmos_2
11869
11870                                             ; 08/08/2023
11871                                             ;call   far [daycnttoday]
11872                                             ;;call  ds:daycnttoday ; call far [daycnttoday]
11873                                                                     ; convert to bcd format
11874 00000430 E80700                             call    daycnttoday
11875
11876 00000433 FA                                 cli
11877 00000434 B405                               mov     ah, 5
11878 00000436 CD1A                               int     1Ah             ; CLOCK - SET DATE IN REAL TIME       CLOCK (AT,XT286,CONV,PS)
11879                                                                     ; DL = day in BCD, DH =       month in BCD, CL = year       in BCD
11880                                                                     ; CH = century (19h or 20h)
11881                                                                     ; Return: CMOS clock set
11882 00000438 FB                                 sti
11883                                   no_cmos_2:
11884                                             ; 12/12/2022
11885                                             ; cf=0
11886                                             ;clc
11887 00000439 C3                                 retn
11888
11889                                   ; --------------------------------------------------------------------------
11890
11891                                   ; 08/08/2023 - Retro DOS v4.2 IO.SYS (optimization)
11892                                   ; PCDOS 7.1 IBMBIO.COM - BIOSCODE:0440h
11893                                   %if 1
11894
11895                                   ; CMOS Clock setting support routines used by MSCLOCK.
11896                                   ; Warning!!! This code will be dynamically relocated by MSINIT.
11897
11898                                   daycnttoday:        ; proc near
11899
11900                                   ; entry: [daycnt] = number of days since 1-1-80
11901                                   ;
11902                                   ; return: ch - century in bcd
11903                                   ;     cl - year in bcd
11904                                   ;     dh - month in bcd
11905                                   ;     dl - day in bcd
11906
11907                                         ; 20/12/2023 - Retro DOS v5.0
11908
11909                                         ; 08/08/2023 (ds:) (near proc)
11910                                         ; 16/10/2022 (cs:) (far proc)
11911 0000043A FF36[8904]                         push    word [daycnt] ; save daycnt
11912 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 86C2                           xchg    dl, al
000004CB E81100                         call    bintobcd        ; dh = bcd month, al = year
000004CE 86C6                           xchg    dh, al
000004D0 E80C00                         call    bintobcd        ; cl = bcd year, al = century
000004D3 86C1                           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 86E0                   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
                                ;   ...
```

```
12284                                   ;   restore current head settle value
12285                                   ;
12286                                   ;
12287                                   ;---------------------------------------
12288                                   multrk_on   equ     10000000b       ;user spcified mutitrack=on, or system turns
12289                                                                       ; it on after handling config.sys file as a
12290                                                                       ; default value, if multrk_flag = multrk_off1.
12291                                   multrk_off1 equ     00000000b       ;initial value. no "multitrack=" command entered.
12292                                   multrk_off2 equ     00000001b       ;user specified multitrack=off.
12293                                   
12294                                   ; close data segment, open Bios_Code segment
12295                                   
12296                                   ; 15/10/2022
12297                                   
12298                                   ; BIOSCODE:04A2h (MSDOS 6.21, IO.SYS)
12299                                   
12300                                   ;--------------------------------------------------------------------
12301                                   ;   command jump table
12302                                   ;--------------------------------------------------------------------
12303                                   
12304 0000056E 00                                 db 0
12305                                   
12306                                   ; 11/12/2022
12307                                   %if 0
12308                                   
12309                                   dsktbl:             db 26                   ; 2C7h:4A2h = 70h:2A12h
12310                                                                       ; ((dtbl_siz-1)/2) ; this is the size of the table ; 26
12311                                                       dw 1742h                ; dsk_init
12312                                                       dw 4EBh                 ; media_chk
12313                                                       dw 592h                 ; get_bpb
12314                                                       dw 0D5h                 ; bc_cmderr
12315                                                       dw 857h                 ; dsk_read
12316                                                       dw 83Dh                 ; x_bus_exit
12317                                                       dw 558h                 ; ret_carry_clear
12318                                                       dw 558h                 ; ret_carry_clear
12319                                                       dw 849h                 ; dsk_writ
12320                                                       dw 841h                 ; dsk_writv
12321                                                       dw 558h                 ; ret_carry_clear
12322                                                       dw 558h                 ; ret_carry_clear
12323                                                       dw 0D5h                 ; bc_cmderr
12324                                                       dw 80Ah                 ; dsk_open
12325                                                       dw 81Ah                 ; dsk_close
12326                                                       dw 831h                 ; dsk_rem
12327                                                       dw 558h                 ; ret_carry_clear
12328                                                       dw 558h                 ; ret_carry_clear
12329                                                       dw 558h                 ; ret_carry_clear
12330                                                       dw 0C6Bh                ; do_generic_ioctl
12331                                                       dw 558h                 ; ret_carry_clear
12332                                                       dw 558h                 ; ret_carry_clear
12333                                                       dw 558h                 ; ret_carry_clear
12334                                                       dw 1124h                ; ioctl_getown
12335                                                       dw 1142h                ; ioctl_setown
12336                                                       dw 129Ah                ; ioctl_support_query
12337                                   
12338                                   ;dtbl_siz equ $-dsktbl
12339                                   
12340                                   %endif
12341                                   
12342                                   ; 21/12/2023 - Retro DOS v5.0 (Modified PCDOS 7.1)
12343                                   ; PCDOS 7.1 IBMBIO.COM - BIOSCODE:0579h
12344                                   
12345                                                       ; 21/12/2023 - Retro DOS v5.0
12346                                                       ; 11/12/2022
12347 0000056F 1A                     dsktbl:                   db (dtbl_siz-1)/2     ; 26 ; this is the size of the table
12348 00000570 [4A1A]                                     dw dsk_init
12349 00000572 [B805]                                     dw media_chk
12350 00000574 [5706]                                     dw get_bpb
12351                                                      ;dw bc_cmderr
12352 00000576 [4A0E]                                     dw ioctl_input ; PCDOS 7 ; 21/12/2023
12353 00000578 [7209]                                     dw dsk_read
12354 0000057A [5809]                                     dw x_bus_exit
12355 0000057C [2206]                                     dw ret_carry_clear
12356 0000057E [2206]                                     dw ret_carry_clear
12357 00000580 [6409]                                     dw dsk_writ
12358 00000582 [5C09]                                     dw dsk_writv
12359 00000584 [2206]                                     dw ret_carry_clear
12360 00000586 [2206]                                     dw ret_carry_clear
12361                                                      ;dw bc_cmderr
12362 00000588 [F70D]                                     dw ioctl_output ; PCDOS 7 ; 21/12/2023
12363 0000058A [2909]                                     dw dsk_open
12364 0000058C [3809]                                     dw dsk_close
12365 0000058E [4E09]                                     dw dsk_rem
12366 00000590 [2206]                                     dw ret_carry_clear
12367 00000592 [2206]                                     dw ret_carry_clear
12368 00000594 [2206]                                     dw ret_carry_clear
12369 00000596 [CC0E]                                     dw do_generic_ioctl
12370 00000598 [2206]                                     dw ret_carry_clear
12371 0000059A [2206]                                     dw ret_carry_clear
12372 0000059C [2206]                                     dw ret_carry_clear
12373 0000059E [A713]                                     dw ioctl_getown
12374 000005A0 [C413]                                     dw ioctl_setown
12375 000005A2 [1C15]                                     dw ioctl_support_query
12376                                   
12377                                   dtbl_siz equ $-dsktbl
12378                                   
12379                                   ; =============== S U B   R O U T I N E =======================================
12380                                   
12381                                   ; ---------------------------------------------------------------------------
12382                                   ; setdrive scans through the data structure of bdss, and returns a pointer to
12383                                   ; the one that belongs to the drive specified. carry is set if none exists
12384                                   ; for the drive. Pointer is returned in es:[di]
12385                                   ;
12386                                   ;  AL contains the logical drive number.
12387                                   ; ---------------------------------------------------------------------------
12388                                   
12389                                   SetDrive:
12390                                                       ;les    di, dword ptr ds:start_bds ; Point es:di to first bds
12391 000005A4 C43E[1901]                                 les     di, [start_bds] ; 19/10/2022
12392                                   X_Scan_Loop:
12393 000005A8 26384505                                   cmp     [es:di+5], al
12394 000005AC 7409                                       jz      short X_SetDrv
12395 000005AE 26C43D                                     les     di, [es:di]     ; [es:di+BDS.link] ; Go      to next bds
12396 000005B1 83FFFF                                     cmp     di, 0FFFFh
12397 000005B4 75F2                                       jnz     short X_Scan_Loop
12398 000005B6 F9                                         stc
12399                                   X_SetDrv:
12400 000005B7 C3                                         retn
12401                                   
12402                                   ; ---------------------------------------------------------------------------
12403                                   
12404                                   ; 15/10/2022
12405                                   
12406                                       ; 21/12/2023 - Retro DOS v5.0 (Modified PCDOS 7.1)
12407                                       ; PCDOS 7.1 IBMBIO.COM - BIOSCODE:05C2h
```

```
12408
12409                                 ; ---------------------------------------------------------------------
12410                                 ; if id is f9, have a 96tpi disk else
12411                                 ; if bit 2 is 0 then media is not removable and could not have changed
12412                                 ;  otherwise if within 2 secs of last disk operation media could not
12413                                 ;    have changed, otherwise dont know if media has changed
12414                                 ; ---------------------------------------------------------------------
12415
12416                                 media_chk:                      ; 2C7h:4EBh = 70h:2A5Bh
12417 000005B8 E8E9FF                           call    SetDrive
12418 000005BB BE0100                           mov     si, 1
12419                                           ; 21/12/2023
12420 000005BE 26F6454001                       test    byte [es:di+40h], 1
12421                                           ;test   byte [es:di+24h], 1 ; [es:di+BDS.flags+1]
12422                                                                   ; fchanged_by_format
12423 000005C3 7415                           jz      short WeAreNotFakingIt
12424                                           ; 21/12/2023
12425 000005C5 26806540FE                     and     byte [es:di+40h], 0FEh
12426                                           ; 12/12/2022
12427                                           ;and    byte [es:di+24h], 0FEh ; ~fchanged_by_format
12428                                           ;;and   word [es:di+23h], 0FEFFh ; [es:di+BDS.flags]
12429                                                                   ; ~fchanged_by_format ;       reset flag
12430 000005CA C606[1E01]FF                   mov     byte [tim_drv], 0FFh ; -1
12431                                                                   ; Ensure that we ask the rom if media has changed
12432                                           ; 21/12/2023
12433 000005CF 26F6453F01                     test    byte [es:di+3Fh], 1
12434                                           ;test   byte [es:di+23h], 1 ; [es:di+BDS.flags]
12435                                                                   ; fnon_removable
12436 000005D4 740B                           jz      short wehaveafloppy
12437                                           ;mov    si, 0FFFFh      ; Indicate media changed
12438                                           ; 11/08/2023
12439 000005D6 F7DE                           neg     si              ; PCDOS 7.1 IBMBIO.COM - BIOSCODE:05E0h
12440 000005D8 EB2B                           jmp     short Media_Done ; Media_Done
12441                                 ; ---------------------------------------------------------------------------
12442
12443                                 WeAreNotFakingIt:
12444                                           ; 21/12/2023
12445 000005DA 26F6453F01                     test    byte [es:di+3Fh], 1
12446                                           ;test   byte [es:di+BDS.flags], fnon_removable
12447                                           ;test   byte [es:di+23h], 1
12448 000005DF 7524                           jnz     short Media_Done
12449                                 wehaveafloppy:
12450                                           ;xor    si, si ; 0      ; Presume "I don't know"
12451                                           ; 11/08/2023
12452 000005E1 4E                             dec     si ; 0          ; PCDOS 7.1 IBMBIO.COM - BIOSCODE:05EBh
12453
12454                                           ; If we have a floppy with changeline support, we ask the ROM
12455                                           ; to determine if media has changed. We do not perform the
12456                                           ; 2 second check for these drives.
12457
12458 000005E2 803E[7700]00                    cmp     byte [fhave96], 0       ; Do we have changeline       support?
12459 000005E7 740A                           jz      short mChk_NoChangeLine       ; Brif not
12460 000005E9 E83615                         call    mediacheck      ;  Call into removable routine
12461 000005EC 7236                           jb      short err_exitj
12462 000005EE E89A16                         call    haschange
12463 000005F1 7512                           jnz     short Media_Done
12464                                 mChk_NoChangeLine:
12465                                           ; If we come here, we have a floppy with no changeline support
12466
12467 000005F3 BE0100                         mov     si, 1           ; Presume no change
12468 000005F6 A0[1E01]                       mov     al, [tim_drv]   ; Last drive accessed
12469 000005F9 263A4504                       cmp     al, [es:di+4]   ; [es:di+BDS.drivenum]
12470                                                                   ; Is drive of last access the same?
12471 000005FD 7505                           jnz     short Media_Unk       ; No, then "i don't know"
12472 000005FF E82800                         call    Check_Time_Of_Access
12473 00000602 EB01                           jmp     short Media_Done
12474                                 ; ---------------------------------------------------------------------------
12475
12476                                 Media_Unk:
12477 00000604 4E                             dec     si              ; 0 ; Return "I don't know"
12478
12479                                           ; SI now contains the correct value for media change.
12480                                           ; Clean up the left overs
12481                                 Media_Done:
12482                                           ; 19/10/2022
12483 00000605 06                             push    es
12484 00000606 C41E[1200]                     les     bx, [ptrsav]
12485 0000060A 2689770E                       mov     [es:bx+0Eh], si       ; [es:bx+trans]
12486 0000060E 07                             pop     es
12487 0000060F 09F6                           or      si, si
12488 00000611 790F                           jns     short ret_carry_clear ;       volidok
12489 00000613 803E[7700]00                   cmp     byte [fhave96], 0
12490 00000618 7403                           jz      short mChk1_NoChangeLine ; Brif      no changeline support
12491 0000061A E80016                         call    media_set_vid
12492                                 mChk1_NoChangeLine:
12493 0000061D C606[1E01]FF                   mov     byte [tim_drv], 0FFh ; -1
12494                                                                   ; Make sure we ask rom for media check
12495                                 ret_carry_clear:
12496 00000622 F8                             clc                     ; volidok
12497 00000623 C3                             retn
12498                                 ; ---------------------------------------------------------------------------
12499
12500                                 err_exitj:
12501 00000624 E88207                         call    maperror        ; guaranteed to       set carry
12502                                 ret81:
12503 00000627 B481                           mov     ah, 81h         ; return error status
12504 00000629 C3                             retn                    ; return with carry set
12505
12506                                 ; =============== S U B   R O U T I N E =======================================
12507
12508                                 ; ---------------------------------------------------------------------------
12509                                 ; perform a check on the time passed since the last access for this physical
12510                                 ; drive.
12511                                 ; we are accessing the same drive. if the time of last successful access was
12512                                 ; less than 2 seconds ago, then we may presume that the disk was not changed.
12513                                 ; returns in si:
12514                                 ;   0 - if time of last access was >= 2 seconds
12515                                 ;   1 - if time was < 2 seconds (i.e no media change assumed)
12516                                 ; registers affected ax,cx,dx, flags.
12517                                 ;
12518                                 ;   assume es:di -> bds, ds->Bios_Data
12519                                 ; ---------------------------------------------------------------------------
12520
12521                                           ; 21/12/2023 - Retro DOS v5.0 IBMBIO.COM
12522                                           ; 19/10/2022
12523                                 Check_Time_Of_Access:
12524 0000062A BE0100                         mov     si, 1           ; presume no change.
12525 0000062D E801FF                         call    GetTickCnt      ; cx:dx is the elapsed time
12526                                           ; 21/12/2023
12527 00000630 268B4579                       mov     ax, [es:di+79h]
12528                                           ;mov    ax, [es:di+47h]       ; [es:di+BDS.tim_lo]
12529                                                                   ; get stored time
12530 00000634 29C2                           sub     dx, ax
12531                                           ; 21/12/2023
```

```
12532 00000636 268B457B                         mov     ax, [es:di+7Bh]
12533                                           ;mov    ax, [es:di+49h]       ; [es:di+BDS.tim_hi]
12534 0000063A 19C1                             sbb     cx, ax
12535                                           ; 11/08/2023
12536                                           ; PCDOS 7.1 IBMBIO.COM - BIOSCODE:0646h
12537                                           ;mov    al, [accesscount]
12538 0000063C 7515                             jnz     short timecheck_unk ; cx<>0 => >1 hour
12539 0000063E 09D2                             or      dx, dx          ; time must pass
12540 00000640 750C                             jnz     short timepassed ; yes, examine max value
12541                                           ; 11/08/2023
12542                                           ;inc    al
12543                                           ;cmp    al, 5
12544                                           ;;inc   byte [accesscount]
12545                                           ;;cmp   byte [accesscount], 5
12546                                           ;                       ; if count is less than threshold, ok
12547                                           ;jb     short timecheck_ret
12548                                           ;;dec   byte [accesscount] ; don't let the count wrap
12549                                           ; 11/08/2023
12550                                           ;dec    al
12551                                           ;jmp    short timecheck_unk ; "i don't know" if media changed
12552                                           ; 11/08/2023
12553 00000642 803E[1D01]04                     cmp     byte [accesscount], 4
12554 00000647 730A                             jnb     short timecheck_unk
12555 00000649 FE06[1D01]                       inc     byte [accesscount]
12556 0000064D C3                               retn
12557
12558                                   ; ---------------------------------------------------------------------------
12559
12560                                   timepassed:
12561 0000064E 83FA24                             cmp     dx, 36          ; 18*2 ; 18.2 tics per second.
12562                                                                     ; min elapsed time? (2 seconds)
12563 00000651 7601                               jbe     short timecheck_ret ; yes, presume no change
12564
12565                                             ; everything indicates that we do not know what has happened.
12566                                   timecheck_unk:
12567 00000653 4E                                 dec     si              ; presume i don't know
12568                                   timecheck_ret:
12569                                             ; 11/08/2023
12570                                             ;mov    [accesscount], al
12571 00000654 C3                                 retn
12572
12573                                   ; ---------------------------------------------------------------------------
12574                                   ; 15/10/2022
12575                                   Err_Exitj2:
12576 00000655 EBCD                               jmp     short err_exitj
12577
12578                                   ; ---------------------------------------------------------------------------
12579
12580                                   ; 15/10/2022
12581
12582                                   ; ============================================================================
12583                                   ; Build a valid bpb for the disk in the drive.
12584                                   ; ============================================================================
12585
12586                                             ; 21/12/2023 - Retro DOS v5.0 IBMBIO.COM
12587                                             ; 19/10/2022
12588                                   get_bpb:                          ; 2C7h:592h = 70h:2B02h
12589 00000657 268A25                             mov     ah, [es:di]     ; get fat id byte read by dos
12590 0000065A E847FF                             call    SetDrive        ; get the correct bds for the drive
12591                                             ; 21/12/2023
12592 0000065D 26F6453F01                         test    byte [es:di+3Fh], 1
12593                                             ;test   byte [es:di+23h], 1 ; [es:di+BDS.flags]
12594                                                                     ; fnon_removable
12595 00000662 7523                               jnz     short already_gotbpb ; no need to build     for fixed disks
12596
12597                                             ; let's set the default value for volid,vol_serial,
12598                                             ; filesys_id in bds table
12599
12600 00000664 E83600                             call    clear_ids
12601                                             ;mov    ds:set_id_flag,       1 ; indicate to       set system id in bds
12602 00000667 C606[9B04]01                       mov     byte [set_id_flag], 1
12603 0000066C E86700                             call    GetBp           ; build a bpb if necessary
12604 0000066F 72B6                               jb      short ret81
12605                                             ;cmp    ds:set_id_flag,       2 ; already, volume_label set from boot
12606 00000671 803E[9B04]02                       cmp     byte [set_id_flag], 2
12607                                             ;mov    ds:set_id_flag,       0 ; record to bds table?
12608 00000676 C606[9B04]00                       mov     byte [set_id_flag], 0
12609 0000067B 740A                               jz      short already_gotbpb ; do not set it again from     root dir
12610                                                                     ; otherwise, conventional boot record
12611                                             ;cmp    ds:fhave96, 0  ; do we have changeline       support?
12612 0000067D 803E[7700]00                       cmp     byte [fhave96], 0
12613 00000682 7403                               jz      short already_gotbpb ; brif not
12614 00000684 E80A16                             call    set_volume_id
12615                                   already_gotbpb:
12616 00000687 83C706                             add     di, 6           ; BDS.BPB
12617                                                                     ; return the bpb from the current bds
12618
12619                                   ;           fall into setptrsav, es:di -> result
12620
12621                                   ; ---------------------------------------------------------------------------
12622
12623                                   ; 15/10/2022
12624
12625                                   ; ============================================================================
12626                                   ;Setptrsav is also jumped to from dsk_init (msbio2.asm). In both cases, the
12627                                   ;pointer to be returned is in es:di. We were incorrectly returning ds:di.
12628                                   ;Note that this works in most cases because most pointers are in Bios_Data.
12629                                   ;It fails, for instance, when we install an external drive using driver.sys
12630                                   ;because then the BDS segment is no longer Bios_Data.
12631                                   ;NB: It is fine to corrupt cx because this is not a return value and anyway
12632                                   ;this returns to Chardev_entry (msbio1.asm) where all registers are
12633                                   ;restored before returning to the caller.
12634                                   ; ============================================================================
12635
12636                                   ; 21/12/2023
12637                                   %if 0
12638                                             ; 19/10/2022
12639                                   SetPtrSav:  ; return point for dsk_init
12640                                             mov     cx, es          ; save es
12641                                             ;les    bx, ds:ptrsav
12642                                             les     bx, [ptrsav]
12643                                             mov     [es:bx+0Dh], ah         ; [es:bx+media]
12644                                             mov     [es:bx+12h], di         ; [es:bx+count]
12645                                             mov     [es:bx+14h], cx         ; [es:bx+count+2]
12646                                             clc
12647                                             retn
12648                                   %endif
12649                                             ; 21/12/2023 - Retro DOS v5.0 (Modified PCDOS 7.1)
12650                                             ; PCDOS 7.1 IBMBIO.COM - BIOSCODE:0698h
12651                                   SetPtrSav:
12652                                             ; return point for dsk_init
12653 0000068A 1E                                 push    ds
12654                                             ;lds    bx, ds:ptrsav
12655 0000068B C51E[1200]                         lds     bx, [ptrsav]
```

```
12656 0000068F 88670D                          mov     [bx+0Dh], ah   ; [bx+media]
12657 00000692 897F12                          mov     [bx+12h], di   ; [bx+count]
12658 00000695 8C4714                          mov     [bx+14h], es   ; [bx+count+2]
12659 00000698 1E                              push    ds
12660 00000699 07                              pop     es
12661 0000069A 1F                              pop     ds
12662 0000069B F8                              clc
12663 0000069C C3                              retn
12664
12665                                  ; ============== S U B   R O U T I N E =======================================
12666
12667                                  ; 15/10/2022
12668
12669                                  ; ------------------------------------------------------
12670                                  ; clear ids in bds table. only applied for floppies.
12671                                  ;input:  es:di -> bds table
12672                                  ;   assumes ds: -> Bios_Data
12673                                  ;output: volid set to "NO NAME    "
12674                                  ;    vol_serial set to 0.
12675                                  ;    filesys_id set to "FAT12   " or "FAT16   "
12676                                  ;      depending on the flag fatsize in bds.
12677                                  ;
12678                                  ;   trashes si, cx
12679                                  ; ------------------------------------------------------
12680
12681                                  ;size_of_EXT_BOOT_VOL_LABEL equ 11
12682                                  ;size_of_EXT_SYSTEM_ID equ 8
12683
12684                                  ; 11/09/2023
12685                                  ; 14/08/2023
12686                                  ;BDS.fatsiz equ 1Fh
12687                                  ; 21/12/2023
12688                                  ;BDS.fatsiz equ 59
12689
12690                                          ; 22/12/2023
12691                                          ; 21/12/2023 - Retro DOS v5.0 (Modified PCDOS 7.1)
12692                                  clear_ids:
12693                                          ;mov    al, [es:di+1Fh] ; mov al,[es:di+BDS.fatsiz]
12694                                          ; 21/12/2023 - Retro DOS v5.0 (PCDOS 7.1 IBMBIO.COM BugFix)
12695 0000069D 268A5D3B                       mov     bl, [es:di+3Bh] ; mov bl,[es:di+BDS.fatsiz]; *+
12696                                  clear_ids_x:
12697                                          ; 21/12/2023
12698                                          ; (PCDOS 7.1 IBMBIO.COM - BIOSCODE:06ABh)
12699                                          ; 11/09/2023
12700                                          ; (MSDOS 5.0 IO.SYS - BIOSCODE:05D9h)
12701 000006A1 57                             push    di
12702 000006A2 31C9                            xor     cx, cx         ; no serial number
12703                                          ; 21/12/2023
12704 000006A4 26898D8900                      mov     [es:di+89h], cx        ; [es:di+BDS.vol_serial]
12705 000006A9 26898D8B00                      mov     [es:di+8Bh], cx        ; [es:di+BDS.vol_serial+2]
12706                                          ;mov    [es:di+57h], cx        ; [es:di+BDS.vol_serial]
12707                                          ;mov    [es:di+59h], cx        ; [es:di+BDS.vol_serial+2]
12708
12709                                          ; BUGBUG - there's a lot in common here and with
12710                                          ; mov_media_ids.. see if we can save some space by
12711                                          ; merging them... jgl
12712
12713                                          ;mov    cx, 11         ; size_of_EXT_BOOT_VOL_LABEL
12714                                          ; 10/12/2022
12715 000006AE B10B                          mov     cl, 11 ; cx = 11
12716
12717                                          ;;mov   si, offset vol_no_name ; "NO NAME    "
12718                                          ;mov    si, vol_no_name        ; 19/10/2022
12719                                          ; 22/12/2023
12720                                          ;mov    si, offset nul_vid ; "NO NAME    "
12721 000006B0 BE[6305]                       mov     si, nul_vid
12722
12723                                          ; 21/12/2023
12724 000006B3 83C77D                          add     di, 125
12725                                          ;add    di, 75         ; BDS.volid
12726
12727                                          ;rep movsb
12728                                          ; 21/12/2023
12729                                          ;rep movs byte ptr es:[di], byte ptr cs:[si] ; cs rep movsb
12730                                          ; 26/12/2023
12731                                          ;cs     ; vol_no_name is in BIOSCODE segment
12732                                          ;rep movsb
12733 000006B6 F3                             rep
12734 000006B7 2E                             cs
12735 000006B8 A4                             movsb
12736
12737                                          ; 11/09/2023 (BugFix, DI is not start addr of BDS structure here)
12738                                          ;;test  byte [es:di+BDS.fatsiz], fbig
12739                                          ; (MSDOS 5.0 IO.SYS - BIOSCODE:05EFh)
12740                                          ;test   byte [es:di+1Fh], 40h
12741                                          ; 21/12/2023 - Retro DOS v5.0
12742                                          ; (PCDOS 7.1 IBMBIO.COM - BIOSCODE:06C3h)
12743                                          ;test   byte [es:di+59], 20h
12744                                          ; (here, es:di points to the BDS offset +136)
12745                                          ; purpose: test byte [es:di+BDS.fatsiz], fbigbig
12746                                          ; applied: test byte [es:BDS.fatsiz+136], fbigbig -BUG!-
12747
12748                                          ; (PCDOS 7.1 BUG note: 26/06/2023 - Erdogan Tan)
12749                                          ;; ! NOTE - 11/08/2023 - Erdogan Tan (Retro DOS v4.2 IO.SYS bugfix)
12750                                          ; Microsoft/IBM code has a bug here because the BDS's
12751                                          ; .volid and .filesys_id fields will be reset
12752                                          ; (to their default text) according to 'BDS.fatsiz' flags
12753                                          ; at the BDS offset 59 but current (this) code checks flags
12754                                          ; at ES:DI+59 while DI points the BDS offset 136!? ; (PCDOS 7.1)
12755                                          ;; at the BDS offset 31 but current (this) code checks flags
12756                                          ;; at ES:DI+31 while DI points the BDS offset 86!? ; (MSDOS 6.22)
12757                                          ;
12758                                          ; Correct Code:
12759                                          ; ;test byte [ES:59],20h or [ES:BDS.fatsiz],fbigbig  ; (PCDOS 7.1)
12760                                          ; ;;test byte [ES:31],40h or [ES:BDS.fatsiz],fbig  ; (MSDOS 6.22)
12761                                          ; 11/09/2023
12762                                          ; (before 'rep movsb') 'mov al,[es:di+BDS.Fatsiz]' and then
12763                                          ; (after 'rep movsb') 'test al,fbig' (AL is free/proper to use here)
12764                                          ;
12765                                          ; Same BUG is existing in MSDOS 6.22 IO.SYS - BIOSCODE:05EFh
12766                                          ; and in Windows ME IO.SYS - BIOSCODE:0E1Ah as 'test byte [es:di+59],20h'
12767
12768                                          ;
12769                                          ; (Why this bug did not affect MSDOS and PCDOS 7.x applications:
12770                                          ; 'clear_ids' is used for floppy disks only and the default
12771                                          ; option of 'clear_ids' is FAT12 volid and filesys_id text
12772                                          ; when the flag bit has wrong value for FAT16/40h or FAT32/20h.)
12773
12774                                  ; 21/12/2023 - Retro DOS v5.0
12775                                  ;mov    si, offset fat_32_id ; "FAT32   "
12776 000006B9 BE[5B05]                mov     si, fat_32_id
12777
12778                                  ; 21/12/2023
12779                                  ; BugFix (of the PCDOS 7.1 IBMBIO.COM BUG) ; *+
```

```
                                        ;test  bl, fbigbig ; FAT32 flag
000006BC F6C320                         test   bl, 20h ; * ; BL = [es:BDS.fatsiz] = [es:59]
000006BF 750B                           jnz    short ci_bigfat

                                        ;mov   si, offset fat_16_id ; "FAT16 "
000006C1 BE[5305]                       mov    si, fat_16_id  ; 19/10/2022

                                        ; 21/12/2023
                                        ; !BUG! (PCDOS 7.1 IBMBIO.COM BIOSCODE:06CDh)
                                        ;test  byte [es:di+59], 40h ; [es:di+BDS.fatsiz], fbig
                                        ; BugFix ; *+
                                        ;test  bl, fbig ; FAT16 flag
000006C4 F6C340                         test   bl, 40h ; * ; Retro DOS v5.0
                                        ;;test al, 40h ; * ; Retro DOS v4.2
000006C7 7503                           jnz    short ci_bigfat

                                        ;mov   si, offset fat_12_id ; "FAT12 "
000006C9 BE[4B05]                       mov    si, fat_12_id  ; 19/10/2022
                                ci_bigfat:
                                        ;mov   cx, 8          ; size_of_EXT_SYSTEM_ID
                                        ; 10/12/2022
000006CC B108                           mov    cl, 8 ; cx = 8
000006CE 83C705                         add    di, 5          ; (BDS.filesys_id-BDS.volid)-size_of_EXT_BOOT_VOL_LABEL
                                                               ; filesys_id field
                                        ;rep movsb
                                        ; 21/12/2023 - Retro DOS v5.0
                                        ;rep movs byte ptr es:[di], byte ptr cs:[si] ; 0F3h,2Eh,0A4h
                                        ; 26/12/2023
                                        ;cs    ; fat32_id, fat16_id and fat12_id are in BIOSCODE segment
                                        ;rep movsb
000006D1 F3                             rep
000006D2 2E                             cs
000006D3 A4                             movsb

000006D4 5F                             pop    di             ; restore bds pointer
                                getret_exit:                ; 21/12/2023
000006D5 C3                             retn

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

                                ; 15/10/2022

                                ; ---------------------------------------------------------------------------
                                ;   getbp - return bpb from the drive specified by the bds.
                                ;       if the return_fake_bpb flag is set, then it does nothing.
                                ;       note that we never come here for fixed disks.
                                ;       for all other cases,
                                ;         - it reads boot sector to pull out the bpb
                                ;         - if no valid bpb is found, it then reads the fat sector,
                                ;           to get the fat id byte to build the bpb from there.
                                ;
                                ;   inputs: es:di point to correct bds.
                                ;
                                ;   outputs:       fills in bpb in current bds if valid bpb or fat id on disk.
                                ;           carry set, and al=7 if invalid disk.
                                ;           carry set and error code in al if other error.
                                ;           if failed to recognize the boot record, then will set the
                                ;           set_id_flag to 0.
                                ;           this routine will only work for a floppy diskette.
                                ;                for a fixed disk, it will just return.
                                ;
                                ;   ****** Note: getbp is a clone of getbp which uses the newer
                                ;     segment definitions. It should be migrated towards.
                                ;      now es:di has the bds, ds: has Bios_Data
                                ; ---------------------------------------------------------------------------

                                        ; 29/12/2023
                                        ; 21/12/2023 - Retro DOS v5.0 (Modified PCDOS 7.1)
                                GetBp:
                                        ; if returning fake bpb then return bpb as is.
                                        ; 21/12/2023
000006D6 26F6453F05                     test   byte [es:di+3Fh], 5 ; PCDOS 7.1
                                        ;test  byte [es:di+BDS.flags], return_fake_bpb|fnon_removable
                                        ;test  byte [es:di+23h], 5 ; MSDOS 6.22 (& MSDOS 5.0)
                                        ;jz    short getbp1   ; getbp1
                                        ;jmp   getret_exit
                                        ; 21/12/2023
000006DB 75F8                           jnz    short getret_exit
                                ; ---------------------------------------------------------------------------
                                getbp1:
000006DD 51                             push   cx
000006DE 52                             push   dx
000006DF 53                             push   bx

                                        ; attempt to read in boot sector and determine bpb.
                                        ; we assume that the 2.x and greater dos disks all
                                        ; have a valid boot sector.

000006E0 E8CF00                         call   readbootsec
000006E3 720A                           jb     short getbp_err_ret_brdg ; carry set if there was error.
000006E5 09DB                           or     bx, bx         ; bx is 0 if boot sector is valid.
000006E7 7509                           jnz    short dofatbpb
000006E9 E81401                         call   movbpb         ; move bpb into       registers
                                        ;jmp   short Has1
                                        ; 21/12/2023 - Retro DOS v5.0 (PCDOS 7.1) IBMBIO.COM
000006EC E9B500                         jmp    getret
                                ; ---------------------------------------------------------------------------

                                getbp_err_ret_brdg:
000006EF E9B600                         jmp    getbp_err_ret
                                ; ---------------------------------------------------------------------------

                                        ; we have a 1.x diskette. In this case read in the fat ID byte
                                        ; and fill in bpb from there.
                                dofatbpb:
000006F2 E8B401                         call   readfat        ; puts media descriptor      byte in ah
000006F5 72F8                           jb     short getbp_err_ret_brdg
                                        ;cmp   ds:fhave96, 0  ;  changeline support available?
000006F7 803E[7700]00                   cmp    byte [fhave96], 0 ; 19/10/2022
000006FC 7403                           jz     short bpb_nochangeline ; brif not
000006FE E83115                         call   hidensity      ; may not return! May add sp, 2      and
                                                              ; jump to has1!!!!!! or      has720K
                                bpb_nochangeline:         ; test for a valid 3.5" medium
                                        ; 21/12/2023 - Retro DOS v5.0
00000701 26807D3E02                     cmp    byte [es:di+3Eh], 2
                                        ;cmp   byte [es:di+22h], 2 ; [es:di+BDS.formfactor]
                                                              ; ffSmall
00000706 7512                           jnz    short is_floppy
00000708 80FCF9                         cmp    ah, 0F9h       ; is it a valid fat id byte for       3.5" ?
0000070B 7512                           jnz    short got_unknown_medium
                                Has720K:
                                        ; 21/12/2023
                                        ;;mov  bx, offset sm92 ; pointer to correct bpb
                                        ;mov   bx, sm92       ; 19/10/2022
```

```
12904
12905                                   ; es points to segment of bds. the following should be modified
12906                                   ; to get spf,csec,spau,spt correctly. it had been wrong if
12907                                   ; driver.sys is loaded since the bds is inside the driver.sys.
12908
12909                                   ; 21/12/2023
12910                                   ;; 10/12/2022
12911                                   ;;mov   al, [bx+0]     ; [bx+bpbtype.spf]
12912                                   ;; 21/12/2022
12913                                   ;mov    al, [bx]
12914                                   ;mov    cx, [bx+3]     ; [bx+bpbtype.csec]
12915                                   ;mov    dx, [bx+5]     ; [bx+bpbtype.spau]
12916                                   ;mov    bx, [bx+1]     ; [bx+bpbtype.spt]
12917                                   ;; 19/10/2022 - Temporary !
12918                                   ;;db    8Ah, 87h, 0, 0 ; mov al, [bx+0]
12919                                   ;;db    8Bh, 8Fh, 3, 0 ; mov cx, [bx+3]
12920                                   ;;db    8Bh, 97h, 5, 0 ; mov dx, [bx+5]
12921                                   ;;db    8Bh, 9Fh, 1, 0 ; mov bx, [bx+1]
12922
12923                                   ; 21/12/2023 - Retro DOS v5.0
12924 0000070D B003                     mov     al, 3          ; bpbtype.sbf = 3
12925 0000070F B9A005                   mov     cx, 1440       ; bpbtype.csec = 1440
12926 00000712 BA0202                   mov     dx, 202h       ; dl = bpbtype.spau = 2
12927                                                          ; dh = bpbtype.chead = 2
12928 00000715 BB0970                   mov     bx, 7009h      ; bl = bpbtype.spt = 9
12929                                                          ; bh = bpbtype.dire = 112
12930 00000718 EB30                     jmp     short Has1
12931                                 ; ---------------------------------------------------------------------------
12932
12933                                 is_floppy:                ; must be a 5.25" floppy if we come here
12934 0000071A 80FCF8                           cmp     ah, 0F8h       ; valid media?? (0F8h-0FFh)
12935                                           ;jb     short got_unknown_medium
12936                                           ; 21/12/2023
12937 0000071D 730A                             jnb     short chk_160K
12938                                 ; ---------------------------------------------------------------------------
12939                                           ; 21/12/2023
12940                                           ; we have a 3.5" diskette for which we cannot build a bpb.
12941                                           ; we do not assume any type of bpb for this medium.
12942                                 got_unknown_medium:
12943                                           ;mov    ds:set_id_flag,       0
12944 0000071F C606[9B04]00                     mov     byte [set_id_flag], 0
12945 00000724 B007                             mov     al, 7
12946 00000726 F9                               stc
12947 00000727 EB7B                             jmp     short getret
12948                                 ; ---------------------------------------------------------------------------
12949                                 chk_160K:
12950 00000729 B001                             mov     al, 1          ; set number of       fat sectors
12951 0000072B BB0840                           mov     bx, 16392      ; 64*256+8
12952                                                                  ; set dir entries and sector max
12953 0000072E B94001                           mov     cx, 320        ; 40*8
12954                                                                  ; set size of drive
12955 00000731 BA0101                           mov     dx, 257        ; 01*256+1
12956                                                                  ; set head limit and sec/all unit
12957                                           ; 21/12/2023
12958                                           ;mov    al, 1          ; bpbtype.sbf = 1
12959                                           ;mov    bx, 4008h      ; bl = bpbtype.spt = 8
12960                                           ;                      ; bh = bpbtype.dire = 64
12961                                           ;mov    cx, 140h       ; bpbtype.csec = 320
12962                                           ;mov    dx, 101h       ; dl = bpbtype.spau = 1
12963                                           ;                      ; dh = bpbtype.chead = 1
12964
12965 00000734 F6C402                           test    ah, 2          ; test for 8 or       9 sector
12966 00000737 7505                             jnz     short has8     ; nz = has 8 sectors
12967
12968                                           ; 29/12/2023
12969                                           ;inc    al      ; 2    ; inc number of       fat sectors
12970                                           ;inc    bl      ; 9    ; inc sector max
12971 00000739 40                               inc     ax
12972 0000073A 43                               inc     bx
12973
12974                                           ;add    cx, 40         ; increase size       (to 360)
12975                                           ; 18/12/2022
12976 0000073B 80C128                           add     cl, 40  ; 28h  ; 180K (360 sectors)
12977                                 has8:
12978 0000073E F6C401                           test    ah, 1          ; test for 1 or       2 heads
12979 00000741 7407                             jz      short Has1     ; jz = 1 head
12980 00000743 01C9                             add     cx, cx         ; double size of disk
12981 00000745 B770                             mov     bh, 112        ; increase number of directory entries
12982 00000747 FEC6                             inc     dh      ; 2    ; inc sec/all unit
12983                                           ; 29/12/2023
12984                                           ;inc    dl      ; 2    ; inc head limit
12985 00000749 42                               inc     dx
12986                                 Has1:
12987                                           ; 02/09/2023 (PCDOS 7.1, IBMBIO.COM - BIOSCODE:0754h)
12988 0000074A 1E                               push    ds
12989 0000074B 06                               push    es
12990 0000074C 1F                               pop     ds
12991
12992                                           ;mov    [es:di+8], dh  ; [es:di+BDS.secperclus]
12993                                           ;mov    [es:di+0Ch], bh        ; [es:di+BDS.direntries]
12994                                           ;mov    [es:di+0Eh], cx        ; [es:di+BDS.totalsecs16]
12995                                           ;mov    [es:di+10h], ah        ; [es:di+BDS.media]
12996                                           ;mov    [es:di+11h], al        ; [es:di+BDS.fatsecs]
12997                                           ;mov    [es:di+13h], bl        ; [es:di+BDS.secpertrack]
12998                                           ;mov    [es:di+15h], dl        ; [es:di+BDS.heads]
12999
13000 0000074D 887508                           mov     [di+8], dh     ; [di+BDS.secperclus]
13001 00000750 30F6                             xor     dh, dh
13002 00000752 895515                           mov     [di+15h], dx   ; [di+BDS.heads]
13003 00000755 88FA                             mov     dl, bh
13004 00000757 89550C                           mov     [di+0Ch], dx   ; [di+BDS.direntries]
13005 0000075A 894D0E                           mov     [di+0Eh], cx   ; [di+BDS.totalsecs16]
13006 0000075D 894D1B                           mov     [di+1Bh], cx   ; [di+BDS.totalsecs32]
13007 00000760 886510                           mov     [di+10h], ah   ; [di+BDS.media]
13008 00000763 88C2                             mov     dl, al
13009 00000765 895511                           mov     [di+11h], dx   ; [di+BDS.fatsecs]
13010 00000768 88DA                             mov     dl, bl
13011 0000076A 895513                           mov     [di+13h], dx   ; [di+BDS.secpertrack]
13012
13013                                           ; the BDS_BPB.BPB_HIDDENSECTORS+2 field and the
13014                                           ; BDS_BPB.BPB_BIGTOTALSECTORS field need to be set
13015                                           ; to 0 since this code is for floppies
13016
13017                                           ; 18/12/2022
13018                                           ;mov    word [es:di+19h], 0 ; [es:di+BDS.hiddensecs+2]
13019                                           ;mov    word [es:di+17h], 0 ; [es:di+BDS.hiddensecs]
13020                                           ;mov    word [es:di+1Dh], 0 ; [es:di+BDS.totalsecs32+2]
13021                                           ; 18/12/2022
13022 0000076D 29C9                             sub     cx, cx ; 0
13023                                           ;mov    [es:di+19h], cx ; 0 ; [es:di+BDS.hiddensecs+2]
13024                                           ;mov    [es:di+17h], cx ; 0 ; [es:di+BDS.hiddensecs]
13025                                           ;mov    [es:di+1Dh], cx ; 0 ; [es:di+BDS.totalsecs32+2]
13026
13027                                           ; 02/09/2023
```

```
13028 0000076F 894D19                          mov     [di+19h], cx ; 0 ; [di+BDS.hiddensecs+2]
13029 00000772 894D17                          mov     [di+17h], cx ; 0 ; [di+BDS.hiddensecs]
13030 00000775 894D1D                          mov     [di+1Dh], cx ; 0 ; [di+BDS.totalsecs32+2]
13031
13032                                          ; 21/12/2023 - Retro DOS v5.0
13033 00000778 894D1F                          mov     [di+1Fh], cx    ; [di+BDS.fatsecs32] ; BPB_FATSz32
13034 0000077B 894D21                          mov     [di+21h], cx    ; [di+BDS.fatsecs32+2]
13035 0000077E 894D27                          mov     [di+27h], cx    ; [di+BDS.rootdirclust]
13036 00000781 894D29                          mov     [di+29h], cx    ; [di+BDS.rootdirclust+2]
13037 00000784 894D2F                          mov     [di+2Fh], cx    ; [di+BDS.reserved]
13038                                                                  ;     BPB_Reserved (12 zero bytes)
13039 00000787 894D31                          mov     [di+31h], cx
13040 0000078A 894D33                          mov     [di+33h], cx
13041 0000078D 894D35                          mov     [di+35h], cx
13042 00000790 894D37                          mov     [di+37h], cx
13043 00000793 894D39                          mov     [di+39h], cx
13044 00000796 894D23                          mov     [di+23h], cx    ; [di+BDS.extflags] ; BPB_ExtFlags
13045 00000799 894D25                          mov     [di+25h], cx    ; [di+BDS.fsver] ; BPB_FSVer
13046
13047 0000079C 49                              dec     cx              ; -1 ; 0FFFFFFFFh
13048 0000079D 894D2B                          mov     [di+2Bh], cx    ; [di+BDS.fsinfo] ; BPB_FSInfo
13049 000007A0 894D2D                          mov     [di+2Dh], cx    ; [di+BDS.bkbootsec] ; BPB_BkBootSec
13050
13051 000007A3 1F                              pop     ds ; 02/09/2023
13052                                  getret:
13053 000007A4 5B                              pop     bx
13054 000007A5 5A                              pop     dx
13055 000007A6 59                              pop     cx
13056                                  ;getret_exit:              ; 21/12/2023
13057 000007A7 C3                              retn
13058                                  ; ---------------------------------------------------------------------------
13059
13060                                  getbp_err_ret:     ; before doing anything else, set set_id_flag to 0.
13061                                              ;mov   ds:set_id_flag,       0
13062                                              ; 19/10/2022
13063 000007A8 C606[9B04]00                      mov     byte [set_id_flag], 0
13064 000007AD E8F905                            call    maperror
13065 000007B0 EBF2                              jmp     short getret
13066                                  ; ---------------------------------------------------------------------------
13067                                  ; 21/12/2023
13068                                  ;           ; we have a 3.5" diskette for which we cannot build a bpb.
13069                                  ;           ; we do not assume any type of bpb for this medium.
13070                                  ;
13071                                  ;got_unknown_medium:
13072                                  ;           ;mov    ds:set_id_flag,       0
13073                                  ;           mov     byte [set_id_flag], 0
13074                                  ;           mov     al, 7
13075                                  ;           stc
13076                                  ;           jmp     short getret
13077
13078                                  ; =============== S U B   R O U T I N E =======================================
13079
13080                                  ; 15/10/2022 - Retro DOS 4.0 (Modified MSDOS 5.0 IO.SYS)
13081
13082                                  ; ---------------------------------------------------------------
13083                                  ; read in the boot sector. set carry if error in reading sector.
13084                                  ; bx is set to 1 if the boot sector is invalid, otherwise it is 0.
13085                                  ;
13086                                  ;   assumes es:di -> bds, ds-> Bios_Data
13087                                  ; ---------------------------------------------------------------
13088
13089                                  ; 10/03/2019 - Retro DOS v4.0
13090
13091                                  ; 30/12/2022 - Retro DOS v4.2
13092                                  ; (MSDOS 6.21 IO.SYS, BIOSCODE:06C3h)
13093                                  ; ((MSDOS 6.22 IO.SYS, BIOSCODE:06C3h)) ; 22/12/2023
13094
13095                                  ; 22/12/2023 - Retro DOS v5.0
13096                                  ; (PCDOS 7.1 IBMBIO.COM, BIOSCODE:07C6h)
13097
13098                                  readbootsec:
13099 000007B2 B600                              mov     dh, 0           ; head 0
13100 000007B4 B90100                            mov     cx, 1           ; cylinder 0, sector 1
13101 000007B7 E8FC00                            call    read_sector
13102 000007BA 7243                              jb      short err_ret
13103 000007BC 31DB                              xor     bx, bx          ; assume valid boot sector
13104
13105                                            ; put a sanity check for the boot sector in here to detect
13106                                            ; boot sectors that do not have valid bpbs. we examine the
13107                                            ; first two bytes - they must contain a long jump (69h) or a
13108                                            ; short jump (EBh) followed by a nop (90h), or a short jump
13109                                            ; (E9h). if this test is passed, we further check by examining
13110                                            ; the signature at the end of the boot sector for the word
13111                                            ; AA55h. if the signature is not present, we examine the media
13112                                            ; descriptor byte to see if it is valid. for dos 3.3, this
13113                                            ; logic is modified a little bit. we are not going to check
13114                                            ; signature. instead we are going to sanity check the media
13115                                            ; byte in bpb regardless of the validity of signature. this is
13116                                            ; to save the already developed commercial products that have
13117                                            ; good jump instruction and signature but with the false bpb
13118                                            ; informations
13119
13120                                  ; that will crash the diskette drive operation. (for example, symphony diskette).
13121
13122                                            ; 02/09/2023
13123                                            ; 19/10/2022
13124                                            ;cmp    byte [disksector], 69h ; is it a direct jump?
13125                                            ;jz     short check_bpb_mediabyte ; don't need to find a nop
13126                                            ;cmp    byte [disksector], 0E9h ; dos 2.0 jump?
13127                                            ;jz     short check_bpb_mediabyte ; no need for     nop
13128                                            ;cmp    byte [disksector], 0EBh ; how about a short jump?
13129                                            ;jnz    short invalidbootsec
13130                                            ;cmp    byte [disksector+2], 90h ; is next one a nop?
13131                                            ;jnz    short invalidbootsec
13132
13133                                            ; 02/09/2023 (PCDOS 7.1)
13134 000007BE A0[5201]                        mov     al, [disksector]
13135 000007C1 3C69                            cmp     al, 69h         ; is it a direct jump?
13136 000007C3 740F                            je      short check_bpb_mediabyte
13137                                                                  ; don't need to find a nop
13138 000007C5 3CE9                            cmp     al, 0E9h        ; dos 2.0 jump?
13139 000007C7 740B                            je      short check_bpb_mediabyte
13140                                                                  ; no need for nop
13141 000007C9 3CEB                            cmp     al, 0EBh        ; how about a short jump?
13142 000007CB 7530                            jne     short invalidbootsec
13143 000007CD 803E[5401]90                    cmp     byte [disksector+2], 90h ; is next one a nop?
13144 000007D2 7529                            jne     short invalidbootsec
13145
13146                                  ; 15/10/5022 - Retro DOS v4.0 (Modified MSDOS 5.0 IO.SYS)
13147                                  ;
13148                                  ;; 10/03/2019
13149                                  ;; (MSDOS 3.3, MSDISK.ASM, 1988)
13150                                  ;;
13151                                  ;; Don't have to perform the following signature check since
```

```
                                ;; we need to check the media byte even with the good signatured diskette.
                                ;;
                                ;;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
```

```
13276 0000080E 8B4CD3                          mov     cx, [si-45]     ; si = disksector+64 -> 64-45 = 19
13277                                                                  ; disksektor+19 = BPB_TotSec16
13278 00000811 31C0                            xor     ax, ax
13279 00000813 E308                            jcxz    movbpb_bigdisk
13280 00000815 26894DE0                        mov     [es:di-32], cx ; write 16 bit total sectors
13281                                                                  ; to 32 bit total sectors field
13282 00000819 268945E2                        mov     [es:di-30], ax ; BPB_TotalSec32+2 (BDS offset 29, BPB offset 23)
13283                                  movbpb_bigdisk:
13284 0000081D 3944D6                          cmp     [si-42], ax     ; BPB_FATSz16 = disksector+22
13285 00000820 7410                            jz      short movbpb_fat32
13286                                  movbpb_fat:
13287 00000822 83EF1C                          sub     di, 28          ; di = BDS offset 31 (BPB offset 25)
13288                                          ; 29/12/2023
13289 00000825 B90C00                          mov     cx, 12          ; clear 12 byte extended BDS (FAT32) fields
13290                                                                  ; (which are used only for FAT32 disks)
13291 00000828 F3AA                            rep stosb
13292 0000082A 48                              dec     ax              ; -1 ; 0FFFFh
13293 0000082B AB                              stosw                   ; set BDS offset 43 (dword) to -1
13294                                                                  ; dword [BDS.BPB_FSInfo] = 0FFFFFFFFh
13295 0000082C AB                              stosw
13296 0000082D 40                              inc     ax              ; ax = 0
13297 0000082E B10C                            mov     cl, 12
13298                                          ;mov    cx, 12          ; clear BDS offset 47 to 59
13299                                                                  ; (BPB offset 41 to 53) (disksector offset 52 to 64)
13300 00000830 F3AA                            rep stosb
13301                                  movbpb_fat32:
13302 00000832 5F                              pop     di
13303                                  %endif
13304                                          ;;;
13305
13306 00000833 803E[9B04]01                    cmp     byte [set_id_flag], 1 ; called by get_bpb?
13307 00000838 75C4                            jnz     short movbpb_ret
13308 0000083A E81200                          call    mov_media_ids
13309 0000083D 7205                            jb      short movbpb_conv ; conventional boot record?
13310 0000083F C606[9B04]02                    mov     byte [set_id_flag], 2 ; signals that volume id is set
13311                                  movbpb_conv:
13312 00000844 803E[7700]01                    cmp     byte [fhave96], 1
13313 00000849 75B3                            jnz     short movbpb_ret
13314 0000084B E83714                          call    resetchanged    ; reset flags in bds to       not fchanged.
13315                                          ; 10/12/2022
13316                                          ; cf = 0
13317                                  ;movbpb_ret:
13318                                          ;clc
13319 0000084E C3                              retn
13320
13321                                  ; ============== S U B   R O U T I N E =======================================
13322
13323                                  ;copy the boot_serial number, volume id, and filesystem id from the
13324                                  ;***extended boot record*** in ds:disksector to the bds table pointed
13325                                  ;by es:di.
13326
13327                                  ;in.) es:di -> bds
13328                                  ;     ds:disksector = valid extended boot record.
13329                                  ;out.) vol_serial, bds_volid and bds_system_id in bds are set according to
13330                                  ;      the boot record information.
13331                                  ;     carry flag set if not an extended bpb.
13332                                  ;     all registers saved except the flag.
13333
13334                                  ; 22/12/2023
13335                                  %if 0
13336                                          ; 19/10/2022
13337                                  mov_media_ids:
13338                                          cmp     byte [disksector+26h], 29h
13339                                                                  ; [disksector+EXT_BOOT.SIG],
13340                                                                  ; EXT_BOOT_SIGNATURE
13341                                          jnz     short mmi_not_ext
13342                                          push    cx
13343                                          mov     cx, [disksector+27h]
13344                                                                  ; [disksector+EXT_BOOT.SERIAL]
13345                                          mov     [es:di+57h], cx         ; [es:di+BDS.vol_serial]
13346                                          mov     cx, [disksector+29h]
13347                                                                  ; [disksector+EXT_BOOT.SERIAL+2]
13348                                          mov     [es:di+59h], cx         ; [es:di+BDS.vol_serial+2]
13349                                          push    di
13350                                          push    si
13351                                          mov     cx, 11          ; size_of_EXT_BOOT_VOL_LABEL
13352                                          mov     si, disksector+2Bh
13353                                          ;mov    si, (offset disksector+2Bh) ;
13354                                                                  ; disksector+EXT_BOOT.VOL_LABEL
13355                                          add     di, 75          ; BDS.volid
13356                                          rep movsb
13357                                          ;mov    cx, 8           ; size_of_EXT_SYSTEM_ID
13358                                          ; 10/12/2022
13359                                          mov     cl, 8 ; cx = 8
13360                                          mov     si, disksector+36h
13361                                          ;mov    si, (offset disksector+36h) ; disksector+EXT_BOOT.SYSTEM_ID
13362                                          add     di, 5           ; (BDS.filesys_id-BDS.volid)-size_of_EXT_BOOT_VOL_LABEL
13363                                          rep movsb
13364                                          pop     si
13365                                          pop     di
13366                                          pop     cx
13367                                          ; 10/12/2022
13368                                          ; cf = 0
13369                                          ;clc            ; this clc is not required (16/06/2019 - Erdogan Tan)
13370                                                          ; (20/09/2022)
13371                                          retn
13372                                  %else
13373                                          ; 22/12/2023 - Retro DOS v5.0
13374                                          ; (PCDOS 7.1 IBMBIO.COM, BIOSCODE:0865h)
13375                                          ;;;
13376                                  mov_media_ids:
13377 0000084F 833E[6801]00                    cmp     word [disksector+16h], 0 ; BPB.FATSz16
13378 00000854 7507                            jnz     short mmi_chk_fat
13379 00000856 803E[9401]29                    cmp     byte [disksector+42h], 29h
13380                                                                  ; [disksector+FAT32_EXT_BOOT.SIG],
13381                                                                  ; EXT_BOOT_SIGNATURE
13382 0000085B EB05                            jmp     short mmi_chk_fat32
13383                                  mmi_chk_fat:
13384 0000085D 803E[7801]29                    cmp     byte [disksector+26h], 29h
13385                                                                  ; [disksector+EXT_BOOT.SIG],EXT_BOOT_SIGNATURE
13386                                  mmi_chk_fat32:
13387 00000862 7543                            jnz     short mmi_not_ext
13388 00000864 51                              push    cx
13389 00000865 50                              push    ax
13390 00000866 57                              push    di
13391 00000867 56                              push    si
13392 00000868 1E                              push    ds
13393 00000869 833E[6801]00                    cmp     word [disksector+16h], 0 ; BPB.FATSz16
13394 0000086E 750C                            jnz     short mmi_fat
13395
13396                                  mmi_fat32:                      ; FAT32 file system
13397                                          ;lds    cx, dword ptr ds:disksector+43h
13398 00000870 C50E[9501]                      lds     cx, [disksector+43h]  ; BS_FAT32_VolID
13399 00000874 BE[9901]                        mov     si, disksector+47h    ; BS_FAT32_VolLab
```

```
13400 00000877 B8[A401]                         mov     ax, disksector+52h    ; BS_FAT32_FilSysType
13401 0000087A EB0A                             jmp     short mmi_do
13402
13403                                   mmi_fat:
13404                                           ;lds    cx, dword ptr ds:disksector+27h
13405 0000087C C50E[7901]                       lds     cx, [disksector+27h]  ; BS_VolID
13406 00000880 BE[7D01]                         mov     si, disksector+2Bh    ; BS_VolLab
13407 00000883 B8[8801]                         mov     ax, disksector+36h    ; BS_FilSysType
13408                                   mmi_do:
13409 00000886 26898D8900                       mov     [es:di+89h], cx       ; [es:di+BDS.vol_serial]
13410                                                                         ; (BDS offset 137)
13411 0000088B 268C9D8B00                       mov     [es:di+8Bh], ds       ; [es:di+BDS.vol_serial+2]
13412 00000890 1F                               pop     ds
13413 00000891 B90B00                           mov     cx, 11
13414 00000894 83C77D                           add     di, 125       ; di = di+125 = BDS.volid
13415 00000897 F3A4                             rep movsb
13416 00000899 B108                             mov     cl, 8         ; di = di+136
13417 0000089B 89C6                             mov     si, ax        ; BS_FilSysType or BS_FAT32_FilSysType
13418 0000089D 83C705                           add     di, 5         ; di = di+141 = BDS.filesys_id
13419 000008A0 F3A4                             rep movsb
13420 000008A2 5E                               pop     si
13421 000008A3 5F                               pop     di
13422 000008A4 58                               pop     ax
13423 000008A5 59                               pop     cx
13424                                           ;clc    ; this clc is not required (16/06/2019 - Erdogan Tan)
13425                                                   ; (20/09/2022 - 27/06/2023) MSDOS 6.21 .. PCDOS 7.1
13426 000008A6 C3                               retn
13427                                   %endif
13428                                           ;;;
13429
13430                                   ; ---------------------------------------------------------------------------
13431
13432                                   mmi_not_ext:
13433 000008A7 F9                               stc
13434 000008A8 C3                               retn
13435
13436                                   ; =============== S U B   R O U TI N E =======================================
13437
13438                                   ; 15/10/2022
13439                                   ; ------------------------------------------------------------
13440                                   ; read in the fat sector and get the media byte from it.
13441                                   ; input : es:di -> bds
13442                                   ; output:
13443                                   ;     carry set if an error occurs, ax contains error code.
13444                                   ;     otherwise, ah contains media byte on exit
13445                                   ; ------------------------------------------------------------
13446
13447                                   readfat:
13448                                           ;mov    dh, 0
13449                                           ; 10/12/2022
13450 000008A9 30F6                             xor     dh, dh
13451 000008AB B90200                           mov     cx, 2         ; head 0
13452                                                                 ; cylinder 0, sector 2
13453 000008AE E80500                           call    read_sector
13454 000008B1 7202                             jb      short bad_fat_ret
13455 000008B3 8A27                             mov     ah, [bx]      ; media byte
13456                                   bad_fat_ret:
13457 000008B5 C3                               retn
13458
13459                                   ; =============== S U B   R O U TI N E =======================================
13460
13461                                   ; 15/10/2022
13462
13463                                   ; ---------------------------------------------------------------------------
13464                                   ; read a single sector into the temp buffer.
13465                                   ; perform three retries in case of error.
13466                                   ;   inputs: es:[di].bds_drivenum has physical drive to use
13467                                   ;           cx has sector and cylinder
13468                                   ;           dh has head
13469                                   ;           es:di has bds
13470                                   ;           ds has Bios_Data
13471                                   ;
13472                                   ;   outputs:       carry clear
13473                                   ;                Bios_Data:bx point to sector
13474                                   ;                   (note: some callers assume location of buffer)
13475                                   ;
13476                                   ;           carry set
13477                                   ;               ax has rom error code
13478                                   ;
13479                                   ; register bp is preserved.
13480                                   ; ---------------------------------------------------------------------------
13481
13482                                   ; 10/03/2019 - Retro DOS v4.0
13483                                   ; 22/12/2023 - Retro DOS v5.0
13484
13485                                           ; 19/10/2022
13486                                   read_sector:
13487 000008B6 55                               push    bp
13488 000008B7 BD0300                           mov     bp, 3         ; make 3 attempts
13489 000008BA 268A5504                         mov     dl, [es:di+4]  ; [es:di+BDS.drivenum]
13490 000008BE BB[5201]                         mov     bx, disksector ; get es:bx to point to      buffer
13491                                   rd_ret:
13492 000008C1 06                               push    es
13493 000008C2 1E                               push    ds
13494 000008C3 07                               pop     es
13495 000008C4 B80102                           mov     ax, 201h
13496 000008C7 CD13                             int     13h           ; DISK - READ SECTORS INTO MEMORY
13497                                                                 ; AL = number of sectors to read, CH = track, CL = sector
13498                                                                 ; DH = head, DL       = drive, ES:BX -> buffer to fill
13499                                                                 ; Return: CF set on error, AH =      status, AL = number of sectors read
13500 000008C9 07                               pop     es
13501 000008CA 734A                             jnb     short okret2
13502                                   rd_rty:
13503 000008CC E81205                           call    again         ; reset disk, decrement       bp, preserve ax
13504 000008CF 7442                             jz      short err_rd_ret
13505
13506                                           ; 22/12/2023 - Retro DOS v5.0 (PCDOS 7.1 IBMBIO.COM)
13507 000008D1 26F6453F01                       test    byte [es:di+3Fh], 1
13508                                           ;test   byte [es:di+23h], 1
13509                                           ;;test  byte ptr [es:di+23h], 1      ; [es:di+BDS.flags]
13510                                                                 ; fnon_removable
13511 000008D6 75E9                             jnz     short rd_ret
13512 000008D8 803E[A905]00                     cmp     byte [media_set_for_format], 0
13513 000008DD 7510                             jnz     short rd_skip1_dpt
13514 000008DF 50                               push    ax
13515 000008E0 1E                               push    ds            ; for retry, set the head settle time to 0Fh
13516 000008E1 C536[2D01]                       lds     si, [dpt]
13517                                           ;mov    al, [si+9]    ; [si+DISK_PARMS.DISK_HEAD_STTL]
13518                                           ;mov    byte [si+9], 15 ; [si+DISK_PARMS.DISK_HEAD_STTL]
13519                                           ;                     ; NORMSETTLE
13520                                           ; 12/12/2022
13521 000008E5 B00F                             mov     al, 15
13522 000008E7 864409                           xchg    al, [si+9]
13523                                           ;
```

```
13524 000008EA 1F                               pop     ds
13525 000008EB A2[2A01]                         mov     [save_head_sttl], al
13526 000008EE 58                               pop     ax
13527                                   rd_skip1_dpt:
13528 000008EF 06                               push    es
13529 000008F0 1E                               push    ds
13530 000008F1 07                               pop     es
13531 000008F2 B80102                           mov     ax, 201h
13532 000008F5 CD13                             int     13h             ; DISK - READ SECTORS INTO MEMORY
13533                                                                   ; AL = number of sectors to read, CH = track, CL = sector
13534                                                                   ; DH = head, DL       = drive, ES:BX -> buffer to fill
13535                                                                   ; Return: CF set on error, AH =      status, AL = number of sectors read
13536 000008F7 07                               pop     es
13537 000008F8 9C                               pushf
13538 000008F9 803E[A905]00                     cmp     byte [media_set_for_format], 0
13539 000008FE 750E                             jnz     short rd_skip2_dpt
13540 00000900 50                               push    ax
13541 00000901 A0[2A01]                         mov     al, [save_head_sttl]
13542 00000904 1E                               push    ds
13543 00000905 C536[2D01]                       lds     si, [dpt]
13544 00000909 884409                           mov     [si+9], al      ; [si+DISK_PARMS.DISK_HEAD_STTL]
13545 0000090C 1F                               pop     ds
13546 0000090D 58                               pop     ax
13547                                   rd_skip2_dpt:
13548 0000090E 9D                               popf
13549 0000090F 7305                             jnb     short okret2
13550 00000911 EBB9                             jmp     short rd_rty
13551                                   ; ---------------------------------------------------------------------------
13552
13553                                   err_rd_ret:
13554 00000913 B2FF                             mov     dl, 0FFh        ; make sure we ask rom if media      has changed
13555                                                                   ; return error
13556 00000915 F9                               stc
13557
13558                                   ; update information pertaining to last drive accessed, time of access, last
13559                                   ; track accessed in that drive.
13560
13561                                   okret2:
13562 00000916 8816[7600]                       mov     [step_drv], dl ; set up for head settle logic in disk
13563 0000091A 8816[1E01]                       mov     [tim_drv], dl  ; save drive last accessed
13564
13565                                                                   ; 22/12/2023 - Retro DOS v5.0 (PCDOS 7.1 IBMBIO.COM)
13566 0000091E 26886D78                         mov     [es:di+78h], ch
13567                                           ;mov    [es:di+46h], ch       ; [es:di+BDS.track]
13568                                                                   ; save last track accessed on this drive
13569                                                                   ; preserve flags in case error occurred
13570 00000922 9C                               pushf
13571 00000923 E89B04                           call    set_tim
13572 00000926 9D                               popf                    ; restore flags
13573 00000927 5D                               pop     bp
13574 00000928 C3                               retn
13575
13576                                   ;---------------------------------------------------------------------------
13577                                   ;   disk open/close routines
13578                                   ;---------------------------------------------------------------------------
13579
13580                                   dsk_open:                       ; 2C7h:80Ah = 70h:2D7Ah
13581 00000929 803E[7700]00                     cmp     byte [fhave96], 0
13582 0000092E 7407                             jz      short dsk_open_exit ; done if no changeline support
13583 00000930 E871FC                           call    SetDrive        ; get bds for drive
13584                                                                   ; 22/12/2023 - Retro DOS v5.0 (PCDOS 7.1 IBMBIO.COM)
13585 00000933 26FF453C                         inc     word [es:di+3Ch] ; [es:di+BDS.opcnt] ; BDS offset 60
13586                                           ;inc    word [es:di+20h] ; [es:di+BDS.opcnt]
13587                                   dsk_open_exit:
13588                                                                   ; 10/12/2022
13589                                                                   ; cf = 0
13590                                                                   ;clc            ; CF is already ZERO here (18/09/2022, MSDOS 5.0 IO.SYS)
13591                                                                   ; (19/07/2019 -       Erdogan Tan - MSDOS 6.0       IO.SYS - retrodos4.s)
13592 00000937 C3                               retn
13593                                   ; ---------------------------------------------------------------------------
13594
13595                                   dsk_close:                      ; 2C7h:81Ah = 70h:2D8Ah
13596 00000938 803E[7700]00                     cmp     byte [fhave96], 0
13597 0000093D 740E                             jz      short exitjx    ; done if no changeline      support
13598 0000093F E862FC                           call    SetDrive        ; get bds for drive
13599                                                                   ; 22/12/2023 - Retro DOS v5.0 (PCDOS 7.1 IBMBIO.COM)
13600 00000942 26837D3C00                       cmp     word [es:di+3Ch], 0 ; [es:di+BDS.opcnt] ; BDS off 60
13601                                           ;cmp    word [es:di+20h], 0 ; [es:di+BDS.opcnt]
13602 00000947 7404                             jz      short exitjx    ; watch out for wrap
13603                                                                   ; 22/12/2023
13604 00000949 26FF4D3C                         dec     word [es:di+3Ch]
13605                                           ;dec    word [es:di+20h]
13606                                   exitjx:
13607                                                                   ; 10/12/2022
13608                                                                   ; cf = 0
13609                                                                   ;clc            ; CF is already ZERO here (18/09/2022, MSDOS 5.0 IO.SYS)
13610                                                                   ; (19/07/2019 -       Erdogan Tan - MSDOS 6.0       IO.SYS - retrodos4.s)
13611 0000094D C3                               retn
13612
13613                                   ;---------------------------------------------------------------------------
13614                                   ;           disk removable routine
13615                                   ;---------------------------------------------------------------------------
13616
13617                                                   ; al is unit #
13618                                   dsk_rem:                        ; 2C7h:831h = 70h:2DA1h
13619 0000094E E853FC                           call    SetDrive        ; get bds for this drive
13620                                                                   ; 22/12/2023 - Retro DOS v5.0 (PCDOS 7.1 IBMBIO.COM)
13621                                                                   ;test   byte [es:di+BDS.flags], fnon_removable
13622 00000951 26F6453F01                       test    byte [es:di+3Fh], 1 ; [es:di+BDS.flags], fnon_removable
13623 00000956 74F5                             jz      short exitjx
13624                                                                   ;test   byte [es:di+23h], 1 ; [es:di+BDS.flags], fnon_removable
13625                                                                   ;;jnz   short x_bus_exit ; non_rem
13626                                                                   ;jnz    short non_rem  ; 15/10/2022
13627                                                                   ;; 10/12/2022
13628                                                                   ;; cf = 0
13629                                                                   ;;clc                  ; CF is already ZERO here
13630                                                                   ;                      ; 15/10/2022
13631                                                                   ;retn
13632                                   ; ---------------------------------------------------------------------------
13633
13634                                   non_rem:
13635                                   x_bus_exit:
13636 00000958 B403                             mov     ah, 3           ; 2C7h:83Dh = 0070h:2DADh
13637                                                                   ; return busy status
13638 0000095A F9                               stc
13639                                   dsk_ret:
13640 0000095B C3                               retn
13641
13642                                   ;---------------------------------------------------------------------------
13643                                   ;           disk i/o routines
13644                                   ;---------------------------------------------------------------------------
13645
13646                                   dsk_writv:                      ; 2C7h:841h = 70h:2DB1h
13647                                           ;mov    word [wrtverify], 103h
```

```
                                        ; 19/10/2022
0000095C C706[2001]0301                 mov     word [rflag], 103h
                                        ;mov    word ptr ds:rflag, 103h       ; write and verify
00000962 EB06                           jmp     short dsk_cl
                                ; ---------------------------------------------------------------------------

                                dsk_writ:                               ; 2C7h:849h = 70h:2DB9h
                                        ;mov    word [wrtverify], 3
                                        ; 19/10/2022
00000964 C706[2001]0300                 mov     word [rflag], 3
                                        ;mov    word ptr ds:rflag, 3 ; romwrite
                                dsk_cl:
0000096A E8A400                         call    diskio          ; romwrite
                                ; ---------------------------------------------------------------------------

                                dsk_io:
0000096D 73EC                           jnb     short dsk_ret
0000096F E965F7                         jmp     bc_err_cnt
                                ; ---------------------------------------------------------------------------

                                dsk_read:                               ; ; 2C7h:857h =       70h:2DC7h
00000972 E89700                         call    diskrd
00000975 EBF6                           jmp     short dsk_io

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

                                ; 15/10/2022
                                ; 10/03/2019 - Retro DOS v4.0
                                ; 22/12/2023 - Retro DOS v5.0

                                ;--------------------------------------------------------
                                ; miscellaneous odd jump routines.
                                ; moved out of mainline for speed.

                                ; if we have a system where we have virtual drives, we need
                                ; to prompt the user to place the correct disk in the drive.
                                ;
                                ;   assume es:di -> bds, ds:->Bios_Data
                                ;--------------------------------------------------------

                                        ; 19/10/2022
                                checksingle:
00000977 50                             push    ax
00000978 53                             push    bx
                                        ; 22/12/2023 - Retro DOS v5.0 (PCDOS 7.1 IBMBIO.COM)
00000979 268B5D3F                       mov     bx, [es:di+3Fh]       ; [es:di+BDS.flags]
                                        ;mov    bx, [es:di+23h]       ; [es:di+BDS.flags]

                                ; if hard drive, cannot change disk.
                                ; if current owner of physical drive, no need to change diskette.

0000097D F6C321                         test    bl, 21h         ; fnon_removable|fi_own_physical
00000980 7573                           jnz     short singleret
00000982 F6C310                         test    bl, 10h         ; fi_am_mult
                                                                ; is there a drive sharing this      physical drive?
00000985 746E                           jz      short singleret

                                ; look for the previous owner of this physical drive
                                ; and reset its ownership flag.

00000987 268A4504                       mov     al, [es:di+4]  ; [es:di+BDS.drivenum]
                                                                ; get physical drive number
0000098B 06                             push    es              ; preserve pointer to current bds
0000098C 57                             push    di
0000098D C43E[1901]                     les     di, [start_bds] ; get first bds
                                scan_list:
00000991 26384504                       cmp     [es:di+4], al
00000995 7553                           jnz     short scan_skip       ; Not our drive. Try next bds.
00000997 B320                           mov     bl, 20h ; ' '   ; fi_own_physical ; test ownership flag
                                        ; 22/12/2023 - Retro DOS v5.0 (PCDOS 7.1 IBMBIO.COM)
00000999 26845D3F                       test    [es:di+3Fh], bl       ; [es:di+BDS.flags]
                                        ;test   [es:di+23h], bl
0000099D 744B                           jz      short scan_skip       ; he doesn't own it either. continue
0000099F 26305D3F                       xor     [es:di+3Fh], bl
                                        ;xor    [es:di+23h], bl       ; reset ownership flag
000009A3 5F                             pop     di              ; restore pointer to current bds
000009A4 07                             pop     es
000009A5 26085D3F                       or      [es:di+3Fh], bl
                                        ;or     [es:di+23h], bl       ; ; set ownership flag

                                ; we examine the fsetowner flag. if it is set, then we are using the code in
                                ; checksingle to just set the owner of a drive. we must not issue the prompt
                                ; in this case.
000009A9 803E[7A00]01                   cmp     byte [fsetowner], 1
000009AE 7517                           jnz     short not_fsetowner
                                        ;cmp    byte ptr es:[di+4], 0 ;       are we handling       drive number 0 ?
000009B0 26807D0400                     cmp     byte [es:di+4], 0
000009B5 753E                           jnz     short singleret
000009B7 268A4505                       mov     al, [es:di+5]
                                        ;mov    al, es:[di+5]  ; [es:di+BDS.drivelet]
                                                                ; get the DOS drive letter
000009BB 06                             push    es
000009BC 8E06[1A00]                     mov     es, [zeroseg]
000009C0 26A20405                       mov     [es:LSTDRV], al
                                        ;mov    es:504h, al    ; [es:LSTDRV]
                                                                ; set up sdsb
000009C4 07                             pop     es              ; restore bds pointer
000009C5 EB2E                           jmp     short singleret
                                ; ---------------------------------------------------------------------------

                                ; to support "backward" compatibility with ibm's "single drive status byte"
                                ; we now check to see if we are in a single drive system and the application
                                ; has "cleverly" diddled the sdsb

                                not_fsetowner:
000009C7 803E[7800]02                   cmp     byte [single], 2 ; if (single_drive_system)
000009CC 7517                           jnz     short ignore_sdsb
000009CE 50                             push    ax
000009CF 268A4505                       mov     al, [es:di+5]  ; if (curr_drv == req_drv)
000009D3 88C4                           mov     ah, al
000009D5 06                             push    es
000009D6 8E06[1A00]                     mov     es, [zeroseg]
000009DA 2686060405                     xchg    al, [es:LSTDRV]
                                        ;xchg   al, es:504h    ; [es:LSTDRV]
                                                                ; then swap(curr_drv,req_drv)
000009DF 07                             pop     es
000009E0 38C4                           cmp     ah, al          ; else
000009E2 58                             pop     ax              ; swap(curr_drv,req_drv)
000009E3 7410                           jz      short singleret       ; issue swap_dsk_msg
                                ignore_sdsb:
000009E5 E8B310                         call    swpdsk
000009E8 EB0B                           jmp     short singleret
                                ; ---------------------------------------------------------------------------

```

```
13772                                   scan_skip:
13773 000009EA 26C43D                             les     di, [es:di]
13774                                             ;les    di, es:[di]     ; [es:di+BDS.link]
13775                                                                     ; go to next bds
13776 000009ED 83FFFF                             cmp     di, 0FFFFh ; -1         ; end of list?
13777 000009F0 759F                               jnz     short scan_list         ; continue until hit end of list
13778 000009F2 F9                                 stc
13779 000009F3 5F                                 pop     di              ; restore current bds
13780 000009F4 07                                 pop     es
13781                                   singleret:
13782 000009F5 5B                                 pop     bx
13783 000009F6 58                                 pop     ax
13784 000009F7 C3                                 retn
13785
13786                                   ; 22/12/2023
13787                                   %if 0
13788                                   ; ---------------------------------------------------------------------------
13789
13790                                   baddrive:
13791                                             mov     al, 8           ; sector not found
13792                                             jmp     short baddrive_ret
13793                                   %endif
13794
13795                                   ; ---------------------------------------------------------------------------
13796
13797                                   unformatteddrive:
13798 000009F8 B007                               mov     al, 7           ; unknown media
13799                                   ;baddrive_ret:
13800 000009FA F9                                 stc
13801                                   ; ---------------------------------------------------------------------------
13802
13803                                   ioret:
13804 000009FB C3                                 retn
13805
13806                                   ; ---------------------------------------------------------------------------
13807
13808                                             ; 22/12/2023 - Retro DOS v5.0
13809                                             ; PCDOS 7.1 IBMBIO.COM - BIOSCODE:0A1Bh
13810
13811 000009FC 10                      LBA_Packet: db 16                 ; ...
13812                                                                     ; DAP buffer
13813 000009FD 00                                   db 0
13814 000009FE 0000                    dap_block_cnt:     dw 0                  ; ...
13815 00000A00 00000000                dap_trans_buf:     dd 0                  ; ...
13816 00000A04 00000000                dap_lba_value:     dd 0                  ; ...
13817 00000A08 00000000                           dd 0
13818
13819                                   ; ---------------------------------------------------------------------------
13820
13821                                   ; 15/10/2022
13822
13823                                   ; ---------------------------------------------------------------------------
13824                                   ;   disk i/o handler
13825                                   ;
13826                                   ;   al = drive number (0-6)
13827                                   ;   ah = media descriptor
13828                                   ;   cx = sector count
13829                                   ;   dx = first sector (low)
13830                                   ;   [start_sec_h] = first sector (high)  32 bit calculation.
13831                                   ;   ds = cs
13832                                   ;   es:di = transfer address
13833                                   ;   [rflag]=operation (2=read, 3=write)
13834                                   ;   [verify]=1 for verify after write
13835                                   ;
13836                                   ;   if successful carry flag = 0
13837                                   ;     else cf=1 and al contains error code
13838                                   ; ---------------------------------------------------------------------------
13839
13840                                             ; 12/12/2023
13841                                             ; ds = biosdata segment (cs = bioscode segment)
13842                                   diskrd:
13843                                             ;mov    ds:rflag, 2     ; romread
13844                                             ; 19/10/2022
13845 00000A0C C606[2001]02                       mov     byte [rflag], 2 ; romread
13846
13847                                   ; =============== S U B   R O U T I N E =======================================
13848
13849                                             ; 22/12/2023 - Retro DOS v5.0
13850                                             ; PCDOS 7.1 IBMBIO.COM - BIOSCODE:0A30h
13851                                   ; 22/12/2023
13852                                   %if 0
13853                                             ; 19/10/2022
13854                                   diskio:
13855                                             mov     bx, di          ; es:bx = transfer address
13856                                             mov     [xfer_seg], es ; save transfer       segment
13857                                             call    SetDrive
13858                                             mov     al, [es:di+10h]         ; [es:di+BDS.media]
13859                                             mov     [medbyt], al
13860                                             ;jcxz   short ioret
13861                                             jcxz    ioret
13862
13863                                   ; see if the media is formatted or not by checking the flags field in
13864                                   ; in the bds. if it is unformatted we cannot allow i/o, so we should
13865                                   ; go to the error exit at label unformatteddrive.
13866
13867                                             test    byte [es:di+24h], 2
13868                                             ;test   byte ptr es:[di+24h], 2       ; [es:di+BDS.flags+1]
13869                                                                     ; unformatted_media
13870                                             jnz     short unformatteddrive
13871                                             mov     [seccnt], cx    ; save sector count
13872                                             mov     [spsav], sp     ; save sp
13873
13874                                   ; ensure that we are trying to access valid sectors on the drive
13875
13876                                             mov     ax, dx
13877                                             xor     si, si ; 0
13878                                             add     dx, cx
13879                                             ;adc    si, 0
13880                                             ; 02/09/2023 (PCDOS 7.1)
13881                                             rcl     si, 1
13882                                             cmp     word [es:di+0Eh], 0 ; [es:di+BDS.totalsecs16]
13883                                                                     ; 32 bit sector ?
13884                                             jz      short sanity32
13885                                             ;cmp    si, 0
13886                                             ; 02/09/2023
13887                                             or      si, si
13888                                             jnz     short baddrive
13889                                             cmp     dx, [es:di+0Eh]         ; [es:di+BDS.totalsecs16]
13890                                             ja      short baddrive
13891                                             jmp     short sanityok
13892                                   ; ---------------------------------------------------------------------------
13893
13894                                   sanity32:
13895                                             add     si, [start_sec_h]
```

```
13896                                           cmp     si, [es:di+1Dh]       ; [es:di+BDS.totalsecs32+2]
13897                                           jb      short sanityok
13898                                           ja      short baddrive
13899                                           cmp     dx, [es:di+1Bh]       ; [es:di+BDS.totalsecs32]
13900                                           ja      short baddrive
13901                                   sanityok:
13902                                           mov     dx, [start_sec_h]
13903                                           add     ax, [es:di+17h]       ; [es:di+BDS.hiddensecs]
13904                                           adc     dx, [es:di+19h]       ; [es:di+BDS.hiddensecs+2]
13905
13906                                   ; now dx;ax have the physical first sector.
13907                                   ; since the following procedures is going to destroy ax, let's
13908                                   ; save it temporarily to saved_word.
13909
13910                                           mov     [saved_word], ax ; save the sector number (low)
13911
13912                                   ; set up pointer to disk base table in [dpt]. we cannot assume that iosetup
13913                                   ; will do it because we will skip the set up stuff with hard disks.
13914
13915                                           push    es
13916                                           ;mov    es, [zeroseg]
13917                                           ; 02/09/2023
13918                                           xor     si, si ; 0
13919                                           mov     es, si
13920                                           les     si, [es:DSKADR]
13921                                           ;les    si, es:78h     ; [es:DSKADR]
13922                                                                  ; current disk parm table
13923                                           mov     [dpt], si
13924                                           mov     [dpt+2], es
13925                                           pop     es
13926                                           test    byte [es:di+23h], 1 ; [es:di+BDS.flags]
13927                                                                  ; fnon_removable
13928                                           jnz     short skip_setup
13929                                           call    checksingle
13930
13931                                   ; check to see if we have previously noted a change line. the routine
13932                                   ; returns if everything is ok. otherwise, it pops off the stack and returns
13933                                   ; the proper error code.
13934
13935                                           cmp     byte [fhave96], 0 ; do we have changeline support?
13936                                           jz      short diskio_nochangeline ; brif not
13937                                           call    checklatchio   ; will do a sneaky pop stack return
13938                                                                  ; if a disk error occurs
13939                                   diskio_nochangeline:
13940                                           call    iosetup        ; set up tables       and variables for i/o
13941
13942                                   ; now the settle values are correct for the following code
13943
13944                                   skip_setup:
13945
13946                                   ; 32 bit sector calculation.
13947                                   ; dx:[saved_word] = starting sector number.
13948
13949                                           mov     ax, dx
13950                                           xor     dx, dx
13951                                           ;div    word [es:di+13h] ; [es:di+BDS.secpertrack]
13952                                                                  ; divide by sec per track
13953                                           ; 02/09/2023
13954                                           mov     cx, [es:di+13h]
13955                                           div     cx
13956                                           mov     [temp_h], ax
13957                                           mov     ax, [saved_word]
13958                                           div     cx ; 02/09/2023
13959                                           ;div    word [es:di+13h] ; [es:di+BDS.secpertrack]
13960                                                                  ; now, [temp_h]:ax = track #, dx = sector
13961                                           ;inc    dl             ; sector number       is 1 based.
13962                                           ; 18/12/2022
13963                                           inc     dx
13964                                           mov     [cursec], dl   ; save current sector
13965                                           mov     cx, [es:di+15h]       ; es:di+BDS.heads]
13966                                                                  ; get number of       heads
13967                                           push    ax
13968                                           xor     dx, dx
13969                                           mov     ax, [temp_h]   ; divide tracks       by heads per cylinder
13970                                           div     cx
13971                                           mov     [temp_h], ax
13972                                           pop     ax
13973                                           div     cx             ; now, [temp_h]:ax = cylinder #, dx = head
13974                                           cmp     word [temp_h], 0
13975                                           ja      short baddrive_brdg
13976                                           cmp     ax, 1024       ; 2^10 currently maxium      for track #.
13977                                           ja      short baddrive_brdg
13978                                           mov     [curhd], dl    ; save current head
13979                                           mov     [curtrk], ax   ; save current track
13980
13981                                   ; we are now set up for the i/o. normally, we consider the dma boundary
13982                                   ; violations here. not true. we perform the operation as if everything is
13983                                   ; symmetric; let the int 13 handler worry about the dma violations.
13984
13985                                           mov     ax, [seccnt]
13986                                           call    block          ; (cas - call/ret)
13987                                           ;call   done
13988                                           ;retn
13989                                           ; 18/12/2022
13990                                           jmp     done
13991                                   %else
13992                                           ;;;     ; 22/12/2023
13993                                   diskio:
13994 00000A11 89FB                               mov     bx, di         ; al = drive number
13995                                                                    ; cx = sector count
13996                                                                    ; dx = first sector (low)
13997                                                                    ; [start_sec_h] = first sector (high)
13998                                                                    ;
13999                                                                    ; es:bx = transfer address
14000 00000A13 8C06[A804]                         mov     [xfer_seg], es ; save transfer segment
14001 00000A17 E88AFB                             call    SetDrive
14002 00000A1A 268A4510                           mov     al, [es:di+10h]       ; [es:di+BDS.media]
14003 00000A1E A2[1F01]                           mov     [medbyt], al
14004 00000A21 E3D8                               jcxz    ioret
14005
14006                                   ; see if the media is formatted or not by checking the flags field in
14007                                   ; in the bds. if it is unformatted we cannot allow i/o, so we should
14008                                   ; go to the error exit at label unformatteddrive.
14009
14010 00000A23 26F6454002                         test    byte [es:di+40h], 2 ; [es:di+BDS.flags+1]
14011                                                                    ; unformatted_media
14012 00000A28 75CE                               jnz     short unformatteddrive
14013 00000A2A 890E[2201]                         mov     [seccnt], cx   ; save sector count
14014 00000A2E 8926[3501]                         mov     [spsav], sp    ; save sp
14015
14016                                   ; ensure that we are trying to access valid sectors on the drive
14017
14018 00000A32 89D0                               mov     ax, dx
14019 00000A34 31F6                               xor     si, si ; 0
```

```
14020 00000A36 01CA                             add     dx, cx
14021 00000A38 D1D6                             rcl     si, 1
14022 00000A3A 26837D0E00                       cmp     word [es:di+0Eh], 0 ; [es:di+BDS.totalsecs16]
14023                                                                   ; > 32 bit sector ?
14024 00000A3F 740E                             jz      short sanity32
14025 00000A41 09F6                             or      si, si
14026 00000A43 7506                             jnz     short baddrive
14027 00000A45 263B550E                         cmp     dx, [es:di+0Eh]       ; [es:di+BDS.totalsecs16]
14028                                           ;ja     short baddrive
14029                                           ;jmp    short sanityok
14030                                           ; 22/12/2023
14031 00000A49 7616                             jna     short sanityok
14032                                   ; 29/12/2023
14033                                   ; 22/12/2023
14034                                   ;%if 1
14035                                   ; ---------------------------------------------------------------------------
14036
14037                                   baddrive:
14038 00000A4B B008                             mov     al, 8           ; sector not found
14039                                           ;jmp    short baddrive_ret
14040                                   ; ---------------------------------------------------------------------------
14041                                   ;unformatteddrive:
14042                                           ;mov    al, 7           ; unknown media
14043                                   baddrive_ret:
14044 00000A4D F9                               stc
14045                                   ;ioret:
14046 00000A4E C3                               retn
14047                                   ;%endif
14048
14049                                   ; ---------------------------------------------------------------------------
14050
14051                                   sanity32:
14052 00000A4F 0336[9C04]                       add     si, [start_sec_h]
14053 00000A53 263B751D                         cmp     si, [es:di+1Dh]       ; [es:di+BDS.totalsecs32+2]
14054 00000A57 7208                             jb      short sanityok
14055 00000A59 77F0                             ja      short baddrive
14056 00000A5B 263B551B                         cmp     dx, [es:di+1Bh]       ; [es:di+BDS.totalsecs32]
14057 00000A5F 77EA                             ja      short baddrive
14058                                   sanityok:
14059 00000A61 8B16[9C04]                       mov     dx, [start_sec_h]
14060 00000A65 26034517                         add     ax, [es:di+17h]       ; [es:di+BDS.hiddensecs]
14061 00000A69 26135519                         adc     dx, [es:di+19h]       ; [es:di+BDS.hiddensecs+2]
14062
14063                                   ; now dx;ax have the physical first sector.
14064                                   ; since the following procedures is going to destroy ax, let's
14065                                   ; save it temporarily to saved_word.
14066
14067 00000A6D A3[9E04]                          mov     [saved_word], ax ; save the sector number (low)
14068
14069                                   ; set up pointer to disk base table in [dpt]. we cannot assume that iosetup
14070                                   ; will do it because we will skip the set up stuff with hard disks.
14071
14072 00000A70 06                                push    es
14073 00000A71 31F6                             xor     si, si ; 0
14074 00000A73 8EC6                             mov     es, si
14075                                           ;les    si, dword ptr es:78h
14076 00000A75 26C4367800                       les     si, [es:78h]   ; INT 1Eh vector address
14077                                                                   ; [es:DSKADR] - current disk parm table
14078 00000A7A 8936[2D01]                       mov     [dpt], si
14079 00000A7E 8C06[2F01]                       mov     [dpt+2], es
14080 00000A82 07                               pop     es
14081 00000A83 26F6453F01                       test    byte [es:di+3Fh], 1 ; [es:di+BDS.flags], fnon_removable
14082 00000A88 7510                             jnz     short chk_13h_ext_flag
14083 00000A8A E8EAFE                           call    checksingle
14084
14085                                   ; check to see if we have previously noted a change line. the routine
14086                                   ; returns if everything is ok. otherwise, it pops off the stack and returns
14087                                   ; the proper error code.
14088
14089 00000A8D 803E[7700]00                     cmp     byte [fhave96], 0 ; do we have changeline support?
14090 00000A92 7403                             jz      short diskio_nochangeline ; brif not
14091 00000A94 E8D210                           call    checklatchio   ; will do a sneaky pop stack return
14092                                                                   ; if a disk error occurs
14093                                   diskio_nochangeline:
14094 00000A97 E8E000                           call    iosetup        ; set up tables and variables for i/o
14095
14096                                   chk_13h_ext_flag:
14097 00000A9A 26F6454004                       test    byte [es:di+40h], 4 ; [es:di+BDS.flags+1], fLBArw
14098                                                                   ; LBA read/write flag
14099 00000A9F 7539                             jnz     short set_lbarw_1
14100                                           ;jmp    skip_setup
14101                                           ; 22/12/2023
14102                                   ; ---------------------------------------------------------------------------
14103
14104                                   ; now the settle values are correct for the following code
14105
14106                                   skip_setup:
14107
14108                                   ; 32 bit sector calculation.
14109                                   ; dx:[saved_word] = starting sector number.
14110
14111                                           ;push   bp ; ! (not necessary) ; 22/12/2023
14112 00000AA1 92                               xchg    ax, dx ; mov ax,dx
14113 00000AA2 31D2                             xor     dx, dx
14114 00000AA4 268B4D13                         mov     cx, [es:di+13h]       ; [es:di+BDS.secpertrack]
14115                                                                   ; divide by sec per track
14116 00000AA8 F7F1                             div     cx
14117 00000AAA 95                               xchg    ax, bp ; mov bp,ax
14118 00000AAB A1[9E04]                         mov     ax, [saved_word]
14119 00000AAE F7F1                             div     cx              ; [es:di+BDS.secpertrack]
14120                                                                   ; now, bp:ax = track #, dx = sector
14121                                                                   ; sector number is 1 based.
14122 00000AB0 42                               inc     dx
14123 00000AB1 8816[3101]                       mov     [cursec], dl   ; save current sector
14124 00000AB5 268B4D15                         mov     cx, [es:di+15h] ; [es:di+BDS.heads]
14125                                                                   ; get number of heads
14126                                           ; 22/12/2023
14127                                           ;push   ax ; *
14128 00000AB9 31D2                             xor     dx, dx
14129 00000ABB 95                               xchg    ax, bp ; bp = *       ; divide tracks by heads per cylinder
14130 00000ABC F7F1                             div     cx
14131 00000ABE 95                               xchg    ax, bp ; ax = *, bp = **
14132                                           ;pop    ax ; *
14133 00000ABF F7F1                             div     cx              ; now, bp:ax = cylinder #, dx = head
14134 00000AC1 09ED                             or      bp, bp ; ** = 0 ?
14135                                           ;pop    bp ; ! ; 22/12/2023
14136                                           ;jnz    short baddrive_brdg
14137                                           ; 22/12/2023
14138 00000AC3 7586                             jnz     short baddrive
14139
14140                                           ;cmp    ax, 1024       ; 2^10 currently maximum for track #.
14141                                           ;jnb    short baddrive_brdg
14142                                           ; 22/12/2023
14143 00000AC5 80FC04                           cmp     ah, 4   ; if ax >= 4*256 (1024)
```

```
14144 00000AC8 7381                           jnb     short baddrive
14145
14146 00000ACA 8816[3201]                     mov     [curhd], dl     ; save current head
14147 00000ACE A3[3301]                       mov     [curtrk], ax    ; save current track
14148
14149                                 ; we are now set up for the i/o. normally, we consider the dma boundary
14150                                 ; violations here. not true. we perform the operation as if everything is
14151                                 ; symmetric; let the int 13 handler worry about the dma violations.
14152
14153 00000AD1 A1[2201]                       mov     ax, [seccnt]
14154 00000AD4 E81F01                       call    block
14155                                       ;call   done
14156                                       ;retn
14157                                       ; 22/12/2023
14158 00000AD7 E9E500                       jmp     done
14159
14160                                 ; ---------------------------------------------------------------------------
14161
14162                                 set_lbarw_1:
14163 00000ADA A1[9E04]                     mov     ax, [saved_word] ; check for mini disk
14164                                                                ; (logical dos drive/partition)
14165 00000ADD 26837D7901                   cmp     word [es:di+79h], 1 ; [di+BDS.bdsm_ismini]
14166                                                                 ; logical dos partition
14167 00000AE2 750F                         jnz     short set_lbarw_2 ; not a logical dos partition/drive
14168 00000AE4 26837D7B00                   cmp     word [es:di+7Bh], 0 ; [di+BDS.bdsm_hidden_trks] (> 0)
14169 00000AE9 7408                         jz      short set_lbarw_2
14170 00000AEB 26034517                     add     ax, [es:di+17h]       ; [es:di+BDS.hiddensecs]
14171 00000AEF 26135519                     adc     dx, [es:di+19h]       ; [es:di+BDS.hiddensecs+2]
14172
14173                                 set_lbarw_2:
14174 00000AF3 2EA3[040A]                   mov     [cs:dap_lba_value], ax
14175 00000AF7 2E8916[060A]                 mov     [cs:dap_lba_value+2], dx
14176 00000AFC 2E891E[000A]                 mov     [cs:dap_trans_buf], bx
14177 00000B01 A1[A804]                     mov     ax, [xfer_seg]
14178 00000B04 2EA3[020A]                   mov     [cs:dap_trans_buf+2], ax
14179 00000B08 A1[2201]                     mov     ax, [seccnt]
14180 00000B0B 2EA3[FE09]                   mov     [cs:dap_block_cnt], ax
14181 00000B0F BD0500                       mov     bp, 5
14182 00000B12 892E[A304]                   mov     [vretry_cnt], bp ; verify op. retry cnt for write-verify
14183 00000B16 892E[A504]                   mov     [soft_ecc_cnt], bp ; soft ecc error retry count
14184
14185                                 set_lbarw_3:
14186 00000B1A 268A5504                     mov     dl, [es:di+4]  ; [es:di+BDS.drivenum]
14187 00000B1E 8A26[2001]                   mov     ah, [rflag]    ; get read/write indicator
14188 00000B22 80C440                       add     ah, 40h
14189 00000B25 30C0                         xor     al, al
14190 00000B27 1E                           push    ds
14191 00000B28 0E                           push    cs
14192 00000B29 1F                           pop     ds
14193 00000B2A BE[FC09]                     mov     si, LBA_Packet
14194 00000B2D CD13                         int     13h             ; LBA read/write
14195 00000B2F 1F                           pop     ds
14196 00000B30 731A                         jnc     short set_lbarw_7
14197 00000B32 E8AC02                       call    again
14198                                 set_lbarw_9:
14199 00000B35 7503                         jnz     short set_lbarw_4
14200 00000B37 E92B02                       jmp     harderr
14201                                 ; ---------------------------------------------------------------------------
14202
14203                                 set_lbarw_4:
14204                                 ;set_lbarw_9:      ; 22/12/2023
14205 00000B3A 80FCCC                       cmp     ah, 0CCh        ; Write fault (hard disk)
14206 00000B3D 7505                         jnz     short set_lbarw_5
14207 00000B3F BD0100                       mov     bp, 1
14208                                       ;jmp    short set_lbarw_6
14209                                       ; 17/04/2024
14210 00000B42 EBD6                         jmp     short set_lbarw_3
14211                                 ; ---------------------------------------------------------------------------
14212
14213                                 set_lbarw_5:
14214                                 set_lbarw_10:      ; 22/12/2023
14215 00000B44 C706[A504]0500               mov     word [soft_ecc_cnt], 5 ; soft ecc error retry count
14216                                 set_lbarw_6:
14217                                 set_lbarw_11:
14218 00000B4A EBCE                         jmp     short set_lbarw_3
14219                                 ; ---------------------------------------------------------------------------
14220
14221                                 set_lbarw_7:
14222 00000B4C 813E[2001]0301               cmp     word [rflag], 103h
14223 00000B52 7523                         jnz     short set_lbarw_12
14224 00000B54 B444                         mov     ah, 44h
14225 00000B56 1E                           push    ds
14226 00000B57 0E                           push    cs
14227 00000B58 1F                           pop     ds
14228 00000B59 CD13                         int     13h             ; DISK - IBM/MS Extension - VERIFY SECTORS
14229                                                               ;  (DL - drive, [SI - disk address packet)
14230 00000B5B 1F                           pop     ds
14231 00000B5C 7319                         jnc     short set_lbarw_12
14232 00000B5E 80FC11                       cmp     ah, 11h         ; ECC corrected data error (soft error - retried OK )
14233 00000B61 7506                         jnz     short set_lbarw_8
14234 00000B63 FF0E[A504]                   dec     word [soft_ecc_cnt]
14235                                 ;set_lbarw_8:
14236 00000B67 740E                         jz      short set_lbarw_12
14237                                 set_lbarw_8:
14238 00000B69 E8CF07                       call    ResetDisk
14239 00000B6C 80FC11                       cmp     ah, 11h
14240 00000B6F 74D9                         jz      short set_lbarw_11
14241 00000B71 FF0E[A304]                   dec     word [vretry_cnt]
14242                                       ;jnz    short set_lbarw_9
14243                                       ;jmp    harderr
14244                                       ; 22/12/2023
14245 00000B75 EBBE                         jmp     short set_lbarw_9
14246
14247                                 ; ---------------------------------------------------------------------------
14248                                 ;           ; 22/12/2023
14249                                 ;set_lbarw_9:
14250                                 ;           cmp     ah, 0CCh
14251                                 ;           jnz     short set_lbarw_10
14252                                 ;           mov     bp, 1
14253                                 ;           jmp     short set_lbarw_11
14254                                 ; ---------------------------------------------------------------------------
14255                                 ;           ; 22/12/2023
14256                                 ;set_lbarw_10:
14257                                 ;           mov     word [soft_ecc_cnt], 5 ; soft ecc error retry count
14258                                 ;set_lbarw_11:
14259                                 ;           jmp     short set_lbarw_3
14260                                 ; ---------------------------------------------------------------------------
14261
14262                                 set_lbarw_12:
14263 00000B77 31C0                         xor     ax, ax
14264                                 skip_dpt_setting:  ; 23/12/2023
14265 00000B79 C3                           retn
14266                                       ;;;     ; 22/12/2023
14267                                 %endif
```

```
14268
14269                                 ; ---------------------------------------------------------------------------
14270
14271                                           ; 22/12/2023
14272                                 ;baddrive_brdg:
14273                                           ;jmp   baddrive
14274
14275                                 ; =============== S U B   R O U T I N E =======================================
14276
14277                                 ;--------------------------------------------------------------
14278                                 ; set the drive-last-accessed flag for diskette only.
14279                                 ; we know that the hard disk will not be removed.
14280                                 ; es:di -> current bds.
14281                                 ; ds -> Bios_Data
14282                                 ; ax,cx,si are destroyed.
14283                                 ;--------------------------------------------------------------
14284
14285                                           ; 23/12/2023 - Retro DOS v5.0
14286
14287                                           ; 19/10/2022
14288                                 iosetup:
14289 00000B7A 268A4504                         mov    al, [es:di+4]  ; [es:di+BDS.drivenum]
14290 00000B7E A2[1E01]                         mov    [tim_drv], al  ; save drive letter
14291
14292                                 ; determine proper head settle values
14293
14294 00000B81 803E[A905]00                     cmp    byte [media_set_for_format], 0
14295 00000B86 75F1                             jnz    short skip_dpt_setting
14296 00000B88 A0[2C01]                         mov    al, [eot]      ; fetch up eot before changing ds
14297 00000B8B 1E                               push   ds
14298 00000B8C C536[2D01]                       lds    si, [dpt]      ; get pointer to disk base table
14299 00000B90 884404                           mov    [si+4],al
14300
14301                                           ;; 23/12/2023
14302                                           ;mov   ah, al
14303                                           ;mov   al, [si+10]    ; [si+DISK_PARMS.DISK_MOTOR_STRT]
14304                                           ;;mov  ah, [si+4]     ; [si+DISK_PARMS.DISK_EOT]
14305                                           ;pop   ds
14306                                           ;mov   [motorstartup], al
14307                                           ;mov   [save_eot], ah
14308                                           ; 06/04/2024
14309 00000B93 8A640A                          mov    ah,[si+10]
14310 00000B96 1F                              pop    ds
14311 00000B97 8826[2601]                      mov    [motorstartup], ah
14312 00000B9B A2[2B01]                        mov    [save_eot], al
14313
14314                                 ; for 3.5" drives, both external as well as on the k09, we need to set the
14315                                 ; motor start time to 4. this checking for every i/o is going to affect
14316                                 ; performance across the board, but is necessary!!
14317
14318 00000B9E 1E                               push   ds
14319 00000B9F C536[2D01]                      lds    si, [dpt]      ; get pointer to disk base table
14320                                           ; 23/12/2023  - Retro DOS v5.0
14321 00000BA3 26807D3E02                      cmp    byte [es:di+3Eh], 2 ; (PCDOS 7.1 IBMBIO.COM)
14322                                           ;cmp   byte [es:di+22h], 2 ; [es:di+BDS.formfactor]
14323                                                                      ; ffSmall
14324 00000BA8 7505                            jnz    short motor_start_ok
14325 00000BAA B004                            mov    al, 4
14326 00000BAC 86440A                          xchg   al, [si+10]    ; [si+DISK_PARMS.DISK_MOTOR_STRT]
14327                                 motor_start_ok:
14328
14329                                 ; ds:si now points to disk parameter table.
14330                                 ; get current settle and set fast settle
14331
14332                                           ;xor   al, al
14333                                           ;inc   al             ; ibm wants fast settle      to be 1
14334                                           ; 18/12/2022
14335 00000BAF 31C0                            xor    ax, ax
14336 00000BB1 40                              inc    ax
14337 00000BB2 864409                          xchg   al, [si+9]     ; [si+DISK_PARMS.DISK_HEAD_STTL]
14338                                                                 ; get settle and set up      for fast
14339 00000BB5 1F                              pop    ds
14340 00000BB6 A2[2701]                        mov    [settlecurrent], al
14341 00000BB9 B00F                            mov    al, 15         ; NORMSETTLE
14342                                                                 ; someone has diddled the settle
14343 00000BBB A2[2801]                        mov    [settleslow], al
14344                                           ; 23/12/2023
14345                                 ;skip_dpt_setting:
14346 00000BBE C3                              retn
14347
14348                                 ; =============== S U B   R O U T I N E =======================================
14349
14350                                 ;--------------------------------------------------------------
14351                                 ; set time of last access, and reset default values in the dpt.
14352                                 ;
14353                                 ;     note: trashes (at least) si
14354                                 ;--------------------------------------------------------------
14355
14356                                           ; 23/12/2023 - Retro DOS v5.0
14357
14358                                           ; 19/10/2022
14359                                 done:
14360                                           ;test  byte [es:di+23h], 1 ; [es:di+BDS.flags]
14361                                                                      ; fnon_removable
14362                                           ; 23/12/2023
14363 00000BBF 26F6453F01                      test   byte [es:di+3Fh], 1 ; (PCDOS 7.1 IBMBIO.COM)
14364 00000BC4 752F                            jnz    short ddbx     ; do not set for non-removable media
14365 00000BC6 E8F801                          call   set_tim
14366                                 ;diddleback:
14367                                 ; 09/12/2022
14368                                 diddle_back:
14369 00000BC9 9C                              pushf
14370 00000BCA 803E[A905]00                    cmp    byte [media_set_for_format], 0
14371 00000BCF 7523                            jnz    short nodiddleback
14372 00000BD1 50                              push   ax
14373 00000BD2 06                              push   es
14374 00000BD3 C436[2D01]                      les    si, [dpt]
14375 00000BD7 A0[2B01]                        mov    al, [save_eot]
14376 00000BDA 26884404                        mov    [es:si+4], al  ; [es:si+DISK_PARMS.DISK_EOT]
14377 00000BDE A0[2701]                        mov    al, [settlecurrent]
14378 00000BE1 8A26[2601]                      mov    ah, [motorstartup]
14379 00000BE5 26884409                        mov    [es:si+9], al  ; [es:si+DISK_PARMS.DISK_HEAD_STTL]
14380 00000BE9 26C6440302                      mov    byte [es:si+3], 2 ; [es:si+DISK_PARMS.DISK_SECTOR_SIZ]
14381 00000BEE 2688640A                        mov    [es:si+0Ah], ah       ; [es:si+DISK_PARMS.DISK_MOTOR_STRT]
14382 00000BF2 07                              pop    es
14383 00000BF3 58                              pop    ax
14384                                 nodiddleback:
14385 00000BF4 9D                              popf
14386                                 ddbx:
14387 00000BF5 C3                              retn
14388
14389                                 ; =============== S U B   R O U T I N E =======================================
14390
14391                                 ;--------------------------------------------------------------
```

```
                                ;read the number of sectors specified in ax,
                                ;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 TI 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.
```

```
14516                                   disk_not_mini:
14517 00000C68 D0CE                             ror     dh, 1
14518 00000C6A D0CE                             ror     dh, 1
14519 00000C6C 0A36[3101]                       or      dh, [cursec]
14520 00000C70 89D1                             mov     cx, dx
14521 00000C72 86CD                             xchg    ch, cl          ;  cl = sector, ch = cylinder
14522 00000C74 8A36[3201]                       mov     dh, [curhd]     ; load current head number and
14523 00000C78 268A5504                         mov     dl, [es:di+4]   ; physical drive number
14524                                                                   ; [es:di+BDS.drivenum]
14525                                           ; 23/12/2023
14526 00000C7C 26807D3E05                       cmp     byte [es:di+3Eh], 5
14527                                           ;cmp    byte [es:di+22h], 5 ; [es:di+BDS.formfactor], ffHardFile
14528 00000C81 7411                             jz      short do_fast  ; hard files use fast speed
14529
14530                                   ; if we have [step_drv] set to -1, we use the slow settle time.
14531                                   ; this helps when we have just done a reset disk operation and the head has
14532                                   ; been moved to another cylinder - the problem crops up with 3.5" drives.
14533
14534 00000C83 803E[7600]FF                       cmp     byte [step_drv], 0FFh ; -1
14535                                           ;jz     short do_writej
14536                                           ; 23/12/2023
14537 00000C88 746A                             jz      short do_write
14538 00000C8A 80FC02                           cmp     ah, 2           ; romread
14539 00000C8D 7405                             jz      short do_fast
14540 00000C8F 80FC04                           cmp     ah, 4           ; romverify
14541                                           ;jz     short do_fast
14542                                           ; 23/12/2023
14543 00000C92 7560                             jnz     short do_write
14544                                   ;do_writej:
14545
14546                                   ; reads always fast, unless we have just done a disk reset operation
14547
14548                                           ;jmp    short do_write ; reads always fast
14549                                   ; ---------------------------------------------------------------------------
14550
14551                                   do_fast:
14552 00000C94 E80501                           call    fastspeed       ; change settle       mode
14553                                   testerr:
14554 00000C97 7297                             jb      short dskerr_brdg
14555
14556                                           ; 23/12/2023 Retro DOS v5.0
14557                                           ; (PCDOS 7.1 IBMBIO.COM)
14558 00000C99 83FD05                           cmp     bp, 5           ; is there retry ?
14559 00000C9C 7505                             jnz     short testerror         ; yes
14560 00000C9E 80FCBB                           cmp     ah, 0BBh        ; Undefined error (hard disk)
14561 00000CA1 748D                             jz      short dskerr_brdg
14562                                   testerror:
14563
14564                                   ; set drive and track of last access
14565
14566 00000CA3 8816[7600]                       mov     [step_drv], dl
14567                                           ; 23/12/2023
14568 00000CA7 26886D78                         mov     [es:di+78h], ch
14569                                           ;mov    [es:di+46h], ch         ; [es:di+BDS.track]
14570                                   no_set:
14571                                           ;cmp    word [wrtverify], 103h
14572 00000CAB 813E[2001]0301                   cmp     word [rflag], 103h ; check for write and verify
14573 00000CB1 7452                             jz      short doverify
14574                                   noverify:
14575 00000CB3 58                               pop     ax
14576
14577                                   ; check the flags word in the bds to see if the drive is non removable
14578                                   ; if not we needn't do anything special
14579                                   ; if it is a hard disk then check to see if multi-track operation
14580                                   ; is specified. if specified we don't have to calculate for the next
14581                                   ; track since we are already done. so we can go to the exit of this routine.
14582
14583                                           ; 23/12/2023
14584 00000CB4 26F6453F01                       test    byte [es:di+3Fh], 1
14585                                           ;test   byte [es:di+23h], 1 ; [es:di+BDS.flags]
14586                                                                   ; fnon_removable
14587 00000CB9 7407                             jz      short its_removable
14588 00000CBB F606[A004]80                     test    byte [multrk_flag], 80h ; multrk_on
14589 00000CC0 7530                             jnz     short disk_ret
14590                                   its_removable:
14591 00000CC2 80E13F                           and     cl, 3Fh         ; eliminate cylinder bits from sector
14592 00000CC5 30E4                             xor     ah, ah
14593 00000CC7 2906[2201]                       sub     [seccnt], ax    ; reduce count of sectors to go      next sector
14594 00000CCB 00C1                             add     cl, al
14595 00000CCD 880E[3101]                       mov     [cursec], cl
14596 00000CD1 263A4D13                         cmp     cl, [es:di+13h]         ; [es:di+BDS.secpertrack]
14597                                                                   ; see if sector/track limit reached
14598 00000CD5 761B                             jbe     short disk_ret
14599 00000CD7 C606[3101]01                     mov     byte [cursec], 1 ; start with first sector of next track
14600 00000CDC 8A36[3201]                       mov     dh, [curhd]
14601 00000CE0 FEC6                             inc     dh
14602 00000CE2 263A7515                         cmp     dh, [es:di+15h]         ; [es:di+BDS.heads]
14603 00000CE6 7206                             jb      short noxor
14604 00000CE8 30F6                             xor     dh, dh
14605 00000CEA FF06[3301]                       inc     word [curtrk]
14606                                   noxor:
14607 00000CEE 8836[3201]                       mov     [curhd], dh
14608                                   disk_ret:
14609 00000CF2 F8                               clc
14610 00000CF3 C3                               retn
14611                                   ; ---------------------------------------------------------------------------
14612
14613                                   ; 15/10/2022
14614
14615                                   ; 24/12/2023 - Retro DOS v5.0
14616
14617                                   ;--------------------------------------------------------------
14618                                   ; the request is for write. determine if we are talking about
14619                                   ; the same track and drive. if so, use the fast speed.
14620                                   ;--------------------------------------------------------------
14621
14622                                   do_write:
14623 00000CF4 3A16[7600]                       cmp     dl, [step_drv]
14624 00000CF8 7506                             jnz     short do_norm  ; we have changed drives
14625                                           ; 24/12/2023
14626 00000CFA 263A6D78                         cmp     ch, [es:di+78h]
14627                                           ;cmp    ch, [es:di+46h]         ; [es:di+BDS.track]
14628 00000CFE 7494                             jz      short do_fast  ; we are still on the same track
14629                                   do_norm:
14630 00000D00 E87500                           call    normspeed
14631 00000D03 EB92                             jmp     short testerr
14632                                   ; ---------------------------------------------------------------------------
14633
14634                                   ;--------------------------------------------------------------
14635                                   ; we have a verify request also. get state info and go verify
14636                                   ;--------------------------------------------------------------
14637
14638                                   doverify:
14639 00000D05 58                               pop     ax
```

```
14640 00000D06 50                       push    ax
14641 00000D07 B404                     mov     ah, 4
14642 00000D09 E89000                   call    fastspeed
14643 00000D0C 73A5                     jnb     short noverify
14644
14645                           ; check the error returned in ah to see if it is a soft ecc error.
14646                           ; if it is not we needn't do anything special. if it is a soft
14647                           ; ecc error then decrement the soft_ecc_cnt error retry count. if
14648                           ; this retry count becomes 0 then we just ignore the error and go to
14649                           ; no_verify but if we can still try then we call the routine to reset
14650                           ; the disk and go to dskerr1 to retry the operation.
14651
14652 00000D0E 80FC11                   cmp     ah, 11h         ; soft ecc error ?
14653 00000D11 750B                     jnz     short not_softecc_err
14654 00000D13 FF0E[A504]               dec     word [soft_ecc_cnt]
14655 00000D17 749A                     jz      short noverify ; no more retry
14656 00000D19 E81F06                   call    ResetDisk       ; reset disk
14657 00000D1C EB3E                     jmp     short dskerr1  ; retry
14658                           ; ---------------------------------------------------------------------------
14659
14660                           not_softecc_err:                ; other error.
14661 00000D1E E81A06                   call    ResetDisk
14662 00000D21 FF0E[A304]               dec     word [vretry_cnt]
14663 00000D25 EB1C                     jmp     short dskerr0
14664                           ; ---------------------------------------------------------------------------
14665
14666                           ;--------------------------------------------------------------
14667                           ; need to special case the change-line error ah=06h.
14668                           ; if we get this, we need to return it.
14669                           ;--------------------------------------------------------------
14670
14671                           dskerr:
14672 00000D27 803E[7700]00             cmp     byte [fhave96], 0     ; do we have changeline       support?
14673 00000D2C 7403                     jz      short dskerr_nochangeline ; brif not
14674 00000D2E E8BE0E                   call    checkio
14675                           dskerr_nochangeline:
14676 00000D31 803E[A704]01             cmp     byte [multitrk_format_flag], 1 ; multi trk format request?
14677 00000D36 7508                     jnz     short dochkagain ; no more retry.
14678 00000D38 BD0100                   mov     bp, 1
14679 00000D3B C606[A704]00             mov     byte [multitrk_format_flag], 0 ; clear the flag.
14680                           dochkagain:
14681 00000D40 E89E00                   call    again
14682                           dskerr0:
14683 00000D43 7420                     jz      short harderr
14684                                   ; 24/12/2023
14685 00000D45 26F6453F01               test    byte [es:di+3Fh], 1
14686                                   ;test   byte [es:di+23h], 1 ; [es:di+BDS.flags]
14687                                                   ; fnon_removable
14688 00000D4A 7505                     jnz     short skip_timeout_chk
14689 00000D4C 80FC80                   cmp     ah, 80h         ; timeout?
14690 00000D4F 7414                     jz      short harderr
14691                           skip_timeout_chk:
14692 00000D51 80FCCC                   cmp     ah, 0CCh        ; write fault error?
14693 00000D54 740A                     jz      short write_fault_err ;       then, don't retry.
14694 00000D56 C706[A504]0500           mov     word [soft_ecc_cnt], 5 ; MAXERR
14695                                                   ; set soft_ecc_cnt back       to maxerr
14696                           dskerr1:
14697 00000D5C 58                       pop     ax              ; restore sector count
14698                                   ;jmp    retry
14699                                   ; 09/12/2022
14700 00000D5D E9F1FE                   jmp     _retry
14701                           ; ---------------------------------------------------------------------------
14702
14703                           write_fault_err:
14704 00000D60 BD0100                   mov     bp, 1           ; just retry only once
14705                                                   ; for write fault error.
14706 00000D63 EBF7                     jmp     short dskerr1
14707
14708                                   ; fall into harderr
14709                           ; ---------------------------------------------------------------------------
14710
14711                           ; entry point for routines that call maperror themselves
14712
14713                           harderr:
14714 00000D65 E84100                   call    maperror
14715                           harderr2:
14716 00000D68 C606[1E01]FF             mov     byte [tim_drv], 0FFh
14717                                                   ; force a media check through rom
14718 00000D6D 8B0E[2201]               mov     cx, [seccnt]    ; get count of sectors to go
14719 00000D71 8B26[3501]               mov     sp, [spsav]     ; recover entry       stack pointer
14720
14721                           ; since we are performing a non-local goto, restore the disk parameters
14722
14723                                   ;jmp    diddleback
14724                                   ; 09/12/2022
14725 00000D75 E951FE                   jmp     diddle_back
14726
14727                           ; =============== S U B   R O U T I N E =======================================
14728
14729                           ; change settle value from settlecurrent to whatever is appropriate
14730                           ; note that this routine is never called for a fixed disk.
14731
14732                                   ; 19/10/2022
14733                           normspeed:
14734 00000D78 803E[A905]00             cmp     byte [media_set_for_format], 0
14735 00000D7D 751D                     jnz     short fastspeed
14736 00000D7F 06                       push    es
14737 00000D80 50                       push    ax
14738 00000D81 A0[2801]                 mov     al, [settleslow]
14739 00000D84 C436[2D01]               les     si, [dpt]       ; current disk parm table
14740 00000D88 26884409                 mov     [es:si+9], al  ; [es:si+DISK_PARMS.DISK_HEAD_STTL]
14741 00000D8C 58                       pop     ax
14742 00000D8D 07                       pop     es
14743 00000D8E E80B00                   call    fastspeed
14744                                   ; 24/12/2023
14745                                   ;push   es
14746                                   ;les    si, [dpt]
14747                                   ;mov    byte [es:si+9], 1 ; [es:si+DISK_PARMS.DISK_HEAD_STTL]
14748                                   ;                       ; 1 is fast settle value
14749                                   ;pop    es
14750 00000D91 1E                       push    ds
14751 00000D92 C536[2D01]               lds     si, [dpt]
14752 00000D96 C6440901                 mov     byte [si+9], 1
14753 00000D9A 1F                       pop     ds
14754
14755 00000D9B C3                       retn
14756
14757                           ; =============== S U B   R O U T I N E =======================================
14758
14759                           ; if the drive has been marked as too big (i.e. starting sector of the
14760                           ; partition is > 16 bits, then always return drive not ready.
14761
14762                                   ; 24/12/2023 - Retro DOS v5.0
14763                                   ; (PCDOS 7.1 IBMBIO.COM - BIOSCODE:0DDDh)
```

```
                                fastspeed:
                                        ;;test  byte [es:di+3Bh], 80h ; [es:di+BDS.fatsiz]
                                        ;test   byte [es:di+1Fh], 80h ; [es:di+BDS.fatsiz]
                                        ;                       ; ftoobig
                                        ;jnz    short notready
00000D9C 06                             push    es
00000D9D 8E06[A804]                     mov     es, [xfer_seg]
00000DA1 CD13                           int     13h             ; DISK -
00000DA3 8C06[A804]                     mov     [xfer_seg], es
00000DA7 07                             pop     es
00000DA8 C3                             retn
                                ; ---------------------------------------------------------------------------
                                ;               ; 24/12/2023
                                ;notready:
                                        ;stc
                                        ;mov    ah, 80h
                                        ;retn

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

                                ; map error returned by rom in ah into corresponding code to be returned to
                                ; dos in al. trashes di. guaranteed to set carry.

                                maperror:
00000DA9 51                             push    cx
00000DAA 06                             push    es
00000DAB 1E                             push    ds              ; set es=Bios_Data
00000DAC 07                             pop     es
00000DAD 88E0                           mov     al, ah          ; put error code in al
00000DAF A2[4601]                       mov     [lsterr], al    ; terminate list with error code
                                        ; 24/12/2023
00000DB2 B90B00                         mov     cx, 11 ; PCDOS 7.1 ; 02/09/2023
                                        ;mov    cx, 9           ; numerr (= errout-errin)
                                                                ; number of possible error conditions
00000DB5 BF[3C01]                       mov     di, errin       ; point to error conditions
00000DB8 F2AE                           repne scasb

                                        ; 24/12/2023
                                        ; 02/09/2023
00000DBA 8A450A                         mov     al, [di+10] ; PCDOS 7.1 IBMBIO.COM
                                        ; 10/12/2022
                                        ;mov    al, [di+8]      ; [di+numerr-1]
                                                                ; get translation
                                        ; 19/10/2022 - Temporary !
                                        ;db     8Ah, 85h, 8, 0 ; mov al, [di+8]
00000DBD 07                             pop     es
00000DBE 59                             pop     cx
00000DBF F9                             stc                     ; flag error condition
00000DC0 C3                             retn

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

                                ; set the time of last access for this drive.
                                ; this is done only for removable media. es:di -> bds

                                set_tim:
00000DC1 50                             push    ax
00000DC2 E86CF7                         call    GetTickCnt      ; Does INT 1A ah=0 & updates daycnt

                                ; we have the new time. if we see that the time has passed,
                                ; then we reset the threshold counter...

                                        ; 24/12/2023 - Retro DOS v5.0
00000DC5 263B5579                       cmp     dx, [es:di+79h]         ; PCDOS 7.1 IBMBIO.COM
                                        ;cmp    dx, [es:di+47h]         ; [es:di+BDS.tim_lo]
00000DC9 7506                           jne     short setaccess
                                        ; 24/12/2023
00000DCB 263B4D7B                       cmp     cx, [es:di+7Bh]         ; PCDOS 7.1 IBMBIO.COM
                                        ;cmp    cx, [es:di+49h]         ; [es:di+BDS.tim_hi]
                                        ;jz     short done_set
                                        ; 12/12/2022
00000DCF 740E                           je      short done_set2
                                setaccess:
00000DD1 C606[1D01]00                   mov     byte [accesscount], 0

                                        ; 24/12/2023 - Retro DOS v5.0 (PCDOS 7.1 IBMBIO.COM)
00000DD6 26895579                       mov     [es:di+79h], dx
00000DDA 26894D7B                       mov     [es:di+7Bh], cx
                                        ;mov    [es:di+47h], dx         ; [es:di+BDS.tim_lo]
                                        ;mov    [es:di+49h], cx         ; [es:di+BDS.tim_hi]
                                done_set:
00000DDE F8                             clc
                                done_set2:      ; 12/12/2022
00000DDF 58                             pop     ax
00000DE0 C3                             retn

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

                                ; this routine is called if an error occurs while formatting or verifying.
                                ; it resets the drive,and decrements the retry count.
                                ; on entry - ds:di - points to bds for the drive
                                ;       bp    - contains retry count
                                ; on exit    flags indicate result of decrementing retry count

                                again:
00000DE1 E85705                         call    ResetDisk
00000DE4 80FC06                         cmp     ah, 6
00000DE7 7402                           jz      short dont_dec_retry_count ; If      it is a media change error
                                                                        ; do not decrement retry count.
00000DE9 4D                             dec     bp              ; decrement retry count
00000DEA C3                             retn
                                ; ---------------------------------------------------------------------------

                                dont_dec_retry_count:
00000DEB 08E4                           or      ah, ah
00000DED C3                             retn

                                ;---------------------------------------------------------------------------
                                ; Retro DOS v5.0 - PCDOS 7.1 IBMBIO.COM - BIOSCODE:0E30h
                                ;---------------------------------------------------------------------------
                                ; 24/12/2023 - Retro DOS v5.0
                                ;;;;

00000DEE 00                     ioctl_drvnum:       db 0

                                        ; 24/12/2023

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

                                get_phy_drv_num:
00000DEF E8B2F7                         call    SetDrive        ; get physical drive number
                                                                ; INPUT: al = logical drive number (BDS.drivelet)
                                                                ; OUTPUT: physical drive number (BDS.drivenum)
00000DF2 268A5504                       mov     dl, [es:di+4]   ; [es:di+BDS.drivenum]
```

```
00000DF6 C3                             retn
                                        ; =============== S U B R O U T I N E =======================================
                                        ; 24/12/2023
                                ioctl_output:
00000DF7 E8F5FF                         call    get_phy_drv_num
00000DFA 2E8816[EE0D]                   mov     [cs:ioctl_drvnum], dl
00000DFF B441                           mov     ah, 41h
00000E01 BBAA55                         mov     bx, 55AAh
00000E04 CD13                           int     13h             ; DISK - Check for INT 13h Extensions
                                                                ; BX = 55AAh, DL = drive number
                                                                ; Return: CF set if not supported
                                                                ; AH = extensions version
                                                                ; BX = AA55h
                                                                ; CX = Interface support bit map
00000E06 7235                           jc      short int13h_exts_err
                                ioctl_input_1:
00000E08 C43E[1200]                     les     di, [ptrsav]
00000E0C 26C47D0E                       les     di, [es:di+14] ; [es:di+IOCTL_REQ.MINORFUNCTION]
00000E10 723E                           jc      short ioctl_input_2
00000E12 B80046                         mov     ax, 4600h       ; Eject removable media
00000E15 263805                         cmp     [es:di], al     ; al = 0 ; disk ioctl function = 0
00000E18 7417                           je      short ioctl_output_1
00000E1A 26803D01                       cmp     byte [es:di], 1 ; al = 1 ; disk ioctl function = 1
00000E1E 751B                           jne     short ioctl_output_2
00000E20 B80145                         mov     ax, 4501h       ; Lock/unlock media
                                                                ; (al, 0 = lock, 1 = unlock)
00000E23 26807D0100                     cmp     byte [es:di+1], 0 ; unlock (reverse of INT 13h ah=45h)
00000E28 7407                           jz      short ioctl_output_1
00000E2A 26384501                       cmp     [es:di+1], al  ; lock (reverse of INT 13h ah=45h)
00000E2E 750B                           jne     short ioctl_output_2
00000E30 48                             dec     ax
                                ioctl_output_1:
00000E31 2E8A16[EE0D]                   mov     dl, [cs:ioctl_drvnum]
00000E36 CD13                           int     13h             ; DISK - IBM/MS Extension - LOCK/UNLOCK DRIVE (DL - drive, [SI - disk address
packet)
00000E38 7203                           jc      short int13h_exts_err
                                ioctl_lock_err:
                                        ; cf=0
                                ioctl_output_ret:
                                        ;clc
00000E3A C3                             retn
                                ; ---------------------------------------------------------------------------
                                ioctl_output_2:
00000E3B B401                           mov     ah, 1
                                int13h_exts_err:
00000E3D 80FCB0                         cmp     ah, 0B0h        ; volume not locked in drive
00000E40 74F8                           je      short ioctl_lock_err
00000E42 80FCB4                         cmp     ah, 0B4h        ; lock count exceeded
00000E45 74F3                           je      short ioctl_lock_err
00000E47 E9DAF7                         jmp     err_exitj
                                        ; =============== S U B R O U T I N E =======================================
                                        ; 24/12/2023
                                ioctl_input:
00000E4A E8A2FF                         call    get_phy_drv_num
00000E4D F9                             stc
00000E4E EBB8                           jmp     short ioctl_input_1
                                ioctl_input_2:
00000E50 26803D06                       cmp     byte [es:di], 6         ; disk ioctl function = 6
00000E54 75E5                           jne     short ioctl_output_2
00000E56 B80245                         mov     ax, 4502h       ; get lock status
00000E59 CD13                           int     13h             ; DISK - IBM/MS Extension - LOCK/UNLOCK DRIVE (DL - drive, [SI - disk address
packet)
00000E5B 72E0                           jc      short int13h_exts_err
00000E5D BB0C00                         mov     bx, 0Ch         ; bit 1 lock bit
00000E60 3C00                           cmp     al, 0           ; not locked
00000E62 7402                           jz      short ioctl_input_3
00000E64 B30E                           mov     bl, 0Eh
                                ioctl_input_3:
00000E66 53                             push    bx
00000E67 B404                           mov     ah, 4
00000E69 B90101                         mov     cx, 101h
00000E6C B601                           mov     dh, 1
00000E6E CD13                           int     13h             ; DISK - VERIFY SECTORS
                                                                ; AL = number of sectors to verify, CH = track, CL = sector
                                                                ; DH = head, DL = drive
                                                                ; Return: CF set on error, AH = status
                                                                ; AL = number of sectors verified
00000E70 5B                             pop     bx
00000E71 80FC31                         cmp     ah, 31h         ; no media in drive (IBM/MS INT 13 extensions)
00000E74 740B                           je      short ioctl_input_5
00000E76 80FC80                         cmp     ah, 80h         ; timeout (not ready)
00000E79 7406                           je      short ioctl_input_5
                                ioctl_input_4:
00000E7B 26895D01                       mov     [es:di+1], bx
00000E7F EBB9                           jmp     short ioctl_lock_err
                                ioctl_input_5:
00000E81 81CB0108                       or      bx, 801h        ; bit 0 error bit (1 = error, 31h or 80h)
                                                                ; bit 11 (not ready -removable media error- bit)
                                                                ; if bit 11 = 0, another error (except 31h and 80h)
00000E85 EBF4                           jmp     short ioctl_input_4

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

                                ; 16/10/2022 - Retro DOS 4.0 (Modified MSDOS 5.0 IO.SYS)

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

                                ; 18/03/2019

                                ; ============================================================================
                                ;
                                ; NOTE: GetAccessFlag/SetAccessFlag is unpublished function.
                                ;
                                ;      This function is intended to give the user to control the
                                ;      bds table flags of unformatted_media bit.
                                ;      GetAccessFlag will show the status -
                                ;    a_DiskAccess_Control.dac_access_flag = 0 disk i/o not allowed
                                ;                                           1 disk i/o allowed
                                ;      SetAccessFlag will set/reset the unformatted_media bit in flags -
                                ;    a_DiskAccess_Control.dac_access_flag = 0 allow disk i/o
                                ;                                           1 disallow disk i/o
                                ; ============================================================================

                                        ; generic ioctl dispatch tables
```

```
                                ; BIOSCODE:0C3Ch (MSDOS 6.21, IO.SYS)

                                ; 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)
                                        ; 19/10/2022
                                do_generic_ioctl:               ; 2C7h:0C6Bh = 70h:31DBh
00000ECC E8D5F6                         call    SetDrive        ; ES:DI Points to bds for drive
                                        ; 24/12/2023
                                        ;;;
00000ECF 2EC606[CB0E]00                 mov     byte [cs:new_genioctl], 0
                                                                ; 0, old generic ioctl function
00000ED5 06                             push    es
00000ED6 C41E[1200]                     les     bx, [ptrsav]    ; ES:BX Points to request header
00000EDA 26807F0D08                     cmp     byte [es:bx+0Dh], 8 ; [es:bx+IOCTL_REQ.MAJORFUNCTION]
                                                                ; RAWIO
                                        ; 24/12/2023
                                        ;mov    al, [es:bx+0Eh]         ; [es:bx+IOCTL_REQ.MINORFUNCTION]
                                        ;pop    es
                                        ;jnz    short IoctlFuncErr
00000EDF 740A                           jz      short chk_genioctl_minor
00000EE1 2EFE06[CB0E]                   inc     byte [cs:new_genioctl]
                                                                ; 1, new generic ioctl function (FAT32)
00000EE6 26807F0D48                     cmp     byte [es:bx+0Dh], 48h ; Generic IOCtl Request support
                                                        ; (called only if bit 6 of attribute is set to 1)
                                chk_genioctl_minor:
00000EEB 268A470E                       mov     al, [es:bx+0Eh]         ; [es:bx+IOCTL_REQ.MINORFUNCTION]
00000EEF 07                             pop     es
00000EF0 7525                           jnz     short IoctlFuncErr
                                        ;;;

                                        ; cas note: Could do the above two blocks in reverse order.
                                        ; Would have to preserve al for SetDrive

                                        ; 10/12/2022
00000EF2 BE[870E]                       mov     si, IoReadJumpTable
                                        ;mov    si, 0C3Ch       ; IoReadJumpTable
                                                                ; at 2C7h:0C3Ch       = 70h:31ACh
00000EF5 A820                           test    al, 20h         ; GEN_IOCTL_FN_TST ; test of req. function
00000EF7 7503                           jnz     short NotGenericWrite ; function is a read.
                                        ; 10/12/2022
00000EF9 BE[A80E]                       mov     si, IoWriteJumpTable
                                        ;mov    si, 0C4Fh       ; IoWriteJumpTable
                                                                ; at 2C7h:0C4Fh       = 70h:31BFh
                                NotGenericWrite:
00000EFC 24DF                           and     al, 0DFh        ; ~GEN_IOCTL_FN_TST ; get rid of read/write bit
00000EFE 2C40                           sub     al, 40h         ; offset for base function
00000F00 2E3A04                         cmp     al, [cs:si]
00000F03 7712                           ja      short IoctlFuncErr
00000F05 98                             cbw
                                        ; 24/12/2023
                                        ;shl    ax, 1
00000F06 01C0                           add     ax, ax
00000F08 46                             inc     si
00000F09 01C6                           add     si, ax
00000F0B 2EFF14                         call    near [cs:si]
                                        ;call   word ptr cs:[si]
00000F0E 2E8E1E[3000]                   mov     ds, [cs:BIOSDATAWORD]
                                        ;mov    ds, word ptr cs:BIOSDATAWORD ; [cs:Bios_Data_Word]
                                                                ; 2C7h:30h = 70h:25A0h
00000F13 B481                           mov     ah, 81h         ; Return this status in       case of carry
00000F15 C3                             retn                    ; Pass carry flag through to exit code
                                ; ---------------------------------------------------------------------------

                                        ; Cmd_Error_Proc is called as a procedure and also use
                                        ; as a fall through from above
                                Cmd_Error_Proc:                         ; 2C7h:0CA3h = 70h:3213h
00000F16 5A                             pop     dx
                                IoctlFuncErr:
00000F17 E9BBF1                         jmp     bc_cmderr
                                ; ---------------------------------------------------------------------------

                                ; 16/10/2022

                                ; ===========================================================================
                                ;** GetDeviceParameters:
                                ;
                                ;   GetDeviceParameters implements the generic ioctl function:
                                ;   majorcode=RAWIO, minorcode=GetDeviceParameters (60h)
                                ;
                                ;   ENTRY   (ES:di) = BDS for drive
                                ;           PtrSav = long pointer to request header
                                ;   EXIT    ??? BUGBUG
                                ;   USES    ??? BUGBUG
                                ; ===========================================================================

                                        ; 24/12/2023 - Retro DOS v5.0
                                        ; (PCDOS 7.1 IBMBIO.COM - BIOSCODE:0F5Dh)

                                        ; 19/10/2022
                                GetDeviceParameters:
                                        ; Copy info from bds to the device parameters packet

00000F1A C51E[1200]                     lds     bx, [ptrsav]    ; DS:BX points to request header
00000F1E C55F13                         lds     bx, [bx+19]     ; [bx+IOCTL_REQ.GENERICIOCTL_PACKET]
                                                                ; (DS:BX) = return buffer
                                        ; 24/12/2023
00000F21 268A453E                       mov     al, [es:di+3Eh]
                                        ;mov    al, [es:di+34] ; [es:di+BDS.formfactor]
00000F25 884701                         mov     [bx+1], al      ; [bx+A_DEVICEPARAMETERS.DP_DEVICETYPE]
                                        ; 24/12/2023
00000F28 268B453F                       mov     ax, [es:di+3Fh]
                                        ;mov    ax, [es:di+35] ; [es:di+BDS.flags]
00000F2C 83E003                         and     ax, 3           ; fnon_removable+fchangeline
                                                                ; Mask off other bits
00000F2F 894702                         mov     [bx+2], ax      ; [bx+A_DEVICEPARAMETERS.DP_DEVICEATTRIBUTES]
                                        ; 24/12/2023
00000F32 268B4541                       mov     ax, [es:di+41h]
                                        ;mov    ax, [es:di+37] ; [es:di+BDS.cylinders]
00000F36 894704                         mov     [bx+4], ax      ; [bx+A_DEVICEPARAMETERS.DP_CYLINDERS]
00000F39 30C0                           xor     al, al          ; Set media type to default
00000F3B 884706                         mov     [bx+6], al      ; [bx+A_DEVICEPARAMETERS.DP_MEDIATYPE]

                                        ; copy recommended bpb

                                        ; 24/12/2023
00000F3E 8D7543                         lea     si, [di+43h]
                                        ;lea    si, [di+39]     ; [di+BDS.rbytespersec]       = [di+BDS.R_BPB]
00000F41 F60701                         test    byte [bx], 1    ; [bx+A_DEVICEPARAMETERS.DP_SPECIALFUNCTIONS]
                                                                ; BUILD_DEVICE_BPB
00000F44 7412                           jz      short UseBpbPresent
00000F46 1E                             push    ds              ; Save request packet segment
00000F47 2E8E1E[3000]                   mov     ds, [cs:BIOSDATAWORD]
                                        ;mov    ds, word ptr cs:BIOSDATAWORD ; [cs:Bios_Data_Word]
                                                                ; 2C7h:30h = 70h:25A0h
```

```
15258                                                   ; Point back to Bios_Data
15259 00000F4C E828FA                             call  checksingle
15260 00000F4F E884F7                             call  GetBp           ; Build the bpb from scratch
15261 00000F52 1F                                 pop   ds              ; Restore request packet segment
15262 00000F53 7224                               jb    short GetParmRet
15263 00000F55 8D7506                             lea   si, [di+6]      ; [di+BDS.bytespersec] = [di+BSD.DP_BPB]
15264                                                   ; Use this subfield of bds instead
15265                                   UseBpbPresent:
15266 00000F58 8D7F07                             lea   di, [bx+7]      ; [bx+A_DEVICEPARAMETERS.DP_BPB]
15267                                                   ; This is where       the result goes
15268                                             ; 24/12/2023
15269 00000F5B 31D2                               xor   dx, dx ; 0
15270
15271                                             ; 24/12/2023
15272 00000F5D B91F00                             mov   cx, 31          ; A_BPB.size = 31
15273                                             ;mov  cx, 25          ; A_BPB.size - 6
15274                                                   ; For now use 'small' bpb
15275                                             ; 24/12/2023
15276                                             ;;;
15277 00000F60 2E3816[CB0E]                       cmp   [cs:new_genioctl], dl ; 0 ? ; *
15278 00000F65 7404                               jz    short gdp_1     ; old type (FAT12 & FAT16) structure
15279                                             ;mov  cx, 53          ; FAT32 BPB size
15280                                             ;mov  dx, 32          ; 53+32 = 85 bytes (A_BPB_FAT32.size)
15281 00000F67 B135                               mov   cl, 53
15282 00000F69 B220                               mov   dl, 32
15283                                   gdp_1:
15284                                             ;;;
15285 00000F6B 1E                                 push  ds              ; reverse segments for copy
15286 00000F6C 06                                 push  es
15287 00000F6D 1F                                 pop   ds
15288 00000F6E 07                                 pop   es
15289 00000F6F F3A4                               rep movsb
15290
15291                                             ; 24/12/2023
15292                                             ;;;
15293 00000F71 89D1                               mov   cx, dx          ; 0 or 32
15294 00000F73 E304                               jcxz  gdp_2
15295 00000F75 30C0                               xor   al, al          ; 32 zeros
15296 00000F77 F3AA                               rep stosb
15297                                   gdp_2:
15298                                             ;clc    ; cf is already 0 ; * ; 24/12/2023
15299                                             ;;;
15300
15301                                             ; 12/12/2022
15302                                             ; cf=0 (cmp instruction -above- resets cf)
15303                                             ;clc
15304                                   GetParmRet:
15305 00000F79 C3                                 retn
15306                                   ; ---------------------------------------------------------------------------
15307
15308                                   ; 17/10/2022
15309                                   ; 16/10/2022
15310
15311                                   ; ============================================================================
15312                                   ; SetDeviceParameters:
15313                                   ;
15314                                   ; input: ES:di points to bds for drive
15315                                   ; ============================================================================
15316
15317                                             ; 24/12/2023 - Retro DOS v5.0
15318                                             ; (PCDOS 7.1 IBMBIO.COM - BIOSCODE:0FC0h)
15319
15320                                             ; 19/10/2022
15321                                   SetDeviceParameters:                  ; 2C7h:0CF3h = 70h:3263h
15322 00000F7A C51E[1200]                         lds   bx, [ptrsav]    ; DS:BX points to request header
15323 00000F7E C55F13                             lds   bx, [bx+19]     ; [bx+IOCTL_REQ.GENERICIOCTL_PACKET]
15324                                             ; 24/12/2023
15325 00000F81 26814D3F4001                       or    word [es:di+3Fh], 140h
15326                                             ;or   word [es:di+23h], 140h ; [es:di+BDS.flags]
15327                                                                   ; fchanged_by_format|fchanged
15328 00000F87 F60702                             test  byte [bx], 2    ; [bx+A_DEVICEPARAMETERS.DP_SPECIALFUNCTIONS]
15329                                                                   ; ONLY_SET_TRACKLAYOUT
15330                                             ;jnz  short setTrackTable
15331                                             ; 24/12/2023
15332 00000F8A 7403                               jz    short sdp_1
15333 00000F8C E98000                             jmp   setTrackTable
15334                                   sdp_1:
15335 00000F8F 8A4701                             mov   al, [bx+1]      ; [bx+A_DEVICEPARAMETERS.DP_DEVICETYPE]
15336                                             ; 24/12/2023
15337 00000F92 2688453E                           mov   [es:di+3Eh], al
15338                                             ;mov  [es:di+34], al ; [es:di+BDS.formfactor]
15339 00000F96 8B4704                             mov   ax, [bx+4]      ; [bx+A_DEVICEPARAMETERS.DP_CYLINDERS]
15340                                             ; 24/12/2023
15341 00000F99 26894541                           mov   [es:di+41h], ax
15342                                             ;mov  [es:di+37], ax ; [es:di+BDS.cylinders]
15343 00000F9D 8B4702                             mov   ax, [bx+2]      ; [bx+A_DEVICEPARAMETERS.DP_DEVICEATTRIBUTES]
15344 00000FA0 1E                                 push  ds
15345                                             ; 17/10/2022
15346 00000FA1 2E8E1E[3000]                       mov   ds, [cs:BIOSDATAWORD]
15347                                             ;mov  ds, word ptr cs:BIOSDATAWORD ; [cs:Bios_Data_Word]
15348                                                                   ; 2C7h:30h = 70h:25A0h
15349                                             ;cmp  byte [fhave96], 0
15350 00000FA6 803E[7700]00                       cmp   byte [fhave96], 0
15351 00000FAB 1F                                 pop   ds
15352 00000FAC 7502                               jnz   short HaveChange ; we have changeline support
15353                                             ; 10/12/2022
15354 00000FAE 24FD                               and   al, 0FDh
15355                                             ;and  ax, 0FFFDh      ; ~fchangeline
15356
15357                                             ; Ignore all bits except non_removable and changeline
15358                                   HaveChange:
15359 00000FB0 83E003                             and   ax, 3           ; fnon_removable|fchangeline
15360                                             ; 24/12/2023
15361 00000FB3 268B4D3F                           mov   cx, [es:di+3Fh]
15362                                             ;mov  cx, [es:di+35] ; [es:di+BDS.flags]
15363 00000FB7 81E1F4FD                           and   cx, 0FDF4h      ; ~(fnon_removable|fchangeline|good_tracklayout|unformatted_media)
15364 00000FBB 09C8                               or    ax, cx
15365                                             ; 24/12/2023
15366 00000FBD 2689453F                           mov   [es:di+3Fh], ax
15367                                             ;mov  [es:di+35], ax ; [es:di+BDS.flags]
15368 00000FC1 8A4706                             mov   al, [bx+6]      ; [bx+A_DEVICEPARAMETERS.DP_MEDIATYPE]
15369                                                                   ; Set media type
15370 00000FC4 1E                                 push  ds
15371 00000FC5 2E8E1E[3000]                       mov   ds, [cs:BIOSDATAWORD]
15372                                             ;mov  ds, word ptr cs:BIOSDATAWORD ; [cs:Bios_Data_Word]
15373 00000FCA A2[A805]                           mov   [mediatype], al
15374                                             ;mov  ds:mediatype, al
15375
15376                                             ; 24/12/2023
15377                                             ;;;
15378 00000FCD B93500                             mov   cx, 53          ; FAT32 BPB size
15379 00000FD0 2E803E[CB0E]00                     cmp   byte [cs:new_genioctl], 0
15380 00000FD6 7502                               jnz   short sdp_2     ; new type (FAT32) structure
15381                                             ;mov  cx, 31          ; A_BPB.size = 31
```

```
15382 00000FD8 B11F                             mov     cl, 31
15383                                   sdp_2:
15384                                           ;;;
15385 00000FDA 1F                               pop     ds
15386
15387                                           ; The media changed (maybe) so we will have to do a set dasd
15388                                           ; the next time we format a track
15389
15390                                           ; 24/12/2023
15391 00000FDB 26804D3F80                       or      byte [es:di+3Fh], 80h
15392                                           ; 10/12/2022
15393                                           ;or     byte [es:di+35], 80h
15394                                           ;;or    word [es:di+35], 80h ; [es:di+BDS.flags]
15395                                                                   ; set_dasd_true
15396 00000FE0 57                               push    di              ; Save bds pointer
15397
15398                                           ; Figure out what we are supposed to do with the bpb
15399                                           ; were we asked to install a fake bpb?
15400
15401 00000FE1 F60701                           test    byte [bx], 1    ; [bx+A_DEVICEPARAMETERS.DP_SPECIALFUNCTIONS]
15402                                                                   ; INSTALL_FAKE_BPB
15403 00000FE4 7511                             jnz     short InstallFakeBpb
15404
15405                                           ; were we returning a fake bpb when asked to build a bpb?
15406
15407                                           ; 24/12/2023
15408 00000FE6 26F6453F04                       test    byte [es:di+3Fh], 4
15409                                           ; 10/12/2022
15410                                           ;test   byte [es:di+35], 4
15411                                           ;;test  word [es:di+35], 4 ; [es:di+BDS.flags]
15412                                                                   ; return_fake_bpb
15413 00000FEB 7405                             jz      short InstallRecommendedBpb
15414
15415                                           ; we were returning a fake bpb but we can stop now
15416
15417                                           ; 24/12/2023
15418 00000FED 2680653FFB                       and     byte [es:di+3Fh], 0FBh
15419                                           ; 10/12/2022
15420                                           ;and    byte [es:di+35], 0FBh
15421                                           ;;and   word [es:di+35], 0FFFBh ; [es:di+BDS.flags]
15422                                                                   ; ~return_fake_bpb
15423                                   InstallRecommendedBpb:
15424                                           ; 24/12/2023
15425                                           ;mov    cx, 31          ; A_BPB.size
15426                                           ;lea    di, [di+27h]    ; [di+BDS.R_BPB] = [di+BDS.rbytespersec]
15427                                           ; cx = 53 or 31
15428 00000FF2 8D7D43                           lea     di, [di+43h]    ; (PCDOS 7.1 IBMBIO.COM)
15429 00000FF5 EB08                             jmp     short CopyTheBpb
15430                                   ; ---------------------------------------------------------------------------
15431
15432                                   InstallFakeBpb:
15433                                           ; 24/12/2023
15434 00000FF7 26804D3F04                       or      byte [es:di+3Fh], 4
15435                                           ; 10/12/2022
15436                                           ;or     byte [es:di+35], 4
15437                                           ;;or    word [es:di+35], 4 ; byte [es:di+BDS.flags]
15438                                                                   ; return_fake_bpb
15439                                           ; 24/12/2023
15440                                           ; cx = 53 or 31
15441                                           ;mov    cx, 25          ; A_BPB.size - 6
15442                                                                   ; move 'smaller' bpb
15443 00000FFC 8D7D06                           lea     di, [di+6]      ; [es:di+BDS.BPB] = [es:di+BDS.bytespersec]
15444                                   CopyTheBpb:
15445 00000FFF 8D7707                           lea     si, [bx+7]      ; [bx+A_DEVICEPARAMETERS.DP_BPB]
15446 00001002 F3A4                             rep movsb
15447 00001004 1E                               push    ds              ; Save packet segment
15448                                           ; 17/10/2022
15449 00001005 2E8E1E[3000]                     mov     ds, [cs:BIOSDATAWORD]
15450                                           ;mov    ds, word ptr cs:BIOSDATAWORD ; [cs:Bios_Data_Word]
15451                                                                   ; Setup for ds -> Bios_Data
15452 0000100A E8DD03                           call    RestoreOldDpt   ; Restore the old Dpt from TempDpt
15453 0000100D 1F                               pop     ds              ; Restore packet segment
15454 0000100E 5F                               pop     di              ; Restore bds pointer
15455                                   setTrackTable:
15456                                           ; 24/12/2023
15457                                           ;mov    cx, [bx+38]     ; [bx+26h]
15458                                           ;;;
15459 0000100F 8B4F5C                           mov     cx, [bx+5Ch]    ; [bx+A_DEVICEPARAMETERS.DP_TRACKTABLEENTRIES]
15460                                                                   ; offset 85+7 (A_BPB.size+7) (FAT32)
15461 00001012 2E803E[CB0E]00                   cmp     byte [cs:new_genioctl], 0
15462 00001018 7503                             jnz     short sdp_3     ; new type (FAT32) structure
15463 0000101A 8B4F26                           mov     cx, [bx+26h]    ; [bx+A_DEVICEPARAMETERS.DP_TRACKTABLEENTRIES]
15464                                                                   ; offset 31+7 (A_BPB.size+7)
15465                                   sdp_3:
15466                                           ;;;
15467
15468 0000101D 1E                               push    ds
15469 0000101E 2E8E1E[3000]                     mov     ds, [cs:BIOSDATAWORD]
15470 00001023 890E[AA04]                       mov     [sectorspertrack], cx
15471 00001027 1F                               pop     ds
15472
15473                                           ; 24/12/2023
15474 00001028 2680653FF7                       and     byte [es:di+3Fh], 0F7h
15475                                           ; 10/12/2022
15476                                           ;and    byte [es:di+35], 0F7h
15477                                           ;;and   word [es:di+35], 0FFF7h ; [es:di+BDS.flags]
15478                                                                   ; ~good_tracklayout
15479 0000102D F60704                           test    byte [bx], 4    ; [bx+A_DEVICEPARAMETERS.DP_SPECIALFUNCTIONS]
15480                                                                   ; TRACKLAYOUT_IS_GOOD
15481 00001030 7405                             jz      short UglyTrackLayOut
15482                                           ; 24/12/2023
15483 00001032 26804D3F08                       or      byte [es:di+3Fh], 8
15484                                           ; 10/12/2022
15485                                           ;or     byte [es:di+35], 8
15486                                           ;;or    word [es:di+35], 8 ; [es:di+BDS.flags]
15487                                                                   ; good_tracklayout
15488                                   UglyTrackLayOut:
15489 00001037 83F93F                           cmp     cx, 63          ; MAX_SECTORS_IN_TRACK
15490 0000103A 772D                             ja      short TooManyPerTrack
15491                                           ;jcxz   short SectorInfoSaved
15492 0000103C E329                             jcxz    SectorInfoSaved       ; 19/10/2022
15493
15494 0000103E BF[AC04]                         mov     di, tracktable
15495
15496                                           ; 24/12/2023
15497                                           ;lea    si, [bx+40]     ; [bx+A_DEVICEPARAMETERS.DP_SECTORTABLE]
15498                                           ;;;
15499 00001041 8D775E                           lea     si, [bx+5Eh]    ; [bx+A_DEVICEPARAMETERS.DP_SECTORTABLE]
15500                                                                   ; offset 85+9 (A_BPB.size+9) (FAT32)
15501 00001044 2E803E[CB0E]00                   cmp     byte [cs:new_genioctl], 0
15502 0000104A 7503                             jnz     short sdp_4     ; new type (FAT32) structure
15503 0000104C 8D7728                           lea     si, [bx+28h]    ; [bx+A_DEVICEPARAMETERS.DP_SECTORTABLE]
15504                                                                   ; offset 31+9 (A_BPB.size+9)
15505                                   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
```

```
15630                                   NoSetDasd:
15631 000010BB E8B9F8                             call    checksingle     ; Do any needed       diskette swapping
15632 000010BE 89D0                               mov     ax, dx          ; Get track from packet
15633 000010C0 A3[3901]                           mov     [trknum], ax
15634 000010C3 880E[3801]                         mov     [hdnum], cl     ; Store head from packet
15635 000010C7 88CC                               mov     ah, cl
15636 000010C9 BB[AC04]                           mov     bx, tracktable
15637 000010CC 8B0E[AA04]                         mov     cx, [sectorspertrack]
15638                                             ; 24/12/2023 - Retro DOS 5.0
15639 000010D0 E307                               jcxz    set_fmt_retry_count
15640                                   StoreCylinderHead:
15641 000010D2 8907                               mov     [bx], ax        ; Store into TrackTable
15642 000010D4 83C304                             add     bx, 4           ; Skip to next sector field
15643 000010D7 E2F9                               loop    StoreCylinderHead
15644                                   set_fmt_retry_count:      ; 24/12/2023
15645                                             ;mov    cx, 5           ; MAXERR - Set up retry       count
15646                                             ; 02/09/2023
15647 000010D9 B105                               mov     cl, 5
15648                                   FormatRetry:
15649 000010DB 51                                 push    cx
15650 000010DC BB[AC04]                           mov     bx, tracktable
15651 000010DF A0[AA04]                           mov     al, [sectorspertrack]
15652 000010E2 B405                               mov     ah, 5           ; romformat
15653 000010E4 8C1E[A804]                         mov     [xfer_seg], ds
15654 000010E8 E86602                             call    ToRom
15655 000010EB 59                                 pop     cx
15656 000010EC 7216                               jb      short FormatError
15657 000010EE 51                                 push    cx              ; Now verify the sectors just formatted.
15658                                                                     ; NOTE: because of bug in some BIOSes we have to
15659                                                                     ;       set ES:BX to 00:00
15660 000010EF 53                                 push    bx
15661 000010F0 31DB                               xor     bx, bx
15662 000010F2 891E[A804]                         mov     [xfer_seg], bx
15663 000010F6 A0[AA04]                           mov     al, [sectorspertrack]
15664 000010F9 B404                               mov     ah, 4           ; romverify
15665 000010FB B101                               mov     cl, 1
15666 000010FD E85102                             call    ToRom
15667 00001100 5B                                 pop     bx
15668 00001101 59                                 pop     cx
15669 00001102 7329                               jnb     short FormatOk
15670                                   FormatError:
15671 00001104 E83402                             call    ResetDisk
15672 00001107 C606[AA05]01                       mov     byte [had_format_error], 1
15673 0000110C 50                                 push    ax
15674 0000110D 51                                 push    cx
15675 0000110E 52                                 push    dx
15676 0000110F E89501                             call    SetMediaForFormat
15677 00001112 3C01                               cmp     al, 1
15678 00001114 7503                               jnz     short WhileErr
15679 00001116 E83101                             call    SetDasd
15680                                   WhileErr:
15681 00001119 5A                                 pop     dx
15682 0000111A 59                                 pop     cx
15683 0000111B 58                                 pop     ax
15684 0000111C E2BD                               loop    FormatRetry
15685                                   FormatFailed:
15686 0000111E C606[AA05]01                       mov     byte [had_format_error], 1
15687                                                                   ; Set the format error flag
15688 00001123 80FC06                             cmp     ah, 6           ; DSK_CHANGELINE_ERR - convert change line
15689 00001126 7502                               jnz     short DoMapIt   ; Error to time out error
15690 00001128 B480                               mov     ah, 80h         ; DSK_TIMEOUT_ERR
15691                                   DoMapIt:
15692 0000112A E97CFC                             jmp     maperror
15693                                   ; ---------------------------------------------------------------------------
15694
15695                                   FormatOk:
15696 0000112D C606[AA05]00                       mov     byte [had_format_error], 0 ; reset the format error flag
15697 00001132 C3                                 retn
15698                                   ; ---------------------------------------------------------------------------
15699
15700                                   ; 16/10/2022
15701
15702                                   ; ==========================================================================
15703                                   ;
15704                                   ; VerifyTrack:
15705                                   ;
15706                                   ; input: ES:di points to bds for drive
15707                                   ; ==========================================================================
15708
15709                                             ; 24/12/2023 - Retro DOS 5.0
15710                                   VerifyTrack:
15711 00001133 1E                                 push    ds
15712 00001134 C51E[1200]                         lds     bx, [ptrsav]    ; DS:BX points to request header.
15713 00001138 C55F13                             lds     bx, [bx+19]     ; [bx+IOCTL_REQ.GENERICIOCTL_PACKET]
15714
15715                                             ; Come here with DS:[BX] -> packet, ES:[DI] -> bds
15716
15717 0000113B 8B4F03                             mov     cx, [bx+3]      ; [bx+A_VERIFYPACKET.VP_CYLINDER]
15718 0000113E 8B4701                             mov     ax, [bx+1]      ; [bx+A_VERIFYPACKET.VP_HEAD]
15719 00001141 8B5705                             mov     dx, [bx+5]      ; [bx+A_FORMATPACKET.FP_TRACKCOUNT]
15720 00001144 8A1F                               mov     bl, [bx]        ; [bx+A_FORMATPACKET.FP_SPECIALFUNCTIONS]
15721                                                                     ; Get option flag word
15722 00001146 1F                                 pop     ds
15723 00001147 C606[2001]04                       mov     byte [rflag], 4       ; romverify
15724 0000114C 890E[3301]                         mov     [curtrk], cx
15725 00001150 A2[3201]                           mov     [curhd], al     ; ASSUME heads < 256
15726 00001153 8B0E[AA04]                         mov     cx, [sectorspertrack]
15727
15728                                   ; Check specialfunctions to see if DO_FAST_FORMAT has been
15729                                   ; specified if not we should go to the normal track verification
15730                                   ; routine. If fast format has been specified we should get the
15731                                   ; number of tracks to be verified and check it to see if it is
15732                                   ; > 255. If it is then it is an error and we should go to
15733                                   ; VerifyTrack_Err. If not multiply the number of tracks by the
15734                                   ; sectors per track to get the total number of sectors to be
15735                                   ; verified. This should also be less than equal to 255
15736                                   ; otherwise we go to same error exit. If everything is okay
15737                                   ; we initialise cx to the total sectors. use ax as a temporary
15738                                   ; register.
15739
15740                                                             ; Special function requested?
15741 00001157 F6C302                   test    bl, 2           ; DO_FAST_FORMAT
15742 0000115A 7421                     jz      short NormVerifyTrack
15743 0000115C 89D0                     mov     ax, dx          ; Get ax = number of trks to verify
15744 0000115E 08E4                     or      ah, ah
15745 00001160 752C                     jnz     short VerifyTrack_Err ; #tracks > 255
15746 00001162 F6E1                     mul     cl
15747 00001164 08E4                     or      ah, ah
15748 00001166 7526                     jnz     short VerifyTrack_Err ; #sectors > 255
15749 00001168 89C1                     mov     cx, ax
15750                                   ; 24/12/2023
15751 0000116A 26F6453F01               test    byte [es:di+3Fh], 1 ; PCDOS 7.1 IBMBIO.COM
15752                                   ; 10/12/2022
15753                                   ;test   byte [es:di+35], 1
```

```
                                        ;;test  word [es:di+35], 1 ; [es:di+BDS.flags]
                                                                    ; fnon_removable
0000116F 740C                           jz      short NormVerifyTrack
                                                                    ; Multitrack operation = on?
                                        ; 10/12/2022
                                        ; 19/10/2022
00001171 F606[A004]80                   test    byte [multrk_flag], 80h
                                        ;test   word [multrk_flag], 80h ; MULTI_TRK_ON
                                        ;;test  ds:multrk_flag,       80h ; MULTI_TRK_ON
00001176 7405                           jz      short NormVerifyTrack
00001178 C606[A704]01                   mov     byte [multitrk_format_flag], 1
                                NormVerifyTrack:
0000117D 31C0                           xor     ax, ax          ; 1st sector
0000117F 31DB                           xor     bx, bx
00001181 891E[A804]                     mov     [xfer_seg], bx ; Use 0:0 as the transfer address for verify
00001185 E83F00                         call    TrackIo
00001188 C606[A704]00                   mov     byte [multitrk_format_flag], 0
0000118D C3                             retn
                                ; ---------------------------------------------------------------------------

                                VerifyTrack_Err:
0000118E B401                           mov     ah, 1
00001190 E916FC                         jmp     maperror
                                ; ---------------------------------------------------------------------------

                                ; 16/10/2022

                                ; ===========================================================================
                                ;
                                ; ReadTrack:
                                ;
                                ; input: ES:di points to bds for drive
                                ;
                                ; ===========================================================================

                                ReadTrack:
00001193 C606[2001]02                   mov     byte [rflag], 2       ; romread
00001198 EB05                           jmp     short ReadWriteTrack
                                ; ---------------------------------------------------------------------------

                                WriteTrack:

                                ; ===========================================================================
                                ;
                                ; WriteTrack:
                                ;
                                ; input: ES:di points to bds for drive
                                ;
                                ; ===========================================================================

0000119A C606[2001]03                   mov     byte [rflag], 3       ; romwrite

                                        ; Fall into ReadWriteTrack

                                ; ===========================================================================
                                ;
                                ; readWriteTrack:
                                ;
                                ; input:
                                ;    ES:di points to bds for drive
                                ;    rFlag - 2 for read,3 for write
                                ;
                                ; ===========================================================================

                                ReadWriteTrack:
                                        ; save bds pointer segment so we can use it to access
                                        ; our packet. Notice that this is not the standard register
                                        ; assignment for accessing packets

                                        ; 19/10/2022
0000119F 06                             push    es
000011A0 C41E[1200]                     les     bx, [ptrsav]   ; ES:BX -> to request header
000011A4 26C45F13                       les     bx, [es:bx+19] ; [es:bx+IOCTL_REQ.GENERICIOCTL_PACKET]
000011A8 268B4703                       mov     ax, [es:bx+3]  ; [es:bx+A_TRACKREADWRITEPACKET.TRWP_CYLINDER]
000011AC A3[3301]                       mov     [curtrk], ax
000011AF 268B4701                       mov     ax, [es:bx+1]  ; [es:bx+A_TRACKREADWRITEPACKET.TRWP_HEAD]
000011B3 A2[3201]                       mov     [curhd], al    ; Assume heads < 256!!!
000011B6 268B4705                       mov     ax, [es:bx+5]  ; [es:bx+A_TRACKREADWRITEPACKET.TRWP_FIRSTSECTOR]
000011BA 268B4F07                       mov     cx, [es:bx+7]  ; [es:bx+A_TRACKREADWRITEPACKET.TRWP_SECTORSTOREADWRITE]
000011BE 26C45F09                       les     bx, [es:bx+9]  ; [es:bx+A_TRACKREADWRITEPACKET.TRWP_TRANSFERADDRESS]
                                                                ; Get transfer address

                                        ; we just trashed our packet address, but we no longer care

000011C2 8C06[A804]                     mov     [xfer_seg], es ; Pass transfer       segment
000011C6 07                             pop     es

                                        ; Fall into TrackIo

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

                                ; 16/10/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 IO.SYS)

                                ; ===========================================================================
                                ;
                                ; TrackIo:
                                ;   performs track read/write/verify
                                ;
                                ;  input:
                                ;     rFlag       - 2 = read
                                ;                   3 = write
                                ;                   4 = verify
                                ;     AX   - Index into track table of first sector to io
                                ;     CX   - Number of sectors to io
                                ;     Xfer_Seg:BX - Transfer address
                                ;     ES:DI       - Pointer to bds
                                ;     CurTrk      - Current cylinder
                                ;     CurHd       - Current head
                                ;
                                ; ===========================================================================

                                ; 16/03/2019 - Retro DOS v4.0

                                        ; 24/12/2023 - Retro DOS 5.0

                                        ; 19/10/2022
                                TrackIo:
                                                                ; Procedure `disk' will pop stack to
000011C7 8926[3501]                     mov     [spsav], sp    ; SpSav and return if error
000011CB E8A9F7                         call    checksingle    ; Ensure correct disk is in drv
000011CE 803E[A905]01                   cmp     byte [media_set_for_format], 1
                                                                ; See if we have already set  disk
000011D3 7407                           jz      short Dptalreadyset ; base table
000011D5 50                             push    ax              ; set up tables       and variables for i/o
```

```
15878 000011D6 51                               push    cx
15879 000011D7 E8A0F9                           call    iosetup
15880 000011DA 59                               pop     cx
15881 000011DB 58                               pop     ax
15882                                   Dptalreadyset:                          ; Point si at the table entry of the
15883 000011DC BE[AC04]                         mov     si, tracktable ; first sector to be io'd
15884                                           ; 24/12/2023
15885                                           ;add    ax, ax         ; PCDOS 7.1 IBMBIO.COM
15886                                           ;add    ax, ax
15887 000011DF D1E0                             shl     ax, 1
15888 000011E1 D1E0                             shl     ax, 1
15889 000011E3 01C6                             add     si, ax
15890
15891                                           ; WE WANT:
15892                                           ; CX to be the number of times we have to loop
15893                                           ; DX to be the number of sectors we read on each iteration
15894
15895 000011E5 BA0100                           mov     dx, 1
15896
15897                                           ; 24/12/2023
15898 000011E8 26F6453F08                       test    byte [es:di+3Fh], 8 ; PCDOS 7.1 IBMBIO.COM
15899                                           ; 12/12/2022
15900                                           ;test   byte [es:di+23h], 8
15901                                           ;;test  word [es:di+35], 8 ; [es:di+BDS.flags]
15902                                                                   ; good_tracklayout
15903 000011ED 7402                             jz      short ionextsector
15904
15905 000011EF 87CA                             xchg    dx, cx          ; HEY! We can read all secs in one blow
15906                                   ionextsector:
15907 000011F1 51                               push    cx
15908 000011F2 52                               push    dx
15909 000011F3 46                               inc     si
15910 000011F4 46                               inc     si              ; Skip over the       cylinder and head in
15911                                                                   ; the track table
15912 000011F5 AC                               lodsb                   ; Get sector ID       from track table
15913 000011F6 A2[3101]                         mov     [cursec], al
15914
15915                                           ; assumptions for a fixed disk multi-track disk     i/o
15916                                           ; 1). In the input CX (# of sectors to go) to TrackIo,
15917                                           ;     only CL is valid.
15918                                           ; 2). Sector size should be set      to 512 bytes.
15919                                           ; 3). Good track layout
15920
15921                                           ; 24/12/2023
15922 000011F9 26F6453F01                       test    byte [es:di+3Fh], 1 ; PCDOS 7.1 IBMBIO.COM
15923                                           ; 12/12/2022
15924                                           ;test   byte [es:di+23h], 1
15925                                           ;;test  word [es:di+35], 1 ; [es:di+BDS.flags]
15926                                                                   ; fnon_removable ; Fixed disk?
15927 000011FE 7414                             jz      short IoRemovable ; No
15928
15929                                           ; 12/12/2022
15930 00001200 F606[A004]80                     test    byte [multrk_flag], 80h
15931                                           ;test   word [multrk_flag], 80h ; MULTI_TRK_ON
15932                                                                     ; Allow multi-track operation?
15933 00001205 740D                             jz      short IoRemovable ; No,don't do that.
15934 00001207 8916[2201]                       mov     [seccnt], dx
15935 0000120B 89D0                             mov     ax, dx
15936 0000120D E823FA                           call    Disk
15937 00001210 5A                               pop     dx
15938 00001211 59                               pop     cx
15939 00001212 F8                               clc
15940 00001213 C3                               retn
15941                                   ; ---------------------------------------------------------------------------
15942
15943                                   IoRemovable:
15944 00001214 AC                               lodsb                   ; Get sector size index       from track
15945                                                                   ; table and save it
15946 00001215 50                               push    ax
15947 00001216 56                               push    si
15948 00001217 1E                               push    ds              ; Save Bios_Data
15949 00001218 50                               push    ax
15950 00001219 8A26[2C01]                       mov     ah, [eot]       ; Preserve whatever might be in      ah
15951                                                                   ; Fetch EOT while ds-> Bios_Data
15952 0000121D C536[2D01]                       lds     si, [dpt]
15953 00001221 884403                           mov     [si+3], al      ; [si+DISK_PARMS.DISK_SECTOR_SIZ]
15954 00001224 886404                           mov     [si+4], ah      ; [si+DISK_PARMS.DISK_EOT]
15955 00001227 58                               pop     ax
15956 00001228 1F                               pop     ds
15957 00001229 88D0                             mov     al, dl
15958 0000122B A3[2201]                         mov     [seccnt], ax
15959 0000122E E802FA                           call    Disk
15960 00001231 5E                               pop     si              ; Advance buffer pointer by adding
15961                                                                   ; sector size
15962                                           ;pop    ax
15963                                           ; 24/12/2023
15964 00001232 59                               pop     cx
15965
15966                                           ; 02/09/2023 (PCDOS 7.1)
15967                                           ;call   SectorSizeIndexToSectorSize
15968                                           ;mov    cl, al  ; 24/12/2023
15969 00001233 B88000                           mov     ax, 128
15970 00001236 D3E0                             shl     ax, cl
15971
15972 00001238 01C3                             add     bx, ax
15973 0000123A 5A                               pop     dx
15974 0000123B 59                               pop     cx
15975 0000123C E2B3                             loop    ionextsector
15976 0000123E 803E[A905]01                     cmp     byte [media_set_for_format], 1
15977                                           ;jz     short NoNeedDone
15978                                           ; 12/12/2022
15979 00001243 7404                             je      short NoNeedDone2
15980 00001245 E877F9                           call    done            ; set time of last access, and reset
15981                                                                   ; entries in Dpt.
15982                                   NoNeedDone:
15983 00001248 F8                               clc     ; not necessary ('done' clears cf) ; 24/12/2023
15984                                   NoNeedDone2:
15985 00001249 C3                               retn
15986
15987                                   ; =============== S U B   R O U T I N E =======================================
15988
15989                                   ; ---------------------------------------------------------------------------
15990                                   ;
15991                                   ; The sector size in bytes needs to be converted to an index value for the ibm
15992                                   ; rom. (0=>128,1=>256,2=>512,3=>1024). It is assumed that only these values
15993                                   ; are permissible.
15994                                   ;
15995                                   ; On Input   AX contains sector size in bytes
15996                                   ; On Output  AL Contains index
15997                                   ; All other registers preserved
15998                                   ;
15999                                   ; ---------------------------------------------------------------------------
16000
16001                                   ; 02/09/2023
```

```
16002                                   ;SectSizeToSectIndex:
16003                                   ;           cmp     ah, 2           ; (0=>128,1=>256,2=>512,3=>1024)
16004                                   ;                                   ; examine upper       byte only
16005                                   ;           ja      short OneK
16006                                   ;           mov     al, ah          ; value in AH is the index!
16007                                   ;           retn
16008                                   
16009                                   ; ---------------------------------------------------------------------------
16010                                   ;
16011                                   ;OneK:
16012                                   ;           mov     al, 3
16013                                   ;           retn
16014                                   
16015                                   ; =============== S U B   R O U TI N E =======================================
16016                                   
16017                                   ; 02/09/2023
16018                                   ;SectorSizeIndexToSectorSize:
16019                                   ;           mov     cl, al
16020                                   ;           mov     ax, 128
16021                                   ;           shl     ax, cl
16022                                   ;           retn
16023                                   
16024                                   ; =============== S U B   R O U TI N E =======================================
16025                                   
16026                                   ; 16/10/2022
16027                                   
16028                                   ; ---------------------------------------------------------------------------
16029                                   ;
16030                                   ; SetDASD
16031                                   ;
16032                                   ; Set up the rom for formatting.
16033                                   ; we have to tell the rom bios what type of disk is in the drive.
16034                                   ;
16035                                   ; On Input   - ES:di - Points to bds
16036                                   ;
16037                                   ; ---------------------------------------------------------------------------
16038                                   
16039                                               ; 24/12/2023 - Retro DOS 5.0
16040                                               ; (PCDOS 7.1 IBMBIO.COM - BIOSCODE:129Bh)
16041                                   
16042                                               ; 19/10/2022
16043                                   SetDasd:
16044 0000124A 803E[AA05]01                         cmp     byte [had_format_error], 1 ;
16045                                                                           ; See if we've previously set dasd type
16046 0000124F 740C                                 jz      short DoSetDasd
16047                                               ; 24/12/2023
16048 00001251 26F6453F80                           test    byte [es:di+3Fh], 80h
16049                                               ; 10/12/2022
16050                                               ;test   byte [es:di+23h], 80h
16051                                               ;;test  word [es:di+23h], 80h ; [es:di+BDS.flags]
16052                                                                           ; set_dasd_true
16053 00001256 7446                                 jz      short DasdHasBeenSet
16054                                               ; 24/12/2023
16055 00001258 2680653F7F                           and     byte [es:di+3Fh], 7Fh
16056                                               ; 10/12/2022
16057                                               ;and    byte [es:di+23h], 7Fh
16058                                               ;;and   word [es:di+23h], 0FF7Fh ; [es:di+BDS.flags]
16059                                                                           ; ~set_dasd_true
16060                                   DoSetDasd:
16061 0000125D C606[AA05]00                         mov     byte [had_format_error], 0 ; Reset it
16062 00001262 C606[3B01]50                         mov     byte [gap_patch], 50h ; Format gap for 48tpi disks
16063 00001267 B004                                 mov     al, 4
16064                                               ; 24/12/2023
16065 00001269 268A653E                             mov     ah, [es:di+3Eh]
16066                                               ; 02/09/2023
16067                                               ;mov    ah, [es:di+22h] ; [es:di+BDS.formfactor]
16068 0000126D 80FC02                               cmp     ah, 2
16069                                               ;cmp    byte [es:di+22h], 2 ; [es:di+BDS.formfactor]
16070                                                                           ; DEV_3INCH720KB
16071 00001270 7414                                 jz      short DoSet
16072                                               ; 24/12/2023
16073 00001272 B001                                 mov     al, 1
16074                                               ;cmp    ah, 1
16075 00001274 38C4                                 cmp     ah, al  ; 1
16076                                               ;cmp    byte [es:di+22h], 1 ; [es:di+BDS.formfactor]
16077                                                                           ; DEV_5INCH96TPI
16078                                               ;jz     short GotBig
16079                                               ; 24/12/2023
16080                                               ;mov    al, 1
16081                                               ;jmp    short DoSet
16082                                               ; 02/09/2023
16083 00001276 750E                                 jnz     short DoSet
16084                                   GotBig:
16085                                               ;mov    al, 2           ; 160/320k in a       1.2 meg drive
16086                                               ; 02/09/2023
16087 00001278 40                                   inc     ax  ; mov al, 2
16088 00001279 803E[A805]00                         cmp     byte [mediatype], 0
16089 0000127E 7506                                 jnz     short DoSet
16090                                               ;mov    al, 3           ; 1.2meg in a 1.2meg drive
16091                                               ; 10/12/2022
16092                                               ;inc    al  ; al = 3
16093                                               ; 18/12/2022
16094 00001280 40                                   inc     ax  ; al = 3
16095 00001281 C606[3B01]54                         mov     byte [gap_patch], 54h
16096                                   DoSet:
16097 00001286 1E                                   push    ds
16098 00001287 56                                   push    si
16099                                   
16100                                               ;mov    ds, [zeroseg]  ; Point to interrupt vectors
16101                                               ; 02/09/2023
16102 00001288 31F6                                 xor     si, si
16103 0000128A 8EDE                                 mov     ds, si  ; 0
16104                                   
16105 0000128C C5367800                             lds     si, [DSKADR]
16106                                               ;lds    si, [78h]       ; [DSKADR]  (Int 1Eh)
16107                                               ;;lds   si, ds:78h
16108                                   
16109 00001290 C644090F                             mov     byte [si+9], 0Fh ;
16110                                                                           ; [si+DISK_PARMS.DISK_HEAD_STTL]
16111 00001294 5E                                   pop     si
16112 00001295 1F                                   pop     ds
16113 00001296 B417                                 mov     ah, 17h
16114 00001298 268A5504                             mov     dl, [es:di+4]
16115 0000129C CD13                                 int     13h             ; DISK - DISK -       SET TYPE (AT,XT2,XT286,CONV,PS
16116                                                                           ; AL = disk type AL = 03h - high-capacity disk in high-capacity   drive
16117                                   DasdHasBeenSet:
16118 0000129E 268A6513                             mov     ah, [es:di+13h]       ; [es:di+BDS.secpertrack]
16119 000012A2 8826[3701]                           mov     [formt_eot], ah
16120 000012A6 C3                                   retn
16121                                   
16122                                   ; =============== S U B   R O U TI N E =======================================
16123                                   
16124                                   ; 16/10/2022
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 86CD                           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
```

```
16249                                   ;dec    al
16250                                   ; 18/12/2022
16251 00001332 48                       dec     ax
16252                                   ; al = 2
16253                                   ;mov    al, 1           ; Function not supported.
16254                                   ;jmp    short PopStatRet
16255                           ; ---------------------------------------------------------------------------
16256
16257                           FormatStatIllegalComb:
16258                                   ; 10/12/2022
16259                                   ;dec    al      ; 3 -> 2 or 2 -> 1
16260                                   ; 18/12/2022
16261 00001333 48                       dec     ax
16262                                   ; al = 2
16263                                   ;mov    al, 2           ; Function supported, but
16264                                                           ; Illegal sect/trk,trk combination.
16265                                   ; 10/12/2022
16266                                   ;jmp    short PopStatRet
16267                           ; ---------------------------------------------------------------------------
16268
16269                           FormatStatTimeOut:
16270                                   ; 10/12/2022
16271                                   ; al = 3
16272                                   ;mov    al, 3           ; Function supported, but
16273                                                           ; Media not present.
16274                           PopStatRet:
16275 00001334 5F                       pop     di
16276 00001335 5E                       pop     si
16277 00001336 1F                       pop     ds
16278 00001337 07                       pop     es
16279                           SetMediaRet:
16280 00001338 5A                       pop     dx
16281 00001339 59                       pop     cx
16282 0000133A C3                       retn
16283
16284                           ; =============== S U B   R O U T I N E =======================================
16285
16286                           ; 16/10/2022
16287
16288                           ; ---------------------------------------------------------------------------
16289                           ;
16290                           ; RESET THE DRIVE
16291                           ;
16292                           ; we also set [Step_Drv] to -1 to force the main disk routine to use the
16293                           ; slow head settle time for the next operation. this is because the reset
16294                           ; operation moves the head to cylinder 0,so we need to do a seek the next
16295                           ; time around - there is a problem with 3.5" drives in that the head does
16296                           ; not settle down in time,even for read operations!!
16297                           ;
16298                           ; ---------------------------------------------------------------------------
16299
16300                           ResetDisk:
16301 0000133B 50                         push    ax
16302
16303                                     ; 02/09/2023
16304 0000133C B80100                     mov     ax, 1 ; PCDOS 7.1
16305 0000133F 3806[A905]                 cmp     [media_set_for_format], al ; 1
16306                                     ;cmp    byte [media_set_for_format], 1
16307                                                             ; Reset while formatting?
16308 00001343 7503                       jnz     short ResetDisk_cont
16309                                                             ; Then verify operation in "fmt & vrfy"
16310                                     ;mov    byte [had_format_error], 1 ; Might have failed.
16311 00001345 A2[AA05]                   mov     [had_format_error], al ; 1
16312                           ResetDisk_cont:
16313                                     ; 02/09/2023 (ah=0)
16314                                     ;xor    ah, ah          ; So signals that we had a format error
16315 00001348 CD13                       int     13h             ; DISK - RESET DISK SYSTEM
16316                                                             ; DL = drive (if bit 7 is set both hard     disks and floppy disks reset)
16317 0000134A C606[7600]FF               mov     byte [step_drv], 0FFh ; -1
16318                                                             ; Zap up the speed
16319 0000134F 58                         pop     ax
16320 00001350 C3                         retn
16321
16322                           ; =============== S U B   R O U T I N E =======================================
16323
16324                           ; 16/10/2022
16325
16326                           ; ---------------------------------------------------------------------------
16327                           ;
16328                           ; This routine sets up the drive parameter table with the values needed for
16329                           ; format,does an int 13. values in Dpt are restored after a verify is done.
16330                           ;
16331                           ; on entry  -      ES:DI - points to bds for the drive
16332                           ;          Xfer_Seg:BX - points to trkbuf
16333                           ;          AL     - number of sectors
16334                           ;          AH     - int 13 function code
16335                           ;          CL     - sector number for verify
16336                           ;          DS     - Bios_Data
16337                           ;
16338                           ; ON EXIT   -      DS,DI,ES,BX remain unchanged.
16339                           ;          AX and flags are the results of the int 13
16340                           ;
16341                           ; ---------------------------------------------------------------------------
16342
16343                                     ; 24/12/2023 - Retro DOS 5.0
16344
16345                                     ; 19/10/2022
16346                           ToRom:
16347 00001351 53                         push    bx
16348 00001352 56                         push    si
16349
16350                                     ; Compaq bug fix - check whether we are using new ROM
16351                                     ; functionality to set up format, not merely if it exists.
16352                                     ; This was formerly a check against [new_rom]
16353
16354 00001353 F606[A905]01               test    byte [media_set_for_format], 1
16355 00001358 7534                       jnz     short GotValidDpt
16356 0000135A 50                         push    ax
16357 0000135B 06                         push    es              ; Save bds segment
16358                                     ; 24/12/2023
16359 0000135C 26807D3E02                 cmp     byte [es:di+3Eh], 2
16360                                     ;cmp    byte [es:di+22h], 2 ; [es:di+BDS.formfactor]
16361                                                             ; ffSmall ; is it a 3.5" drive?
16362                                     ; 24/12/2023
16363                                     ;pushf  ; not necessary       ; (Save the cmp result)
16364 00001361 8E06[1A00]                 mov     es, [zeroseg]
16365                                     ;les    si, es:78h      ; Get pointer to disk base table
16366 00001365 26C4367800                 les     si, [es:DSKADR]
16367                                     ;mov    word ptr ds:dpt, si
16368                                     ;mov    word ptr ds:dpt+2, es ;        Save pointer to table
16369 0000136A 8936[2D01]                 mov     [dpt], si
16370 0000136E 8C06[2F01]                 mov     [dpt+2], es     ; Save pointer to table
16371
16372 00001372 A0[3701]                   mov     al, [formt_eot]
```

```
16373 00001375 26884404                         mov     [es:si+4], al  ; [es:si+DISK_PARMS.DISK_EOT]
16374 00001379 A0[3B01]                         mov     al, [gap_patch]
16375 0000137C 26884407                         mov     [es:si+7], al  ; [es:si+DISK_PARMS.DISK_FORMT_GAP]
16376                                                                  ; Important for       format
16377 00001380 26C644090F                       mov     byte [es:si+9], 0Fh ; [es:si+DISK_PARMS.DISK_HEAD_STTL]
16378                                                                  ; Assume we are       doing a seek operation
16379                                                                  ; Setup motor start correctly for 3.5" drives
16380                                           ; 24/12/2023
16381                                           ;popf                  ; Get result of       earlier cmp
16382 00001385 7505                             jnz     short MotorStrtOK
16383 00001387 26C6440A04                       mov     byte [es:si+0Ah], 4 ; [es:si+DISK_PARMS.DISK_MOTOR_STRT]
16384                                   MotorStrtOK:
16385 0000138C 07                               pop     es              ; Restore bds segment
16386 0000138D 58                               pop     ax
16387                                   GotValidDpt:
16388 0000138E 8B16[3901]                       mov     dx, [trknum]    ; Set track number
16389 00001392 88D5                             mov     ch, dl          ; Set low 8 bits in ch
16390 00001394 268A5504                         mov     dl, [es:di+4]   ; Set drive number
16391 00001398 8A36[3801]                       mov     dh, [hdnum]     ; Set head number
16392 0000139C 06                               push    es              ; Save bds segment
16393 0000139D 8E06[A804]                       mov     es, [xfer_seg]
16394 000013A1 CD13                             int     13h             ; DISK -
16395 000013A3 07                               pop     es              ; Restore bds segment
16396 000013A4 5E                               pop     si
16397 000013A5 5B                               pop     bx
16398 000013A6 C3                               retn
16399
16400                                   ; ---------------------------------------------------------------------------
16401
16402                                   ; 16/10/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 IO.SYS)
16403                                   ; 24/12/2023 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMBIO.COM)
16404
16405                                   ; BIOSCODE:1124h (MSDOS 6.21, IO.SYS)
16406                                   ; PCDOS 7.1 IBMBIO.COM - BIOSCODE:1404h
16407
16408                                   ; ============================================================================
16409                                   ;
16410                                   ; get the owner of the physical drive represented by the logical drive in al.
16411                                   ; the assumption is that we **always** keep track of the owner of a drive!!
16412                                   ; if this is not the case, the system may hang, just following the linked list.
16413                                   ;
16414                                   ; ============================================================================
16415
16416                                           ; 24/12/2023 - Retro DOS 5.0
16417
16418                                           ; 19/10/2022
16419                                   ioctl_getown:
16420 000013A7 E8FAF1                           call    SetDrive
16421 000013AA 268A4504                         mov     al, [es:di+4]  ; [es:di+BDS.drivenum]
16422                                                                  ; Get physical drive number
16423 000013AE C43E[1901]                       les     di, [start_bds] ; Get start of bds chain
16424                                   ownloop:
16425 000013B2 26384504                         cmp     [es:di+4], al  ; [es:di+BDS.drivenum]
16426 000013B6 7507                             jnz     short getnextBDS
16427                                           ; 24/12/2023
16428 000013B8 26F6453F20                       test    byte [es:di+3Fh], 20h ; (PCDOS 7.1 IBMBIO.COM)
16429                                           ; 10/12/2022
16430                                           ;test  byte [es:di+23h], 20h
16431                                           ;;test  word [es:di+23h], 20h ; [es:di+BDS.flags]
16432                                                                  ; fi_own_physical
16433 000013BD 7514                             jnz     short exitown
16434                                   getnextBDS:
16435 000013BF 26C43D                           les     di, [es:di]    ; [es:di+BDS.link]
16436 000013C2 EBEE                             jmp     short ownloop
16437                                   ; ---------------------------------------------------------------------------
16438
16439                                   ; ============================================================================
16440                                   ;
16441                                   ; set the ownership of the physical drive represented by the logical drive
16442                                   ; in al to al.
16443                                   ;
16444                                   ; ============================================================================
16445
16446                                           ; 24/12/2023 - Retro DOS 5.0
16447
16448                                           ; 19/10/2022
16449                                   ioctl_setown:
16450 000013C4 E8DDF1                           call    SetDrive
16451 000013C7 C606[7A00]01                     mov     byte [fsetowner], 1
16452                                                                  ; set flag for CheckSingle to look at.
16453 000013CC E8A8F5                           call    checksingle
16454                                           ; 02/09/2023
16455 000013CF FE0E[7A00]                       dec     byte [fsetowner] ; 0
16456                                           ;mov    byte [fsetowner], 0
16457                                                                  ; set ownership       of drive reset flag
16458                                           ; Fall into ExitOwn
16459
16460                                   ; ============================================================================
16461                                   ;
16462                                   ; if there is only one logical drive assigned to this physical drive, return
16463                                   ; 0 to user to indicate this. Enter with ES:di -> the owner's bds.
16464                                   ;
16465                                   ; ============================================================================
16466
16467                                           ; 24/12/2023 - Retro DOS 5.0
16468                                   exitown:
16469 000013D3 30C9                             xor     cl, cl
16470                                           ; 24/12/2023
16471 000013D5 26F6453F10                       test    byte [es:di+3Fh], 10h ; (PCDOS 7.1 IBMBIO.COM)
16472                                           ; 12/12/2022
16473                                           ;test  byte [es:di+23h], 10h
16474                                           ;;test  word [es:di+23h], 10h ; [es:di+BDS.flags]
16475                                                                  ; fi_am_mult
16476 000013DA 7406                             jz      short exitnomult
16477 000013DC 268A4D05                         mov     cl, [es:di+5]  ; [es:di+BDS.drivelet]
16478                                                                  ; Get logical drive number
16479                                                                  ; Get it 1-based
16480 000013E0 FEC1                             inc     cl
16481                                   exitnomult:
16482 000013E2 C51E[1200]                       lds     bx, [ptrsav]
16483 000013E6 884F01                           mov     [bx+1], cl     ; [bx+unit]
16484                                                                  ; Exit normal termination
16485                                           ; 12/12/2022
16486                                           ; cf=0
16487                                           ;clc
16488 000013E9 C3                               retn
16489
16490                                   ; =============== S U B   R O U TI N E =======================================
16491
16492                                   ; 16/10/2022
16493
16494                                   ; ---------------------------------------------------------------------------
16495                                   ;
16496                                   ; moves the old Dpt that had been saved in TempDpt back to Dpt. this is done
```

```
                                ; only if the first byte of TempDpt is not -1.
                                ; all registers (including flags) are preserved.
                                ;
                                ; ---------------------------------------------------------------------------

                                        ; 24/12/2023
                                        ; 19/10/2022
                                RestoreOldDpt:
                                        ; if we have already restored the disk base table earlier,
                                        ; do not do it again.

000013EA 50                             push    ax
000013EB 30C0                           xor     al, al
000013ED A2[AA05]                       mov     [had_format_error], al; Reset flag and
000013F0 8606[A905]                     xchg    al, [media_set_for_format] ; get current flag setting
000013F4 08C0                           or      al, al
000013F6 7418                           jz      short DontRestore
000013F8 56                             push    si
000013F9 1E                             push    ds
000013FA 06                             push    es
000013FB C536[AB05]                     lds     si, [tempdpt]

                                        ; 17/10/2022
                                        ;mov    es, [cs:BIOSDATAWORD]
                                        ;;mov   es, word ptr cs:BIOSDATAWORD ; [cs:Bios_Data_Word]
                                        ;mov    es, [es:zeroseg]
                                        ;;mov   es, es:zeroseg ; CAS -- bleeeech!

                                        ; 24/12/2023
000013FF 31C0                           xor     ax, ax
00001401 8EC0                           mov     es, ax ; 0

                                        ;mov    es:78h, si      ; [es:DSKADR]  (Int 1Eh)
00001403 2689367800                     mov     [es:DSKADR], si
                                        ;mov    word ptr es:7Ah, ds ; [es:DSKADR+2]
00001408 268C1E7A00                     mov     [es:DSKADR+2], ds
0000140D 07                             pop     es
0000140E 1F                             pop     ds
0000140F 5E                             pop     si
                                DontRestore:
00001410 58                             pop     ax
                                        ; 12/12/2022
                                        ; cf=0
                                        ;clc                    ;  Clear carry
00001411 C3                             retn

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

                                ; 16/10/2022

                                ; ============================================================================
                                ;   get media id
                                ; ============================================================================
                                ;
                                ; FUNCTION: get the volume label,the system id and the serial number from
                                ;       the media that has the extended boot record.
                                ;       for the conventional media,this routine will return "unknown
                                ;       media type" error to dos.
                                ;
                                ; INPUT :   ES:di -> bds table for this drive.
                                ;
                                ; OUTPUT:   the request packet filled with the information,if not carry.
                                ;       if carry set,then al contains the device driver error number
                                ;       that will be returned to dos.
                                ;       register DS,DX,AX,CX,DI,SI destroyed.
                                ;
                                ; SUBROUTINES TO BE CALLED:
                                ;   BootIo:NEAR
                                ;
                                ; LOGIC:
                                ;   to recognize the extended boot record,this logic will actually
                                ;   access the boot sector even if it is a hard disk.
                                ;   note:the valid extended bpb is recognized by looking at the mediabyte
                                ;   field of bpb and the extended boot signature.
                                ;
                                ; {
                                ;   get logical drive number from bds table;
                                ;   rFlag = read operation;
                                ;   BootIo;         /*get the media boot record into the buffer
                                ;   if (no error) then
                                ;        if (extended boot record) then
                                ;          { set volume label,volume serial number and system id
                                ;            of the request packet to those of the boot record;
                                ;          };
                                ;        else               /*not an extended bpb */
                                ;          { set register al to "unknown media.." error code;
                                ;            set carry bit;
                                ;          };
                                ;   else
                                ;        ret;      /*already error code is set in the register al
                                ;
                                ; ============================================================================

                                ;size_of_EXT_BOOT_SERIAL equ 4
                                ;;size_of_EXT_BOOT_VOL_LABEL equ 11
                                ;;size_of_EXT_SYSTEM_ID equ 8

                                        ; 24/12/2023 - Retro DOS 5.0
                                        ; (PCDOS 7.1 IBMBIO.COM - BIOSCODE:1478h)

                                        ; 19/10/2022
                                GetMediaId:
00001412 E8B000                         call    ChangeLineChk
00001415 268A4505                       mov     al, [es:di+5]  ; [es:di+BDS.drivelet] ; Logical drive number
00001419 C606[2001]02                   mov     byte [rflag], 2         ; Read operation
0000141E E88C00                         call    BootIo          ; Read boot sector into       DiskSector
00001421 722E                           jb      short IOCtl_If1
                                                                ; Valid? (0F0h-0FFh?)
00001423 803E[6701]F0                   cmp     byte [disksector+15h], 0F0h
                                                                ; [disksector+EXT_BOOT.BPB+EBPB.MEDIADESCRIPTOR]
                                        ;jb     short IOCtl_If2         ; brif not valid (0F0h - 0FFh)
                                        ; 24/12/2023
00001428 7225                           jb      short IOCtl_If7

                                        ; 24/12/2023
                                        ; 10/12/2022
                                        ;mov    si, disksector+26h
                                        ;;;
                                        ; 24/12/2023
                                        ;mov    si, disksector+43h ; BS_FAT32_VolID
0000142A BE[9401]                       mov     si, disksector+42h ; BS_FAT32_BootSig ; 24/12/2023
0000142D 833E[6801]00                   cmp     word [disksector+16h], 0 ; BPB.FATSz16
00001432 7403                           jz      short IOCtl_If3 ; FAT32 fs
00001434 83EE1C                         sub     si, 1Ch         ; FAT (12-16) fs ; 43h-1Ch = 27h ; BS_VolID
```

```
                                        ; si = disksector+26h = BS_BootSig ; 24/12/2023
                                IOCtl_If3:
                                        ;cmp    byte [si-1], 29h ; BS_BootSig or BS_FAT32_BootSig
                                        ;;;
00001437 803C29                         cmp     byte [si], 29h
                                        ;cmp    byte [disksector+26h], 29h ; [disksector+EXT_BOOT.SIG]
                                                                ; EXT_BOOT_SIGNATURE
0000143A 7512                           jne     short IOCtl_If2         ; not extended boot record
0000143C C43E[1200]                     les     di, [ptrsav]    ; es:di points to request header
00001440 26C47F13                       les     di, [es:bx+19] ; [es:bx+IOCTL_REQ.GENERICIOCTL_PACKET]
                                        ; 10/12/2022
                                        ;mov    si, disksector+27h ; disksector+EXT_BOOT.SERIAL
00001444 46                             inc     si
                                        ; 24/12/2023
                                        ; si = disksector+27h (BS_VolID)
                                        ;      or disksector+43h (BS_FAT32_VolID)

00001445 83C702                         add     di, 2           ; A_MEDIA_ID_INFO.MI_SERIAL
                                IOCtl_If4:      ; 24/12/2023
00001448 B91700                         mov     cx, 23          ; size_of_EXT_BOOT_SERIAL
                                                                ; L+size_of_EXT_BOOT_VOL_LABEL
                                                                ; +size_of_EXT_SYSTEM_ID
0000144B F3A4                           rep movsb               ; Move from Bios_Data into request packet

                                        ; 10/12/2022
                                        ; cf = 0
                                        ;clc

0000144D C3                             retn
                                ; ---------------------------------------------------------------------------

                                        ; 24/12/2023
                                IOCtl_If2:
0000144E F9                             stc
                                IOCtl_If7:
0000144F B007                           mov     al, 7           ; error_unknown_media
                                        ;stc
                                IOCtl_If6:
                                IOCtl_If1:
00001451 C3                             retn
                                ; ---------------------------------------------------------------------------

                                ; 16/10/2022

                                ; ============================================================================
                                ;  set media id
                                ; ============================================================================

                                ; function: set the volume label, the system id and the serial number of
                                ;       the media that has the extended boot record.
                                ;       for the conventional media, this routine will return "unknown
                                ;       media.." error to dos.
                                ;       this routine will also set the corresponding informations in
                                ;       the bds table.
                                ;
                                ; input :   ES:di -> bds table for this drive.
                                ;
                                ; output:   the extended boot record in the media will be set according to
                                ;       the request packet.
                                ;       if carry set, then al contains the device driver error number
                                ;       that will be returned to dos.
                                ;
                                ; subroutines to be called:
                                ;   BootIo:NEAR
                                ;
                                ; logic:
                                ;
                                ; {
                                ;   get drive_number from bds;
                                ;   rFlag = "read operation";
                                ;   BootIo;
                                ;   if (no error) then
                                ;        if (extended boot record) then
                                ;          { set volume label,volume serial number and system id
                                ;            of the boot record to those of the request packet;
                                ;            rFlag = "write operation";
                                ;            get drive number from bds;
                                ;            BootIo;        /*write it back*/
                                ;          };
                                ;        else               /*not an extended bpb */
                                ;          { set register al to "unknown media.." error code;
                                ;            set carry bit;
                                ;            ret;   /*return back to caller */
                                ;          };
                                ;   else
                                ;        ret;              /*already error code is set */
                                ;
                                ; ============================================================================

                                        ; 24/12/2023 - Retro DOS 5.0

                                        ; 19/10/2022
                                SetMediaId:
00001452 E87000                         call    ChangeLineChk
00001455 268A4505                       mov     al, [es:di+5]  ; [es:di+BDS.drivelet]
                                                                ; Logical drive       number
00001459 88C2                           mov     dl, al
0000145B C606[2001]02                   mov     byte [rflag], 2         ; romread
00001460 52                             push    dx
00001461 E84900                         call    BootIo          ; Read boot sec       to Bios_Data:DiskSector
00001464 5A                             pop     dx
00001465 72EA                           jb      short IOCtl_If6
                                                                ; Valid? (0F0h-0FFh?)
00001467 803E[6701]F0                   cmp     byte [disksector+15h], 0F0h
                                                                ; [disksector+EXT_BOOT.BPB+EBPB.MEDIADESCRIPTOR]
0000146C 72E1                           jb      short IOCtl_If7         ; Brif not

                                        ; 24/12/2023
                                        ;cmp    byte [disksector+26h], 29h ; [disksector+EXT_BOOT.SIG]
                                        ;                               ; EXT_BOOT_SIGNATURE
                                        ;jnz    short IOCtl_If7         ; not extended boot record

0000146E 06                             push    es              ; Save BDS pointer
0000146F 57                             push    di
00001470 1E                             push    ds              ; Point ES To boot record
00001471 07                             pop     es

                                        ; 24/12/2023
                                        ;;;
                                        ;mov    di, disksector+43h ; disksector+EXT_BOOT.SERIAL
00001472 BF[9401]                       mov     di, disksector+42h ; BS_FAT32_BootSig ; 24/12/2023
00001475 833E[6801]00                   cmp     word [disksector+16h], 0 ; BPB.FATSz16
0000147A 7403                           jz       short IOCtl_If5        ; FAT32 fs
0000147C 83EF1C                         sub     di, 1Ch         ; 67-28 ; offset disksektor+27h
```

```
16745                                           ; di = disksector+26h = BS_BootSig ; 24/12/2023
16746                                   IOCtl_If5:
16747                                           ;cmp    byte [di-1], 29h ; BS_BootSig or BS_FAT32_BootSig
16748 0000147F 803D29                            cmp     byte [di], 29h
16749 00001482 7404                              je      short IOCtl_If8
16750 00001484 5F                                pop     di              ; not extended boot record
16751 00001485 07                                pop     es
16752                                           ;jmp    short IOCtl_If7
16753                                           ; 24/12/2023
16754 00001486 EBC6                              jmp     short IOCtl_If2
16755                                   IOCtl_If8:
16756                                           ;;;
16757                                           ; 24/12/2023
16758                                           ;mov    di, disksector+27h ; disksector+EXT_BOOT.SERIAL
16759 00001488 47                                inc     di
16760                                           ; di = disksector+27h (BS_VolID)
16761                                           ;      or disksector+43h (BS_FAT32_VolID)
16762
16763 00001489 C536[1200]                        lds     si, [ptrsav]    ; ds:si points to request header.
16764 0000148D C57413                            lds     si, [si+19]     ; [si+IOCTL_REQ.GENERICIOCTL_PACKET]
16765 00001490 83C602                            add     si, 2           ; A_MEDIA_ID_INFO.MI_SERIAL
16766
16767                                           ; 24/12/2023
16768                                           ;mov    cx, 23          ; size_of_EXT_BOOT_SERIAL
16769                                           ;                       ; +size_of_EXT_BOOT_VOL_LABEL
16770                                           ;                       ; +size_of_EXT_SYSTEM_ID
16771                                           ;rep movsb
16772 00001493 E8B2FF                            call    IOCtl_If4       ; copy volume serial, label and system id
16773
16774 00001496 06                                push    es              ; point ds back to Bios_Data
16775 00001497 1F                                pop     ds
16776 00001498 5F                                pop     di              ; restore bds pointer
16777 00001499 07                                pop     es
16778 0000149A E8B2F3                            call    mov_media_ids   ; update the bds media id info.
16779 0000149D 88D0                              mov     al, dl
16780 0000149F C606[2001]03                      mov     byte [rflag], 3       ; romwrite
16781 000014A4 E80600                            call    BootIo          ; write it back.
16782 000014A7 C606[1E01]FF                      mov     byte [tim_drv], 0FFh
16783                                                                    ; make sure chk_media check the driver
16784                                                                    ; return with error code from BootIo
16785 000014AC C3                                retn
16786                                   ; ---------------------------------------------------------------------------
16787
16788                                           ; 24/12/2023
16789                                   ;IOCtl_If7:
16790                                   ;       mov     al, 7           ; error_unknown_media
16791                                   ;       stc
16792                                   ;IOCtl_If6:
16793                                   ;       retn
16794
16795                                   ; =============== S U B   R O U T I N E =======================================
16796
16797                                   ; 16/10/2022
16798
16799                                   ; ---------------------------------------------------------------------------
16800                                   ;   BootIo
16801                                   ; ---------------------------------------------------------------------------
16802                                   ;
16803                                   ; function: read/write the boot record into boot sector.
16804                                   ;
16805                                   ; input :
16806                                   ;       al=logical drive number
16807                                   ;       rFlag = operation (read/write)
16808                                   ;
16809                                   ; output:   for read operation,the boot record of the drive specified in bds
16810                                   ;       be read into the DiskSector buffer.
16811                                   ;       for write operation,the DiskSector buffer image will be written
16812                                   ;       to the drive specified in bds.
16813                                   ;       if carry set,then al contains the device driver error number
16814                                   ;       that will be returned to dos.
16815                                   ;       AX,CX,DX register destroyed.
16816                                   ;       if carry set,then al will contain the error code from DiskIO.
16817                                   ;
16818                                   ; subroutines to be called:
16819                                   ;   DiskIO:NEAR
16820                                   ;
16821                                   ; logic:
16822                                   ;
16823                                   ; {
16824                                   ;   first_sector = 0;       /*logical sector 0 is the boot sector */
16825                                   ;   sectorcount = 1;        /*read 1 sector only */
16826                                   ;   buffer = DiskSector;    /*read it into the DiskSector buffer */
16827                                   ;   call DiskIO (rFlag,drive_number,first_sector,sectorcount,buffer);
16828                                   ; }
16829                                   ; ===========================================================================
16830
16831                                           ; 19/10/2022
16832                                   BootIo:
16833 000014AD 06                                push    es
16834 000014AE 57                                push    di
16835 000014AF 53                                push    bx
16836 000014B0 1E                                push    ds
16837 000014B1 07                                pop     es              ; Point ES: to Bios_Data
16838
16839                                           ; Call DiskIO to read/write the boot sec. The parameters which
16840                                           ; need to be initialized for this subroutine out here are
16841                                           ; - Transfer address to Bios_Data:DiskSector
16842                                           ; - Low sector needs to be initalized to 0. this is a reg. param
16843                                           ; - Hi sector in [Start_Sec_H] needs to be initialised to 0.
16844                                           ; - Number of sectors <-- 1
16845
16846 000014B2 BF[5201]                          mov     di, disksector ; es:di -> transfer address
16847 000014B5 31D2                              xor     dx, dx          ; First sector (h) -> 0
16848 000014B7 8916[9C04]                        mov     [start_sec_h], dx ; Start sector (h) -> 0
16849 000014BB B90100                            mov     cx, 1
16850 000014BE E850F5                            call    diskio
16851 000014C1 5B                                pop     bx
16852 000014C2 5F                                pop     di
16853 000014C3 07                                pop     es
16854 000014C4 C3                                retn
16855
16856                                   ; =============== S U B   R O U T I N E =======================================
16857
16858                                   ; 16/10/2022
16859
16860                                   ; ---------------------------------------------------------------------------
16861                                   ;   ChangeLineChk
16862                                   ; ---------------------------------------------------------------------------
16863                                   ;
16864                                   ; when the user calls get/set media id call before dos establishes the media
16865                                   ; by calling "media_chk",the change line activity of the drive is going to be
16866                                   ; lost.     this routine will check the change line activity and will save the
16867                                   ; history in the flags.
16868                                   ;
```

```
16869                                   ; FUNCTION: check the change line error activity
16870                                   ;
16871                                   ; INPUT :  ES:di -> bds table.
16872                                   ;
16873                                   ; OUTPUT:   flag in bds table will be updated if change line occurs.
16874                                   ;
16875                                   ; SUBROUTINES TO BE CALLED:
16876                                   ;   Set_Changed_DL
16877                                   ;
16878                                   ; ---------------------------------------------------------------------------
16879
16880                                           ; 24/12/2023 - Retro DOS 5.0
16881                                   ChangeLineChk:
16882 000014C5 268A5504                           mov     dl, [es:di+4]  ; [es:di+BDS.drivenum]
16883 000014C9 08D2                               or      dl, dl         ; Fixed disk?
16884 000014CB 7821                               js      short ChangeLnChkRet ; Yes, skip it.
16885                                             ; 24/12/2023
16886 000014CD 26F6453F04                         test    byte [es:di+3Fh], 4 ; [es:di+BDS.flags] ; PCDOS 7.1
16887                                             ; 12/12/2022
16888                                             ;test  byte [es:di+23h], 4
16889                                             ;;test  word [es:di+23h], 4 ; [es:di+BDS.flags]
16890                                                                     ; return_fake_bpb
16891 000014D2 751A                               jnz     short ChangeLnChkRet
16892 000014D4 803E[7700]01                       cmp     byte [fhave96], 1   ; This rom support change line?
16893 000014D9 7513                               jnz     short ChangeLnChkRet
16894 000014DB E8AD07                             call    haschange       ; This drive support change line?
16895 000014DE 740E                               jz      short ChangeLnChkRet ; Do nothing
16896
16897                                             ; Execute the rom disk interrupt to check changeline activity.
16898
16899 000014E0 B416                               mov     ah, 16h
16900 000014E2 CD13                               int     13h     ; DISK - FLOPPY       DISK - CHANGE OF DISK STATUS (AT,XT2,XT286,CONV,PS)
16901                                                             ; DL = drive to       check
16902                                                             ; Return: AH = disk change status
16903 000014E4 7308                               jnb     short ChangeLnChkRet
16904 000014E6 53                                 push    bx
16905 000014E7 BB4000                             mov     bx, 40h         ; fchanged
16906                                                                     ; Update flag in BDS for this
16907                                                                     ; physical drive
16908 000014EA E87707                             call    set_changed_dl
16909 000014ED 5B                                 pop     bx
16910                                   ChangeLnChkRet:
16911 000014EE C3                                 retn
16912
16913                                   ; ---------------------------------------------------------------------------
16914
16915                                   ; 16/10/2022
16916
16917                                   ; ============================================================================
16918                                   ;   GetAccessFlag
16919                                   ; ============================================================================
16920                                   ;
16921                                   ; FUNCTION: get the status of UNFORMATTED_MEDIA bit of flags in bds table
16922                                   ;
16923                                   ; INPUT :
16924                                   ;       ES:di -> bds table
16925                                   ;
16926                                   ; OUTPUT:   a_DiskAccess_Control.dac_access_flag = 0 if disk i/o not allowed.
16927                                   ;                                                  = 1 if disk i/o allowed.
16928                                   ; ============================================================================
16929
16930                                           ; 24/12/2023 - Retro DOS 5.0
16931
16932                                           ; 19/10/2022
16933                                   GetAccessFlag:
16934 000014EF C51E[1200]                         lds     bx, [ptrsav]   ; DS:BX points to request header
16935 000014F3 C55F13                             lds     bx, [bx+19]    ; [bx+IOCTL_REQ.GENERICIOCTL_PACKET]
16936                                             ;mov    al, 0          ; Assume result       is unformatted
16937                                             ; 10/12/2022
16938 000014F6 28C0                               sub     al, al
16939                                             ; 24/12/2023
16940 000014F8 26F6454002                         test    byte [es:di+40h], 02h ; (PCDOS 7.1 IBMBIO.COM)
16941                                             ;test  word ptr es:[di+3Fh], 200h
16942                                             ; 10/12/2022
16943                                             ;test  byte [es:di+36], 02h
16944                                             ;;test  word [es:di+35], 200h ; [es:di+BDS.flags]
16945                                                                     ; unformatted_media
16946 000014FD 7501                               jnz     short GafDone  ; Done if unformatted
16947                                             ;inc    al             ; Return true for formatted
16948                                             ; 24/12/2023
16949 000014FF 40                                 inc     ax
16950                                   GafDone:
16951 00001500 884701                             mov     [bx+1], al     ; [bx+A_DISKACCESS_CONTROL.DAC_ACCESS_FLAG]
16952 00001503 C3                                 retn
16953                                   ; ---------------------------------------------------------------------------
16954
16955                                   ; 16/10/2022
16956
16957                                   ; ============================================================================
16958                                   ;   SetAccessFlag
16959                                   ; ============================================================================
16960                                   ;
16961                                   ; function: set/reset the UNFORMATTED_MEDIA bit of flags in bds table
16962                                   ;
16963                                   ; input :
16964                                   ;       ES:di -> bds table
16965                                   ;
16966                                   ; output:   unformtted_media bit modified according to the user request
16967                                   ; ============================================================================
16968
16969                                           ; 24/12/2023 - Retro DOS 5.0
16970
16971                                           ; 19/10/2022
16972                                   SetAccessFlag:
16973 00001504 C51E[1200]                         lds     bx, [ptrsav]   ; ES:BX points to request header
16974 00001508 C55F13                             lds     bx, [bx+19]    ; [bx+IOCTL_REQ.GENERICIOCTL_PACKET]
16975                                             ; 24/12/2023
16976 0000150B 26806540FD                         and     byte [es:di+40h], 0FDh ; (PCDOS 7.1 IBMBIO.COM)
16977                                             ;and   word ptr es:[di+3Fh], 0FDFFh
16978                                             ; 10/12/2022
16979                                             ;and   byte [es:di+36], 0FDh
16980                                             ;;and  word [es:di+35], 0FDFFh ; [es:di+BDS.flags]
16981                                                                     ; ~unformatted_media
16982 00001510 807F0100                           cmp     byte [bx+1], 0 ; [bx+A_DISKACCESS_CONTROL.DAC_ACCESS_FLAG]
16983 00001514 7505                               jnz     short saf_Done
16984                                             ; 24/12/2023
16985 00001516 26804D4002                         or      byte [es:di+40h], 02h ; (PCDOS 7.1 IBMBIO.COM)
16986                                             ;or    word ptr es:[di+3Fh], 200h
16987                                             ; 15/04/2024
16988                                             ; 10/12/2022
16989                                             ;or    byte [es:di+36], 02h
16990                                             ;;or   word [es:di+35], 200h ; [es:di+BDS.flags]
16991                                                                     ; unformatted_media
16992                                   saf_Done:
```

```
16993 0000151B C3                       retn
16994                           ; ---------------------------------------------------------------------------
16995
16996                           ; 16/10/2022
16997
16998                           ; ============================================================================
16999                           ; Ioctl_Support_Query
17000                           ; ============================================================================
17001                           ;
17002                           ; New device command which was added in DOS 5.00 to allow a query of a
17003                           ; specific GENERIC IOCtl to see if it is supported. Bit 7 in the
17004                           ; device attributes specifies if this function is supported.
17005                           ;
17006                           ; ============================================================================
17007
17008                                   ; 24/12/2023 - Retro DOS 5.0
17009
17010                                   ; 19/10/2022
17011                           ioctl_support_query:
17012 0000151C 06                       push    es
17013 0000151D C41E[1200]               les     bx, [ptrsav]   ; ES:BX Points to request header.
17014 00001521 268B470D                 mov     ax, [es:bx+13] ; [es:bx+IOCTL_REQ.MAJORFUNCTION]
17015                                                          ; AL == Major, AH == Minor
17016                                   ; 24/12/2023
17017                                   ; 02/09/2023 (PCDOS 7.1)
17018 00001525 3C48                     cmp     al, 48h        ; IOC_NEW_DC (PCDOS 7.1)
17019                                                          ; new generic ioctl function (FAT32)
17020 00001527 7404                     je      short ioctl_support
17021
17022 00001529 3C08                     cmp     al, 8          ; IOC_DC
17023                                                          ; See if major code is 8
17024 0000152B 7513                     jne     short nosupport
17025                           ioctl_support:
17026 0000152D 0E                       push    cs
17027 0000152E 07                       pop     es
17028                                   ; 24/12/2023
17029                                   ; 02/09/2023
17030 0000152F B90E00                   mov     cx, 14          ; (PCDOS 7.1) IOC_DC_TABLE_LEN
17031                                   ;mov    cx, 11          ; IOC_DC_TABLE_LEN
17032                                   ; 10/12/2022
17033 00001532 BF[BD0E]                 mov     di, IOC_DC_Table
17034                                   ;mov    di, 0C60h       ; IOC_DC_Table
17035                                                          ; at 2C7h:0C60h       = 70h:31D0h
17036 00001535 86E0                     xchg    al, ah          ; Put minor code in AL
17037 00001537 F2AE                     repne scasb             ; Scan for minor code in AL
17038 00001539 7505                     jnz     short nosupport       ; it was not found
17039 0000153B B80001                   mov     ax, 100h
17040                                   ; 10/12/2022
17041                                   ; (jump to ioctlsupexit is not required)
17042                                   ;jmp    short $+2       ; ioctlsupexit
17043                                                          ; Signal ioctl is supported
17044                                   ;;jmp   short ioctlsupexit
17045                           ; ---------------------------------------------------------------------------
17046                           ioctlsupexit:
17047 0000153E 07                       pop     es
17048                                   ; 10/12/2022
17049                                   ; cf = 0
17050                                   ;clc
17051 0000153F C3                       retn
17052                           ; ---------------------------------------------------------------------------
17053                           nosupport:
17054 00001540 07                       pop     es
17055 00001541 E991EB                   jmp     bc_cmderr
17056                           ; ---------------------------------------------------------------------------
17057
17058                           ; 16/10/2022
17059
17060                           ; ============================================================================
17061                           ;   GetMediaSenseStatus
17062                           ; ============================================================================
17063                           ;
17064                           ; FUNCTION: Will return the type of diskette media in the specified DOS
17065                           ;       diskette drive and whether the media is the default type
17066                           ;       for that drive. (default type means the max size for that
17067                           ;       drive)
17068                           ;
17069                           ; INPUT :   ES:DI -> BDS table
17070                           ; OUTPUT:   If carry clear
17071                           ;       DS:BX -> Updated IOCtlPacket
17072                           ;
17073                           ;                   Special Function at offset 0:
17074                           ;                         0      - Media detected is not default type
17075                           ;                         1      - Media detected is default type
17076                           ;
17077                           ;                   Device Type at offset 1:
17078                           ;                         2       - 720K 3.5" 80 tracks
17079                           ;                         7      - 1.44M 3.5" 80 tracks
17080                           ;                         9      - 2.88M 3.5" 80 tracks
17081                           ;
17082                           ; Error Codes returned in AX if carry set:
17083                           ;
17084                           ; 8102 - Drive not ready  - No disk is in the drive.
17085                           ; 8107 - Unknown media type       - Drive doesn't support this function or
17086                           ;                           the media is really unkown, any error
17087                           ;                           other than "media not present"
17088                           ;
17089                           ; ============================================================================
17090
17091                                   ; 19/10/2022
17092                           SenseMediaType:
17093 00001544 C51E[1200]               lds     bx, [ptrsav]   ; DS:BX points to request header.
17094 00001548 C55F13                   lds     bx, [bx+19]    ; bx+IOCTL_REQ.GENERICIOCTL_PACKET]
17095                                   ; 10/10/2022
17096                                   ;mov    word [bx], 0   ; Initialize the 2 packet bytes
17097 0000154B 31D2                     xor     dx, dx
17098 0000154D 8917                     mov     [bx], dx ; 0
17099                                   ;
17100 0000154F 268A5504                 mov     dl, [es:di+4]  ; [es:di+BDS.drivenum]
17101                                                          ; Get int 13h drive number from      BDS
17102                                   ; 10/12/2022
17103                                   ;xor     dh, dh         ; DX = physical       drive number
17104 00001553 B420                     mov     ah, 20h        ; Get Media Type function
17105                                                          ; If no carry media type in AL
17106 00001555 CD13                     int     13h            ; DISK - QCACHE       - DISMOUNT
17107 00001557 7216                     jc      short MediaSenseEr ; error code      in AH
17108 00001559 FE07                     inc     byte [bx]      ; Signal media type is default (bit 1)
17109                           DetermineMediaType:
17110 0000155B FEC8                     dec     al
17111 0000155D 3C02                     cmp     al, 2          ; Chk for 720K ie: (3-1) = 2
17112 0000155F 740A                     jz      short GotMediaType
17113 00001561 0404                     add     al, 4
17114 00001563 3C07                     cmp     al, 7          ; Chk for 1.44M ie: (4-1+4) = 7
17115 00001565 7404                     jz      short GotMediaType
17116 00001567 3C09                     cmp     al, 9          ; Chk for 2.88M       ie: (6-1+4) = 9
```

```
17117 00001569 7510                     jnz     short UnknownMediaType ; Just didn't recognize media type
17118                                   GotMediaType:
17119 0000156B 884701                   mov     [bx+1],al       ; Save the return value
17120                                   ; 10/12/2022
17121                                   ; cf = 0
17122                                   ;clc                    ; Signal success
17123 0000156E C3                       retn
17124                                   ; ---------------------------------------------------------------------------
17125
17126                                   MediaSenseEr:
17127 0000156F 80FC32                   cmp     ah, 32h         ; See if not default media error
17128 00001572 74E7                     jz      short DetermineMediaType ; Not really an error
17129 00001574 B002                     mov     al, 2           ; Now assume drive not ready
17130 00001576 80FC31                   cmp     ah, 31h         ; See if media was present
17131 00001579 7402                     jz      short SenseErrExit ; Return drive not ready
17132                                   UnknownMediaType:
17133 0000157B B007                     mov     al, 7           ; Just don't know the media type
17134                                   SenseErrExit:
17135 0000157D B481                     mov     ah, 81h         ; Signal error return
17136 0000157F F9                       stc
17137 00001580 C3                       retn
17138
17139                                   ; ---------------------------------------------------------------------------
17140                                   ; 10/12/2022
17141                                   ;db    0
17142                                   ; ---------------------------------------------------------------------------
17143
17144                                   ;---------------------------------------------------------------------------
17145                                   ; PCDOS 7.1 IBMBIO.COM - BIOSCODE:15F2h
17146                                   ;---------------------------------------------------------------------------
17147                                   ; 26/12/2023 - Retro DOS v5.0 (Modified PCDOS 7.1)
17148
17149                                   ; =============== S U B R O U T I N E =======================================
17150
17151                                   SetLockState:
17152 00001581 C51E[1200]               lds     bx, [ptrsav]    ; set media lock state
17153 00001585 C55F13                   lds     bx, [bx+13h]    ; [bx+IOCTL_REQ.GENERICIOCTL_PACKET]
17154                                   ;mov    dl, [es:di+4]   ; [es:di+BDS.drivenum]
17155                                   ;call   check_int13h_exts_present
17156                                   ; 26/12/2023
17157 00001588 E82100                   call    check_int13h_exts_p
17158                                   ;mov    al, 3           ; unknown command error
17159 0000158B 721C                     jc      short setlockst_ret
17160 0000158D 8A07                     mov     al, [bx]        ; [bx+A_LOCKSTATE_CONTROL.LOCKSTATE_FUNCTIONS]
17161 0000158F B445                     mov     ah, 45h
17162 00001591 CD13                     int     13h             ; DISK - IBM/MS Extension - LOCK/UNLOCK DRIVE
17163                                   ; (DL - drive, [SI - disk address packet)
17164 00001593 884701                   mov     [bx+1], al      ; 1 = locked, 0 = not locked
17165                                   ; [bx+A_LOCKSTATE_CONTROL.LOCKSTATE_FLAG]
17166
17167                                   ; 26/12/2023
17168 00001596 EB0A                     jmp     short sls_em
17169
17170                                   ;       jnc     short setlockst_ret
17171                                   ;       mov     al, ah
17172                                   ;       call    maperror
17173                                   ;setlockst_ret:
17174                                   ;       mov     ah, 81h         ; Return this status in case of carry
17175                                   ;       retn
17176
17177                                   ; =============== S U B R O U T I N E =======================================
17178
17179                                   EjectMedia:
17180                                   ;mov    dl, [es:di+4]   ; eject media in drive
17181                                   ;                       ; [es:di+BDS.drivenum]
17182                                   ;call   check_int13h_exts_present
17183                                   ; 26/12/2023
17184 00001598 E81100                   call    check_int13h_exts_p
17185                                   ;mov    al, 3           ; unknown command error
17186 0000159B 720C                     jc      short ejectm_ret
17187 0000159D B80046                   mov     ax, 4600h
17188 000015A0 CD13                     int     13h             ; DISK - IBM/MS Extension - EJECT MEDIA
17189                                   ; (DL - drive)
17190                                   sls_em:                 ; 26/12/2023
17191 000015A2 7305                     jnc     short ejectm_ret
17192 000015A4 88E0                     mov     al, ah
17193 000015A6 E800F8                   call    maperror
17194                                   setlockst_ret:          ; 26/12/2023
17195                                   ejectm_ret:
17196 000015A9 B481                     mov     ah, 81h         ; Return this status in case of carry
17197 000015AB C3                       retn
17198
17199                                   ; =============== S U B R O U T I N E =======================================
17200
17201                                   ; 26/12/2023
17202                                   check_int13h_exts_p:
17203 000015AC 268A5504                 mov     dl, [es:di+4]
17204
17205                                   check_int13h_exts_present:
17206 000015B0 B441                     mov     ah, 41h
17207 000015B2 53                       push    bx
17208 000015B3 BBAA55                   mov     bx, 55AAh
17209 000015B6 CD13                     int     13h             ; DISK - Check for INT 13h Extensions
17210                                                           ; BX = 55AAh, DL = drive number
17211                                                           ; Return: CF set if not supported
17212                                                           ; AH = extensions version
17213                                                           ; BX = AA55h
17214                                                           ; CX = Interface support bit map
17215 000015B8 81FB55AA                 cmp     bx, 0AA55h
17216 000015BC 5B                       pop     bx
17217 000015BD 7505                     jnz     short exts_notsupported
17218 000015BF F6C102                   test    cl, 2           ; bit 1 - drive locking and ejecting subset
17219 000015C2 7503                     jnz     short exts_supported
17220                                   exts_notsupported:
17221                                   ; 26/12/2023
17222 000015C4 B003                     mov     al, 3
17223                                   ;
17224 000015C6 F9                       stc
17225                                   exts_supported:
17226 000015C7 C3                       retn
17227
17228                                   ; =============== S U B R O U T I N E =======================================
17229
17230                                   GetDrvMapInfo:
17231 000015C8 8CD9                     mov     cx, ds          ; get drive map information
17232                                                           ;
17233                                                           ; es:di points to BDS which belongs to
17234                                                           ;       the requested logical/dos drive number
17235                                                           ;
17236                                                           ; Format of parameter block:
17237                                                           ; Offset  Description (Table 01570)
17238                                                           ;  00h    (call) length of this buffer (in bytes)
17239                                                           ;  01h    (ret) number of bytes in parameter block
17240                                                           ;          actually used
```

```
                                                        ;  02h    (ret) drive flags
                                                        ;  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
0000162B 80FC13                             cmp     ah, 13h
0000162E 7413                               jz      short int2f_replace_int13
00001630 80FC08                             cmp     ah, 8
00001633 7432                               jz      short mine

                                ; Check for WIN386 startup and return the BIOS instance data

00001635 80FC16                             cmp     ah, 16h         ; MultWin386
00001638 746D                               jz      short win386call
0000163A 80FC4A                             cmp     ah, 4Ah         ; multMULT
0000163D 7503                               jnz     short i2f_handler_iret
0000163F E99800                             jmp     handle_multmult
                                ; ---------------------------------------------------------------------------

                                i2f_handler_iret:
00001642 CF                                 iret
                                ; ---------------------------------------------------------------------------

                                int2f_replace_int13:
00001643 FA                                 cli     ; 26/12/2023
00001644 50                                 push    ax      ; free up a register for caller's ds
00001645 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]
00001647 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.
```

```
17613                                   ;
17614                                   ;   RETURNS : offset of free HMA in DI
17615                                   ;             BIOS_DATA, seg in DS
17616                                   ;
17617                                   ;--------------------------------------------------------------------------
17618
17619                                               ; 17/10/2022
17620                                   HMAPtr:
17621 00001727 2E8E1E[3000]                         mov     ds, [cs:BIOSDATAWORD]
17622                                               ;mov    ds, word ptr cs:BIOSDATAWORD ; [cs:Bios_Data_Word]
17623 0000172C 8B3E[D707]                           mov     di, [FreeHMAPtr]
17624                                               ;mov    di, ds:FreeHMAPtr
17625 00001730 83FFFF                               cmp     di, 0FFFFh
17626 00001733 750F                                 jnz     short HMAPtr_retn
17627 00001735 803E[DD07]00                         cmp     byte [SysinitPresent], 0
17628                                               ;cmp    ds:SysinitPresent, 0
17629 0000173A 7408                                 jz      short HMAPtr_retn
17630 0000173C FF1E[D907]                           call    far [MoveDOSIntoHMA]
17631                                               ;call   ds:MoveDOSIntoHMA ; call far [MoveDOSIntoHMA]
17632 00001740 8B3E[D707]                           mov     di, [FreeHMAPtr]
17633                                               ;mov    di, ds:FreeHMAPtr
17634                                   HMAPtr_retn:
17635 00001744 C3                                   retn
17636
17637                                   ; =============== S U B   R O U TI N E =======================================
17638
17639                                   ; 16/10/2022
17640
17641                                   ; move a 512 byte sector from ds:si to es:di, do not trash cx
17642                                   ; but go ahead and update direction flag, si, & di
17643
17644                                   move_sector:
17645
17646                                   ; The 80386 microprocessor considers an access to WORD 0FFFFh in
17647                                   ; any segment to be a fault. Theoretically, this could be handled
17648                                   ; by the fault handler and the behavior of an 8086 could be emulated
17649                                   ; by wrapping the high byte to offset 0000h. This would be a lot
17650                                   ; of work and was, indeed, blown off by the Win386 guys. COMPAQ
17651                                   ; also handles the fault incorrectly in their ROM BIOS for real
17652                                   ; mode. Their fault handler was only designed to deal with one
17653                                   ; special case which occurred in a magazine benchmark, but didn't
17654                                   ; handle the general case worth beans.
17655                                   ;
17656                                   ; Simply changing this code to do a byte loop would work okay but
17657                                   ; would involve a general case performance hit. Therefore, we'll
17658                                   ; check for either source or destination offsets being within one
17659                                   ; sector of the end of their segments and only in that case fall
17660                                   ; back to a byte move.
17661
17662 00001745 FC                                     cld
17663 00001746 51                                   push    cx
17664 00001747 B90001                               mov     cx, 256
17665 0000174A 81FE00FE                             cmp     si, 0FE00h
17666 0000174E 770A                                 ja      short movsec_bytes
17667 00001750 81FF00FE                             cmp     di, 0FE00h
17668 00001754 7704                                 ja      short movsec_bytes
17669 00001756 F3A5                                 rep movsw
17670 00001758 59                                   pop     cx
17671 00001759 C3                                   retn
17672                                   ; ---------------------------------------------------------------------------
17673
17674                                   movsec_bytes:
17675 0000175A D1E1                                 shl     cx, 1
17676 0000175C F3A4                                 rep movsb
17677 0000175E 59                                   pop     cx
17678 0000175F C3                                   retn
17679
17680                                   ; =============== S U B   R O U TI N E =======================================
17681
17682                                   ; 16/10/2022
17683
17684                                   ; check_wrap is a routine that adjusts the starting sector, starting head
17685                                   ; and starting cylinder for an int 13 request that requests i/o of a lot
17686                                   ; of sectors. it only does this for fixed disks. it is used in the sections
17687                                   ; of code that handle ecc errors and dma errors. it is necessary, because
17688                                   ; ordinarily the rom would take care of wraps around heads and cylinders,
17689                                   ; but we break down a request when we get an ecc or dma error into several
17690                                   ; i/o of one or more sectors. in this case, we may already be beyond the
17691                                   ; number of sectors on a track on the medium, and the request would fail.
17692                                   ;
17693                                   ; input conditions:
17694                                   ;   all registers set up for an int 13 request.
17695                                   ;
17696                                   ; output:
17697                                   ;   dh - contains starting head number for request
17698                                   ;   cx - contains starting sector and cylinder numbers
17699                                   ;   (the above may or may not have been changed, and are 0-based)
17700                                   ;   all other registers preserved.
17701
17702                                               ; 26/12/2023 - Retro DOS 5.0
17703                                   check_wrap:
17704 00001760 50                                   push    ax
17705 00001761 53                                   push    bx
17706 00001762 06                                   push    es
17707 00001763 57                                   push    di
17708 00001764 E86C00                               call    find_bds        ; get pointer to bds for drive in dl
17709 00001767 725E                                 jb      short no_wrap   ; finished if DOS doesn't use it
17710                                               ; 26/12/2023
17711 00001769 26F6453F01                           test    byte [es:di+3Fh], 1
17712                                               ; 12/12/2022
17713                                               ;test   byte [es:di+23h], 1
17714                                               ;;test  word [es:di+23h], 1 ; [es:di+BDS.flags],fnon_removable
17715 0000176E 7457                                 jz      short no_wrap   ; no wrapping for removable media
17716 00001770 268B5D13                             mov     bx, [es:di+13h]         ; [es:di+BDS.secpertrack]
17717 00001774 89C8                                 mov     ax, cx
17718 00001776 83E03F                               and     ax, 3Fh         ; extract sector number
17719 00001779 39D8                                 cmp     ax, bx          ; are we going to wrap?
17720 0000177B 764A                                 jbe     short no_wrap
17721 0000177D F6F3                                 div     bl              ; ah=new sector       #, al=# of head wraps
17722
17723                                   ; we need to be careful here. if the new sector # is 0, then we are on the
17724                                   ; last sector on that track.
17725
17726 0000177F 08E4                                 or      ah, ah
17727 00001781 7503                                 jnz     short not_on_bound
17728                                               ; 18/12/2022
17729 00001783 48                                   dec     ax ; *
17730 00001784 88DC                                 mov     ah, bl          ; set sector=BDS_BPB.BPB_SECTORSPERTRACK
17731                                                                       ; if on boundary
17732                                               ;dec    al ; *          ; also decrement # of head wraps
17733                                   not_on_bound:
17734 00001786 80E1C0                               and     cl, 0C0h        ; zero out sector #
17735 00001789 08E1                                 or      cl, ah          ; or in new sector #
17736 0000178B 30E4                                 xor     ah, ah          ; ax = # of head wraps
```

```
                                        inc     ax
                                        add     al, dh          ; add in starting head #
                                        adc     ah, 0           ; catch any carry
                                        ; 02/09/2023
                                        mov     bx, [es:di+15h]         ; [es:di+BDS.heads]
                                        cmp     ax, bx
                                        ;cmp    ax, [es:di+15h]         ; [es:di+BDS.heads]
                                                                ; are we going to wrap around a      head?
                                        jbe     short no_wrap_head ; do      not lose new head number!!
                                        push    dx              ; preserve drive number and head number
                                        xor     dx, dx
                                        ;mov    bx, [es:di+15h]         ; [es:di+BDS.heads]
                                        div     bx              ; dx=new head #, ax=# of cylinder wraps

                                ; careful here! if new head # is 0, then we are on the last head.

                                        or      dx, dx
                                        jnz     short no_head_bound
                                        mov     dx, bx          ; on boundary. set to BDS_BPB.BPB_HEADS

                                ; if we had some cylinder wraps, we need to reduce them by one!!

                                        or      ax, ax
                                        jz      short no_head_bound
                                        dec     ax              ; reduce number       of cylinder wraps
                                no_head_bound:
                                        mov     bh, dl          ; bh has new head number
                                        pop     dx              ; restore drive number and head number
                                        dec     bh              ; get it 0-based
                                        mov     dh, bh          ; set up new head number in dh
                                        mov     bh, cl
                                        and     bh, 3Fh         ; preserve sector number
                                        mov     bl, 6
                                        xchg    cl, bl
                                        shr     bl, cl          ; get ms cylinder bits to ls end
                                        add     ch, al          ; add in cylinder wrap
                                        adc     bl, ah          ; add in high byte
                                        shl     bl, cl          ; move up to ms       end
                                        xchg    bl, cl          ; restore cylinder bits       into cl
                                        or      cl, bh          ; or in sector number
                                no_wrap:
                                        clc
                                        pop     di
                                        pop     es
                                        pop     bx
                                        pop     ax
                                        retn
                                ; ---------------------------------------------------------------------------

                                no_wrap_head:
                                        mov     dh, al          ; do not lose new head number
                                        dec     dh              ; get it 0-based
                                        jmp     short no_wrap

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

                                ; 16/10/2022

                                ; this is a special version of the bds lookup code which is
                                ; based on physical drives rather than the usual logical drives
                                ; carry is set if the physical drive in dl is found, es:di -> its bds
                                ; otherwise carry is clear
                                ;
                                ; guaranteed to trash no registers except es:di

                                        ; 19/10/2022
                                find_bds:
                                        les     di, [start_bds]         ; point es:di to first bds
                                fbds_1:
                                        cmp     [es:di+4], dl   ; [es:di+BDS.drivenum]
                                        jz      short fdbs_2
                                        les     di, [es:di]     ; [es:di+BDS.link]
                                                                ; go to next bds
                                        cmp     di, 0FFFFh
                                        jnz     short fbds_1
                                        stc
                                fdbs_2:
                                        retn

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

                                ; 16/10/2022
                                        ; 17/10/2022
                                doint:
                                        ; 10/12/2022
                                        mov     dl, [bp+8]      ; [bp+INT13FRAME.olddx]
                                                                ; get physical drive number
                                        ; 19/10/2022 - Temporary !
                                        ;db     8Ah, 96h, 8, 0 ; mov dl, [bp+8]

                                        xor     ah, ah
                                        or      al, al
                                        jz      short dointdone         ; if zero sectors, return ax=0
                                        ; 10/12/2022
                                        mov     ah, [bp+3]      ; [bp+INT13FRAME.oldax+1]
                                                                ; get request code
                                        ;db     8Ah, 0A6h, 3, 0         ; mov ah, [bp+3]
                                        push    word [bp+10h]   ; [bp+INT13FRAME.oldf]
                                        ;db     0FFh, 0B6h, 10h, 0 ; push word [bp+10h]
                                        popf
                                        ;call   far 70h:797h ; MSDOS 6.21 IO.SYS BIOSCODE:14EAh
                                        ; 17/10/2022
                                        call    DOSBIOSSEG:call_orig13
                                        ;;call  call_orig13     ; call far 70h:797h
                                                                ; call far KERNEL_SEGMENT:call_orig13
                                        pushf
                                        ; 10/12/2022
                                        pop     word [bp+10h]   ; [bp+INT13FRAME.oldf]
                                        ;db     8Fh, 86h, 10h, 0 ; pop word [bp+10h]
                                dointdone:
                                        retn

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

                                ; 16/10/2022

                                ; this is the true int 13 handler. we parse the request to see if there is
                                ; a dma violation. if so, depending on the function, we:
                                ;   read/write break the request into three pieces and move the middle one
                                ;           into our internal buffer.
                                ;
                                ;   format      copy the format table into the buffer
                                ;   verify      point the transfer address into the buffer
                                ;
```

```
                                ; this is the biggest bogosity of all. the ibm controller does not handle
                                ; 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
```

```
17985                                           ;call   call_orig13    ; call far 70:797h
17986                                                                   ; call far KERNEL_SEGMENT:call_orig13
17987 00001854 58                               pop     ax
17988 00001855 EB22                             jmp     short ps2_special_stuff_done
17989                                   ; ---------------------------------------------------------------------------
17990
17991                                   ; 17/10/2022
17992                                   ; 16/10/2022
17993
17994                                   ; here is the actual int13 handler
17995
17996                                   i13z:                                   ; 0070h:3ABBh =       02C7h:154Bh
17997
17998                                   ; cas -- inefficient! could push ds and load ds-> Bios_Data before
17999                                   ; vectoring up here from Bios_Data
18000
18001                                           ; 19/10/2022
18002 00001857 1E                               push    ds              ; save caller's ds register first thing
18003                                           ;;mov   ds, word [cs:0030h]
18004                                                                   ; and set up our own ds -> Bios_Data
18005 00001858 2E8E1E[3000]                     mov     ds, [cs:BIOSDATAWORD]
18006                                           ;mov    ds, word ptr cs:BIOSDATAWORD ; [cs:0030h]
18007                                                                   ; = [02C7h:0030h] = [0070h:25A0h]
18008
18009                                   ; let the operation proceed. if there is a dma violation, then we do things
18010
18011 0000185D A3[1E00]                         mov     [prevoper], ax ; save request
18012 00001860 80FC05                           cmp     ah, 5           ; romformat
18013 00001863 74A0                             jz      short format_special_stuff
18014                                                                   ; go do special stuff for format
18015                                   format_special_stuff_done:
18016 00001865 803E[A204]00                     cmp     byte [ec35flag], 0 ; any electrically compat 3.5 inchers?
18017 0000186A 75AA                             jnz     short ec35_special_stuff
18018                                                                   ; go handle it out of line if so
18019                                   ec35_special_stuff_done:
18020                                           ; 26/12/2023
18021                                           ;call   70h:70Bh ; PCDOS 7.1 IBMBIO.COM BIOSCODE:18EDh
18022                                           ;;call  far 70:797h ; MSDOS 6.21 IO.SYS BIOSCODE:1560h
18023 0000186C 9A[0B07]7000                     call    DOSBIOSSEG:call_orig13
18024                                           ;call   call_orig13    ; call far KERNEL_SEGMENT:call_orig13
18025
18026 00001871 9C                               pushf                   ; save result flags
18027
18028 00001872 803E[AF05]FA                     cmp     byte [model_byte], 0FAh ; is this a ps2/30?
18029                                                                   ; mdl_ps2_30
18030 00001877 74C5                             jz      short ps2_special_stuff
18031                                                                   ; exit mainline to address special
18032                                   ps2_special_stuff_done:                 ; ps2/30 problem if so
18033 00001879 9D                               popf
18034 0000187A 7221                             jb      short goterr13 ; error on original orig13 call-thru?
18035                                   ret_from_i13:
18036 0000187C 1F                               pop     ds
18037 0000187D CA0200                           retf    2               ; restore ds & iret w/flags
18038                                   ; ---------------------------------------------------------------------------
18039
18040                                   ; most of our code exits through here. If carry isn't set, then
18041                                   ; just do a simple exit. Else doublecheck that we aren't getting
18042                                   ; a changeline error.
18043
18044                                   i13ret_ck_chglinerr:
18045 00001880 73FA                             jnb     short ret_from_i13 ; done if not an error termination
18046                                   i13_ret_error:
18047 00001882 80FC06                           cmp     ah, 6           ; did i see a change event?
18048 00001885 7513                             jnz     short int13b    ; skip if wrong       error
18049 00001887 08D2                             or      dl, dl          ; is this for the hard disk?
18050 00001889 780F                             js      short int13b    ; yes, ignore
18051 0000188B 803E[7700]00                     cmp     byte [fhave96], 0
18052 00001890 7408                             jz      short int13b    ; just in case ROM returned this
18053                                                                   ; error even though it told us it
18054                                                                   ; never would
18055 00001892 53                               push    bx
18056 00001893 BB4000                           mov     bx, 40h         ; fchanged
18057 00001896 E8CB03                           call    set_changed_dl
18058 00001899 5B                               pop     bx
18059                                   int13b:
18060 0000189A F9                               stc                     ; now return the error
18061 0000189B EBDF                             jmp     short ret_from_i13
18062                                   ; ---------------------------------------------------------------------------
18063
18064                                   ; some kind of error occurred. see if it is dma violation
18065
18066                                   goterr13:
18067 0000189D 80FC09                           cmp     ah, 9           ; dma error?
18068 000018A0 747C                             jz      short gotdmaerr
18069                                   goterr13_xxxx:
18070 000018A2 80FC11                           cmp     ah, 11h         ; ecc error?
18071 000018A5 75DB                             jnz     short i13_ret_error ; other error. just      return back.
18072 000018A7 803E[A905]01                     cmp     byte [media_set_for_format], 1 ; formatting?
18073 000018AC 74D4                             jz      short i13_ret_error
18074
18075 000018AE 803E[1F00]02                     cmp     byte [prevoper+1], 2
18076                                           ;cmp    byte ptr ds:prevoper+1,       2 ; ecc-corrected error
18077                                                                   ; (2 = romread)
18078                                                                   ; ECC correction only applies to reads
18079 000018B3 75CD                             jnz     short i13_ret_error
18080
18081 000018B5 30E4                             xor     ah, ah
18082                                           ;;call  far 70:797h ; MSDOS 6.21 IO.SYS BIOSCODE:15ABh
18083                                           ; 17/10/2022
18084 000018B7 9A[0B07]7000                     call    DOSBIOSSEG:call_orig13
18085                                           ;call   call_orig13    ; call far KERNEL_SEGMENT:call_orig13
18086                                                                   ; call far 70:797h
18087 000018BC A1[1E00]                         mov     ax, [prevoper]
18088 000018BF 30E4                             xor     ah, ah          ; return code =       no error
18089 000018C1 3C01                             cmp     al, 1           ; if request for one sector, assume ok
18090 000018C3 74B7                             jz      short ret_from_i13 ; return with carry clear
18091 000018C5 53                               push    bx
18092 000018C6 51                               push    cx
18093 000018C7 52                               push    dx
18094 000018C8 A2[2000]                         mov     [number_of_sec], al
18095                                   loop_ecc:
18096 000018CB B80102                           mov     ax, 201h        ; read one sector
18097
18098                                   ; we do reads one sector at a time. this ensures that we will eventually
18099                                   ; finish the request since ecc errors on one sector do read in that sector.
18100                                   ;
18101                                   ; we need to put in some "intelligence" into the ecc handler to handle reads
18102                                   ; that attempt to read more sectors than are available on a particular
18103                                   ; track.
18104                                   ;
18105                                   ; we call check_wrap to set up the sector #, head # and cylinder # for
18106                                   ; this request.
18107                                   ;
18108                                   ; at this point, all registers are set up for the call to orig13, except
```

```
                                ; that there may be a starting sector number that is bigger than the number
                                ; 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
```

```
18233 00001951 B480                             mov     ah, 128         ; ah = max number of sectors in      segment
18234 00001953 28F4                             sub     ah, dh
18235
18236                                   ; ah is now the number of sectors that we can successfully write in this
18237                                   ; segment. if this number is above or equal to the requested number, then we
18238                                   ; continue the operation as normal. otherwise, we break it into pieces.
18239                                   ;
18240                                   ; wait a sec. this is goofy. the whole reason we got here in the
18241                                   ; first place is because we got a dma error. so it's impossible
18242                                   ; for the whole block to fit, unless the dma error was returned
18243                                   ; in error.
18244
18245 00001955 38C4                             cmp     ah, al          ; can we fit it       in?
18246 00001957 7236                             jb      short doblock   ; no, perform blocking.
18247
18248                                   ; yes, the request fits. let it happen.
18249
18250 00001959 8A7609                           mov     dh, [bp+9]      ; [bp+INT13FRAME.olddx+1]
18251                                                                   ; set up head number
18252 0000195C E888FE                           call    doint
18253 0000195F E9D900                           jmp     bad13           ; and return from this place
18254                                   ; ---------------------------------------------------------------------------
18255
18256                                   i13_done_dmaerr:
18257 00001962 B409                             mov     ah, 9           ; pass dma error thru to caller
18258 00001964 F9                               stc
18259 00001965 E914FF                           jmp     ret_from_i13    ; return with error,
18260                                                                   ; we know it's not a changeline error
18261                                   ; ---------------------------------------------------------------------------
18262
18263                                   ; verify the given sectors. place the buffer pointer into our space.
18264
18265                                   intverify:
18266 00001968 06                               push    es              ; save caller's dma address
18267 00001969 53                               push    bx
18268 0000196A 1E                               push    ds              ; es:bx -> Bios_Data:disksector
18269 0000196B 07                               pop     es
18270                                   dosimple:
18271 0000196C BB[5201]                         mov     bx, disksector
18272                                                                   ; do the i/o from Bios_Data:disksector
18273                                           ;;call  far 70:797h ; MSDOS 6.21 IO.SYS BIOSCODE:1665h
18274                                           ; 17/10/2022
18275 0000196F 9A[0B07]7000                     call    DOSBIOSSEG:call_orig13
18276                                           ;call   call_orig13     ; call far KERNEL_SEGMENT:call_orig13
18277 00001974 5B                               pop     bx
18278 00001975 07                               pop     es
18279 00001976 E907FF                           jmp     i13ret_ck_chglinerr
18280                                   ; ---------------------------------------------------------------------------
18281
18282                                   ; format operation. copy the parameter table into Bios_Data:disksector
18283
18284                                   intformat:
18285 00001979 06                               push    es
18286 0000197A 53                               push    bx
18287 0000197B 56                               push    si
18288 0000197C 57                               push    di
18289 0000197D 1E                               push    ds
18290
18291                                   ; point ds to the caller's dma buffer, es to Bios_Data
18292                                   ; in other words, swap (ds, es)
18293
18294 0000197E 06                               push    es
18295 0000197F 1E                               push    ds
18296 00001980 07                               pop     es
18297 00001981 1F                               pop     ds
18298 00001982 89DE                             mov     si, bx
18299 00001984 BF[5201]                         mov     di, disksector
18300 00001987 E8BBFD                           call    move_sector     ; user's data into Bios_Data:disksector
18301 0000198A 1F                               pop     ds
18302 0000198B 5F                               pop     di
18303 0000198C 5E                               pop     si              ; do the i/o from
18304 0000198D EBDD                             jmp     short dosimple ; Bios_Data:disksector
18305                                   ; ---------------------------------------------------------------------------
18306
18307                                   ; we can't fit the request into the entire block. perform the operation on
18308                                   ; the first block.
18309                                   ;
18310                                   ; doblock is modified to correctly handle multi-sector disk i/o.
18311                                   ; old doblock had added the number of sectors i/oed (ah in old doblock) after
18312                                   ; the doint call to cl. observing only the lower 6 bits of cl(=max. 64) can
18313                                   ; represent a starting sector, if ah was big, then cl would be clobbered.
18314                                   ; by the way, we still are going to use cl for this purpose since checkwrap
18315                                   ; routine will use it as an input. to prevent cl from being clobbered, a
18316                                   ; safe number of sectors should be calculated like "63 - # of sectors/track".
18317                                   ; doblock will handle the first block of requested sectors within the
18318                                   ; boundary of this safe value.
18319
18320                                           ; 26/12/2023 - Retro DOS v5.0
18321                                   doblock:
18322
18323                                   ; try to get the # of sectors/track from bds via rom drive number.
18324                                   ; for any mini disks installed, here we have to pray that they have the
18325                                   ; same # of sector/track as the main dos partition disk drive.
18326
18327 0000198F 8B5608                           mov     dx, [bp+8]      ; [bp+INT13FRAME.olddx]
18328                                                                   ; get head #, drive #
18329 00001992 51                               push    cx
18330 00001993 06                               push    es
18331 00001994 57                               push    di              ; ah - # of sectors before dma boundary
18332                                                                   ; al - requested # of sectors for i/o.
18333 00001995 E83BFE                           call    find_bds
18334 00001998 268B4D13                         mov     cx, [es:di+13h]       ; [es:di+BDS.secpertrack]
18335                                           ; 26/12/2023
18336 0000199C 26F6453F01                       test    byte [es:di+3Fh], 1
18337                                           ; 12/12/2022
18338                                           ;test   byte [es:di+23h], 1
18339                                           ;;test  word [es:di+23h], 1 ; [es:di+BDS.flags],fnon_removable
18340 000019A1 5F                               pop     di
18341 000019A2 07                               pop     es
18342 000019A3 88E0                             mov     al, ah          ; set al=ah for       floppies
18343 000019A5 7404                             jz      short doblockflop ; they are track by track operation
18344 000019A7 B43F                             mov     ah, 63          ; ah = 63-secpt       (# safe sectors??)
18345 000019A9 28CC                             sub     ah, cl          ; al - # of sectors before dma boundary
18346                                   doblockflop:
18347 000019AB 59                               pop     cx
18348                                   doblockcontinue:
18349 000019AC 38C4                             cmp     ah, al          ; if safe_# >= #_of_sectors_to_go_before dma,
18350 000019AE 7305                             jnb     short doblocklast ; then #_of_sectors_to_go as it is for doint.
18351 000019B0 50                               push    ax
18352 000019B1 88E0                             mov     al, ah          ; otherwise, set al to ah to operate.
18353 000019B3 EB03                             jmp     short doblockdoint
18354                                   ; ---------------------------------------------------------------------------
18355
18356                                   doblocklast:
```

```
18357 000019B5 88C4                             mov     ah, al
18358 000019B7 50                               push    ax
18359                                   doblockdoint:                           ; let ah = al =       # of sectors for this shot
18360 000019B8 E82CFE                           call    doint
18361 000019BB 727E                             jb      short bad13     ; something happened, bye!
18362 000019BD 58                               pop     ax
18363 000019BE 286602                           sub     [bp+2], ah      ; sub [bp+INT13FRAME.oldax], ah
18364                                                                   ; decrement by the successful operation
18365 000019C1 00E1                             add     cl, ah          ; advance sector #. safety gauranteed.
18366 000019C3 00E7                             add     bh, ah          ; advance dma addres
18367 000019C5 00E7                             add     bh, ah          ; twice for 512 byte sectors
18368 000019C7 38C4                             cmp     ah, al          ; check the previous value
18369 000019C9 740A                             jz      short buffer    ; if #_of_sectors_to_go       < safe_#,
18370                                                                   ; then we are done already.
18371 000019CB 28E0                             sub     al, ah          ; otherwise,
18372                                                                   ; #_sector_to_go = #_of_sector_to_go - safe_#
18373 000019CD E890FD                           call    check_wrap      ; get new cx, dh for the next operation.
18374 000019D0 EBDA                             jmp     short doblockcontinue ;       handles next sectors left.
18375                                   ; ---------------------------------------------------------------------------
18376
18377                                   bufferx:
18378 000019D2 8A7609                           mov     dh, [bp+9]      ; [bp+INT13FRAME.olddx+1]
18379                                                                   ; set up head number
18380                                   buffer:
18381 000019D5 53                               push    bx
18382 000019D6 8A6603                           mov     ah, [bp+3]      ; [bp+INT13FRAME.oldax+1]
18383 000019D9 80FC03                           cmp     ah, 3           ; romwrite
18384 000019DC 7525                             jnz     short doread    ;
18385
18386                                   ; copy the offending sector into local buffer
18387
18388 000019DE 06                               push    es
18389 000019DF 1E                               push    ds
18390 000019E0 56                               push    si
18391 000019E1 57                               push    di
18392 000019E2 1E                               push    ds              ; exchange segment registers
18393 000019E3 06                               push    es
18394 000019E4 1F                               pop     ds
18395 000019E5 07                               pop     es
18396 000019E6 BF[5201]                         mov     di, disksector ; where to move
18397 000019E9 57                               push    di              ; save it
18398 000019EA 89DE                             mov     si, bx          ; source
18399 000019EC E856FD                           call    move_sector     ; move sector into local buffer
18400 000019EF 5B                               pop     bx              ; new transfer address
18401                                                                   ; (es:bx = Bios_Data:diskbuffer)
18402 000019F0 5F                               pop     di              ; restore caller's di & si
18403 000019F1 5E                               pop     si
18404 000019F2 1F                               pop     ds              ; restore Bios_Data
18405
18406                                   ; see if we are wrapping around a track or head
18407
18408 000019F3 B001                              mov     al, 1           ; [bp+INT13FRAME.olddx]
18409                                                                   ; get drive number
18410 000019F5 8A5608                           mov     dl, [bp+8]
18411 000019F8 E865FD                           call    check_wrap      ; sets up registers if wrap-around
18412                                                                   ;
18413                                                                   ; ah is function
18414                                                                   ; al is 1 for single sector transfer
18415                                                                   ; es:bx is local transfer addres
18416                                                                   ; cx is track/sector number
18417                                                                   ; dx is head/drive number
18418                                                                   ; si,di unchanged
18419 000019FB E8E9FD                           call    doint
18420 000019FE 07                               pop     es              ; restore caller's dma segment
18421 000019FF 723A                             jb      short bad13     ; go clean up
18422 00001A01 EB22                             jmp     short dotail
18423                                   ; ---------------------------------------------------------------------------
18424
18425                                   ; reading a sector. do int first, then move things around
18426
18427                                   doread:
18428 00001A03 06                               push    es
18429 00001A04 53                               push    bx
18430 00001A05 1E                               push    ds              ; es = Bios_Code
18431 00001A06 07                               pop     es
18432 00001A07 BB[5201]                         mov     bx, disksector
18433 00001A0A B001                             mov     al, 1
18434 00001A0C 8A5608                           mov     dl, [bp+8]      ; [bp+INT13FRAME.olddx]
18435                                                                   ; get drive number
18436 00001A0F E84EFD                           call    check_wrap      ;
18437                                                                   ; ah = function
18438                                                                   ; al = 1 for single sector
18439                                                                   ; es:bx points to local       buffer
18440                                                                   ; cx, dx are track/sector, head/drive
18441 00001A12 E8D2FD                           call    doint
18442 00001A15 5B                               pop     bx
18443 00001A16 07                               pop     es
18444 00001A17 7222                             jb      short bad13
18445 00001A19 56                               push    si
18446 00001A1A 57                               push    di
18447 00001A1B 89DF                             mov     di, bx
18448 00001A1D BE[5201]                         mov     si, disksector
18449 00001A20 E822FD                           call    move_sector
18450 00001A23 5F                               pop     di
18451 00001A24 5E                               pop     si
18452
18453                                   ; note the fact that we've done 1 more sector
18454
18455                                   dotail:
18456 00001A25 5B                               pop     bx              ; retrieve new dma area
18457 00001A26 80C702                           add     bh, 2           ; advance over sector
18458 00001A29 41                               inc     cx
18459 00001A2A 8A4602                           mov     al, [bp+2]      ; [bp+INT13FRAME.oldax]
18460 00001A2D F8                               clc
18461 00001A2E FEC8                             dec     al
18462 00001A30 7409                             jz      short bad13     ; no more i/o
18463
18464                                   ; see if we wrap around a track or head boundary with starting sector
18465                                   ; we already have the correct head number to pass to check_wrap
18466
18467 00001A32 8A5608                           mov     dl, [bp+8]      ; [bp+INT13FRAME.olddx]
18468 00001A35 E828FD                           call    check_wrap
18469 00001A38 E8ACFD                           call    doint
18470
18471                                   ; we are done. ax has the final code; we throw away what we got before
18472
18473                                   ; M046  -- okay gang. Now we've either terminated our DMA loop,
18474                                   ;       or we've finished. If carry is set now, our only
18475                                   ;       hope for salvation is that it was a read operation
18476                                   ;       and the error code is ECC error. In that case, we'll
18477                                   ;       just pop the registers and go do the old ECC thing.
18478                                   ;       When the DMA error that got us here in the first
18479                                   ;       place occurs, it'll handle it.
18480
```

```
18481                                   bad13:
18482 00001A3B 89EC                              mov     sp, bp
18483 00001A3D 5D                                pop     bp
18484 00001A3E 5B                                pop     bx
18485 00001A3F 5B                                pop     bx
18486 00001A40 59                                pop     cx
18487 00001A41 5A                                pop     dx
18488 00001A42 7203                              jb      short xgoterr13_xxxx ; go handle ECC errors
18489 00001A44 E935FE                            jmp     ret_from_i13   ; non-error exit
18490                                   ; ---------------------------------------------------------------------------
18491
18492                                   xgoterr13_xxxx:
18493 00001A47 E958FE                            jmp     goterr13_xxxx
18494
18495                                   ; ---------------------------------------------------------------------------
18496                                            ; 10/12/2022
18497                                            ;db     0
18498                                   ; ---------------------------------------------------------------------------
18499
18500                                   ;Bios_Code ends
18501
18502                                   ; 16/10/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 IO.SYS)
18503
18504                                   ;--------------------------------------------------------------------------
18505                                   ; MSBIO2.ASM - MSDOS 6.0 - 1991
18506                                   ;--------------------------------------------------------------------------
18507                                   ; 17/03/2019 - Retro DOS v4.0
18508
18509                                            ; 19/10/2022
18510                                   dsk_init:                       ; 2C7h:1742h = 70h:3CB2h
18511 00001A4A 8A26[7500]                        mov     ah, [drvmax]
18512 00001A4E BF[3C05]                          mov     di, dskdrvs
18513 00001A51 1E                                push    ds              ; pass result in es:di
18514 00001A52 07                                pop     es
18515 00001A53 E934EC                            jmp     SetPtrSav
18516
18517                                   ; =============== S U B   R O U TI N E =======================================
18518
18519                                   ;--------------------------------------------------------------------------
18520                                   ; install_bds installs a bds at location es:di into the current linked list of
18521                                   ; bds maintained by this device driver. it places the bds at the end of the
18522                                   ; list. Trashes (at least) ax, bx, di, si
18523                                   ;--------------------------------------------------------------------------
18524
18525                                            ; 26/12/2023 - Retro DOS v5.0
18526                                            ; PCDOS 7.1 IBMBIO.COM - BIOSCODE:1AE0h
18527                                   install_bds:
18528 00001A56 1E                                push    ds              ; save Bios_Data segment
18529 00001A57 BE[1901]                          mov     si, start_bds   ; beginning of chain
18530
18531                                            ; ds:si now points to link to first bds
18532                                            ; assume bds list is non-empty
18533                                   loop_next_bds:
18534 00001A5A C534                              lds     si, [si]        ; [si+BDS.link]
18535                                                                    ; fetch next bds
18536 00001A5C 268A4504                          mov     al, [es:di+4]   ; [es:di+BDS.drivenum]
18537 00001A60 384404                            cmp     [si+4], al      ; does this one       share a physical
18538                                                                    ; drive with new one?
18539 00001A63 7518                              jnz     short next_bds
18540 00001A65 B310                              mov     bl, 10h         ; fi_am_mult
18541                                            ; 26/12/2023
18542 00001A67 26085D3F                          or      [es:di+3Fh], bl
18543                                            ;or     [es:di+23h], bl       ; [es:di+BDS.flags]
18544                                                                    ; set both of them to i_am_mult      if so
18545 00001A6B 085C3F                           or      [si+3Fh], bl
18546                                            ;or     [si+23h], bl    ; [si+BDS.flags]
18547 00001A6E 2680653FDF                        and     byte [es:di+3Fh], 0DFh
18548                                            ;and    byte [es:di+23h], 0DFh ; [es:di+BDS.flags],~fi_own_physical
18549                                                                    ; we don't own it
18550 00001A73 8A5C3F                            mov     bl, [si+3Fh]
18551                                            ;mov    bl, [si+23h]    ; [si+BDS.flags]
18552                                                                    ; determine if changeline available
18553 00001A76 80E302                            and     bl, 2           ; fchangeline
18554 00001A79 26085D3F                          or      [es:di+3Fh], bl
18555                                            ;or     [es:di+23h], bl       ; [es:di+BDS.flags]
18556                                   next_bds:
18557                                            ; 02/09/2023 (PCDOS 7.1)
18558 00001A7D B8FFFF                            mov     ax, 0FFFFh      ; -1
18559 00001A80 3904                              cmp     [si], ax        ; [si+BDS.link],-1
18560                                            ;cmp    word [si], 0FFFFh ; [si+BDS.link],-1
18561                                                                    ; are we at end       of list?
18562 00001A82 75D6                              jnz     short loop_next_bds
18563 00001A84 8C4402                            mov     [si+2], es      ; [si+BDS.link+2],es
18564                                                                    ; install bds
18565 00001A87 893C                              mov     [si], di
18566 00001A89 268905                            mov     [es:di], ax     ; [es:di+BDS.link],-1
18567                                            ;mov    word [es:di], 0FFFFh ; [es:di+BDS.link],-1
18568                                                                    ; set next pointer to null
18569 00001A8C 1F                                pop     ds
18570
18571                                   ; 01/07/2023 - Retro DOS v4.2 (MSDOS 6.21 IO.SYS - BIOSCODE:1785h)
18572                                   ; 16/10/2022 (MSDOS 6.0 Code)
18573
18574                                   ; **** If the new drive has a higher EOT value, we must alter the
18575                                   ;      'eot' variable appropriately.
18576
18577                                            ; 26/12/2023
18578 00001A8D 268A4550                          mov     al, [es:di+50h]       ; [es:di+BDS.rsecpertrack]
18579                                            ; 01/06/2019
18580                                            ;mov    al,[es:di+52]
18581                                            ; 22/07/2023
18582                                            ;mov    al,[es:di+BDS.rsecpertrack]
18583 00001A91 3A06[2C01]                        cmp     al,[eot]
18584 00001A95 7603                              jbe     short _eot_ok
18585 00001A97 A2[2C01]                          mov     [eot],al
18586                                   _eot_ok:
18587 00001A9A C3                                retn
18588
18589                                   ; ---------------------------------------------------------------------------
18590
18591                                   ; 17/10/2022
18592                                   ;DRVLET     equ drvlet - DOSBIOSEG_2C7h
18593                                   ;SNGMSG     equ sngmsg - DOSBIOSEG_2C7h
18594                                   ; 09/12/2022
18595                                   DRVLET equ drvlet
18596                                   SNGMSG equ sngmsg
18597
18598                                   ; 16/10/2022
18599
18600                                   ;--------------------------------------------------------------------------
18601                                   ;  ask to swap the disk in drive a:
18602                                   ;   es:di -> bds
18603                                   ;   ds -> Bios_Data
18604                                   ;--------------------------------------------------------------------------
```

```
18605
18606                                   ; 26/12/2023 - Retro DOS v5.0
18607
18608                                   ; 19/10/2022
18609 00001A9B F606[1208]01       swpdsk:       test   byte [IsWin386], 1
18610                                   ;test  ds:IsWin386, 1 ; Is win386 present?
18611 00001AA0 7405                     jz     short no_win386       ; no, skip SetFocus
18612
18613                                   ; set focus to the correct VM
18614                                   ;call  far ptr 70h:813h ; PCDOS 7.1 IBMBIO.COM BIOSCODE:1B2Ch
18615                                   ;;call  far 70h:8D1h   ; MSDOS 6.21 IO.SYS BIOSCODE:179Ah
18616                                   ; 17/10/2022
18617 00001AA2 9A[1308]7000             call   DOSBIOSSEG:V86_Crit_SetFocus ; BIOSDATA:V86_Crit_SetFocus
18618                                   ;call  far ptr V86_Crit_SetFocus ; call far 70h:8D1h
18619                                                  ; call far KERNEL_SEGMENT:V86_Crit_SetFocus
18620                             no_win386:
18621 00001AA7 51                       push   cx
18622 00001AA8 52                       push   dx
18623 00001AA9 268A5505                 mov    dl, [es:di+5]  ; [es:di+BDS.drivelet]
18624                                                  ; get the drive       letter
18625
18626                             ; WARNING : next two instructions assume that if the new disk is for drive B
18627                             ;           then existing dsk is drive A & vice versa
18628
18629 00001AAD 88D6                       mov    dh, dl
18630 00001AAF 80F601                   xor    dh, 1
18631 00001AB2 29C9                     sub    cx, cx         ; nobody has handled swap disk
18632 00001AB4 B8004A                   mov    ax, 4A00h      ; multMULT<<8)|multMULTSWPDSK
18633                                                  ; broadcast code for swap disk
18634                                                  ; Broadcast it
18635 00001AB7 CD2F                     int    2Fh
18636 00001AB9 41                       inc    cx             ; cx == -1 ?
18637 00001ABA 741E                     jz     short swpdsk9  ; somebody has handled it
18638
18639                             ; using a different drive in a one drive system so request the user change disks
18640
18641 00001ABC 80C241                   add    dl, 'A'
18642                                   ; 17/10/2022
18643 00001ABF 2E8816[F91A]             mov    [cs:DRVLET], dl       ; "A: and press any key when ready\r\n\n"
18644                                   ; 16/10/2022
18645                                   ;;mov  byte [cs:drvlet], dl
18646                                   ;mov   byte ptr cs:17E4h, dl ; [cs:drvlet]
18647                                                  ; 0070h:3D54h =       2C7h:17E4h
18648 00001AC4 BE[DD1A]                mov    si, SNGMSG     ; "\r\nInsert diskette for drive "
18649                                   ;mov   si, 17C8h      ; sngmsg
18650                                                  ; 0070h:3D38h =       2C7h:17C8h
18651 00001AC7 53                       push   bx
18652 00001AC8 2E                       cs
18653 00001AC9 AC                       lodsb                ; get the next character of the message
18654                                   ;lods  byte ptr cs:[si]
18655                             wrmsg_loop:
18656 00001ACA CD29                     int    29h            ; DOS 2+ internal - FAST PUTCHAR
18657                                                  ; AL = character to display
18658 00001ACC 2E                       cs
18659 00001ACD AC                       lodsb
18660                                   ;lods  byte ptr cs:[si] ; cs lodsb
18661                                                  ; get the next character of the      message
18662 00001ACE 08C0                     or     al, al
18663 00001AD0 75F8                     jnz    short wrmsg_loop
18664 00001AD2 E833E7                   call   con_flush      ; flush out keyboard queue
18665                                                  ; call rom-bios
18666 00001AD5 30E4                     xor    ah, ah
18667 00001AD7 CD16                     int    16h            ; KEYBOARD - READ CHAR FROM BUFFER, WAIT IF EMPTY
18668                                                  ; Return: AH = scan code, AL = character
18669 00001AD9 5B                       pop    bx
18670                             swpdsk9:
18671 00001ADA 5A                       pop    dx
18672 00001ADB 59                       pop    cx
18673 00001ADC C3                       retn
18674
18675                             ; ---------------------------------------------------------------------------
18676
18677                             ; 16/10/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 IO.SYS)
18678
18679                             ;--------------------------------------------------------
18680                             ; include msbio.cl2 (MSDOS 6.0, 1991)
18681                             ;--------------------------------------------------------
18682                             ; (MSDOS 6.21 IO.SYS BIOSCODE:17D5h)
18683                             ;--------------------------------------------------------
18684                             ; 17/03/2019 - Retro DOS v4.0
18685                             ; 26/12/2023 - Retro DOS v5.0
18686
18687                                   ; MSDOS 5.0 IO.SYS offset 0070h:3D38h or 02C7h:17C8h
18688                                   ; PCDOS 7.1 IBMBIO.COM - BIOSCODE:1B67h
18689 00001ADD 0D0A                 sngmsg:       db 0Dh,0Ah
18690 00001ADF 496E73657274206469-              db 'Insert diskette for drive '
18690 00001AE8 736B6574746520666F-
18690 00001AF1 7220647269766520
18691
18692                                   ; MSDOS 5.0 IO.SYS offset 0070h:3D54h or 02C7h:17E4h
18693                                   ; PCDOS 7.1 IBMBIO.COM - BIOSCODE:1B83h
18694 00001AF9 413A20616E64207072-  drvlet:       db 'A: and press any key when ready',0Dh,0Ah
18694 00001B02 65737320616E79206B-
18694 00001B0B 6579207768656E2072-
18694 00001B14 656164790D0A
18695 00001B1A 0A00                             db 0Ah,0
18696
18697                             ; =============== S U B   R O U T I N E =======================================
18698
18699                             ;---------------------------------------------------------------------------
18700                             ; input : es:di points to current bds for drive.
18701                             ; return : zero set if no open files
18702                             ;      zero reset if open files
18703                             ;---------------------------------------------------------------------------
18704
18705                                   ; 26/12/2023 - Retro DOS v5.0
18706                             chkopcnt:
18707 00001B1C 26837D3C00               cmp     word [es:di+3Ch], 0
18708                                   ;cmp   word [es:di+20h], 0 ; [es:di+BDS.opcnt]
18709 00001B21 C3                       retn
18710
18711                             ; =============== S U B   R O U T I N E =======================================
18712
18713                             ;---------------------------------------------------------------------------
18714                             ; at media check time, we need to really get down and check what the change is.
18715                             ; this is guaranteed to be expensive.
18716                             ;
18717                             ;   es:di -> bds, ds -> Bios_Data
18718                             ;---------------------------------------------------------------------------
18719
18720                                   ; 26/12/2023 - Retro DOS v5.0
18721                                   ; PCDOS 7.1 IBMBIO.COM - BIOSCODE:1BA6h
18722                             mediacheck:
18723 00001B22 E852EE                   call   checksingle    ; make sure correct disk is in place
```

```
18724 00001B25 31F6                             xor     si, si
18725 00001B27 E86101                           call    haschange
18726 00001B2A 742F                             jz      short mediaret
18727                                           ; 26/12/2023
18728                                           ;test  byte [es:di+3Fh], 40h ; [es:di+BDS.flags], fchanged ; 40h
18729 00001B2C E85001                           call    checkromchange
18730 00001B2F 752B                             jnz     short mediadovolid
18731 00001B31 50                               push    ax
18732 00001B32 52                               push    dx
18733 00001B33 268A5504                         mov     dl, [es:di+4]  ; [es:di+BDS.drivenum]
18734                                                                  ; set logical drive number
18735 00001B37 B416                             mov     ah, 16h
18736 00001B39 CD13                             int     13h            ; DISK - FLOPPY       DISK - CHANGE OF DISK STATUS (AT,XT2,XT286,CONV,PS)
18737                                                                  ; DL = drive to       check
18738                                                                  ; Return: AH = disk change status
18739 00001B3B 5A                               pop     dx
18740 00001B3C 58                               pop     ax
18741 00001B3D 721D                             jb      short mediadovolid
18742 00001B3F BE0100                           mov     si, 1          ; signal no change
18743
18744                                   ; there are some drives with changeline that "lose" the changeline indication
18745                                   ; if a different drive is accessed after the current one. in order to avoid
18746                                   ; missing a media change, we return an "i don't know" to dos if the changeline
18747                                   ; is not active and we are accessing a different drive from the last one.
18748                                   ; if we are accessing the same drive, then we can safely rely on the changeline
18749                                   ; status.
18750                                             ; 19/10/2022
18751 00001B42 8A1E[1E01]                       mov     bl, [tim_drv]  ; get last drive accessed
18752 00001B46 26385D04                         cmp     [es:di+4], bl  ; [es:di+BDS.drivenum]
18753                                                                  ; (If the last drive accessed is not current drive
18754                                                                  ; media change status may be incorrect.      So,
18755                                                                  ; "I don't now" will be returned even if it is indicated
18756                                                                  ; as media is not changed.)
18757 00001B4A 740F                             jz      short mediaret ; (same drive,
18758                                                                  ; media changeline indication is reliable)
18759
18760                                   ; do the 2 second twiddle. if time >= 2 seconds, do a volid check.
18761                                   ; otherwise return "i don't know" (strictly speaking, we should return a
18762                                   ; "not changed" here since the 2 second test said no change.)
18763
18764 00001B4C 50                                     push    ax
18765 00001B4D 51                                     push    cx
18766 00001B4E 52                                     push    dx
18767 00001B4F E8D8EA                                 call    Check_Time_Of_Access
18768 00001B52 5A                                     pop     dx
18769 00001B53 59                                     pop     cx
18770 00001B54 58                                     pop     ax
18771 00001B55 09F6                                   or      si, si
18772 00001B57 7403                                   jz      short mediadovolid ; check_time      says ">= 2 secs       passed"
18773                                                                            ; (volume id will be checked)
18774 00001B59 31F6                                   xor     si, si         ; return "i don't know"
18775                                 mediaret:
18776 00001B5B C3                                     retn
18777                                 ; ---------------------------------------------------------------------------
18778
18779                                 ; somehow the media was changed. look at vid to see. we do not look at fat
18780                                 ; because this may be different since we only set medbyt when doing a read
18781                                 ; or write.
18782
18783                                 mediadovolid:
18784 00001B5C E877EB                                 call    GetBp          ; build a new bpb in current bds
18785 00001B5F 72FA                                   jb      short mediaret
18786 00001B61 E82D00                                 call    check_vid
18787 00001B64 73F5                                   jnb     short mediaret
18788 00001B66 E940F2                                 jmp     maperror       ; fix up al for       return to dos
18789                                 ; ---------------------------------------------------------------------------
18790
18791                                 ; simple, quick check of latched change. if no indication, then return
18792                                 ; otherwise do expensive check. if the expensive test fails, pop off the
18793                                 ; return and set al = 15 (for invalid media change) which will be returned to
18794                                 ; dos.
18795                                 ;
18796                                 ; for dos 3.3, this will work only for the drive that has changeline.
18797
18798                                 ;   call with es:di -> bds, ds -> Bios_Data
18799                                 ;   ***** warning:  this routine will return one level up on the stack
18800                                 ;                   if an error occurs!
18801
18802                                 checklatchio:
18803
18804                                 ; if returning fake bpb then assume the disk has not changed
18805
18806                                             ; 26/12/2023
18807                                             ;cmp   word [es:di+3Ch], 0 ; [es:di+BDS.opcnt]
18808 00001B69 E8B0FF                           call    chkopcnt
18809 00001B6C 741B                             jz      short checkret ; done if zero
18810
18811                                 ; check for past rom indications. if no rom change indicated, then return ok.
18812
18813                                             ; 26/12/2023
18814                                             ;test  word [es:di+3Fh], 40h
18815                                             ;                       ; test [es:di+BDS.flags], fchanged ; 40h
18816 00001B6E E80E01                           call    checkromchange
18817 00001B71 7416                             jz      short checkret
18818
18819                                 ; we now see that a change line has been seen in the past. let's do the
18820                                 ; expensive verification.
18821
18822 00001B73 E860EB                           call    GetBp          ; build bpb in current bds
18823 00001B76 720F                             jb      short ret_no_error_map ; getbp has already called maperror
18824 00001B78 E81600                           call    check_vid
18825 00001B7B 7207                             jb      short checklatchret ; disk error trying      to read in.
18826 00001B7D 09F6                             or      si, si         ; is changed for sure?
18827 00001B7F 7908                             jns     short checkret
18828 00001B81 E88F00                           call    returnvid
18829                                 checklatchret:
18830 00001B84 E822F2                           call    maperror       ; fix up al for       return to dos
18831                                 ret_no_error_map:
18832 00001B87 F9                               stc
18833 00001B88 5E                               pop     si             ; pop off return address
18834                                 checkret:
18835 00001B89 C3                               retn
18836                                 ; ---------------------------------------------------------------------------
18837
18838                                 ; check the fat and the vid. return in di -1 or 0. return with carry set
18839                                 ; only if there was a disk error. return that error code in ax.
18840                                 ;
18841                                 ;   called with es:di -> bds, ds -> Bios_Data
18842
18843                                 checkfatvid:
18844 00001B8A E8D101                           call    fat_check      ; check the fat and the vid
18845 00001B8D 09F6                             or      si, si
18846 00001B8F 7835                             js      short changed_drv
18847
```

```
                                ; the fat was the same. fall into check_vid and check volume id.
                                
                                        ; 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
                                        ;pop    ax
00001BE4 7504                           jnz     short ext_changed ; not equal/same
00001BE6 31F6                           xor     si, si          ; 0 ; don't know
00001BE8 EBD8                           jmp     short vid_no_changed ; reset the flag
                                        ;;;
                                %else
                                        ; 02/09/2023
                                        xor     si, si ; 0
                                        cmp     ax, [es:di+57h]         ; [di+BDS.vol_serial]
                                        jnz     short ext_changed
                                        mov     ax, [disksector+29h] ; [DiskSector+EXT_BOOT.SERIAL+2]
                                        cmp     ax, [es:di+59h]         ; [di+BDS.vol_serial+2]
                                        jnz     short ext_changed
                                        ;xor    si, si          ; 0
                                                                ; don't know
                                        pop     ax
                                        jmp     short vid_no_changed
                                                                ; reset the flag
                                %endif

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

                                ext_changed:
                                        ; 26/12/2023
                                        ;pop    ax
                                        ; 02/09/2023
                                        ;dec    si ; mov si, 0FFFFh ; -1
00001BEA BEFFFF                         mov     si, 0FFFFh      ; -1
                                                                ; disk changed!
                                        ; 12/12/2022
                                        ; ('changed_drv' clears cf)
                                        ;clc
00001BED EBD7                           jmp     short changed_drv

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

                                ; at i/o time, we detected the error. now we need to determine whether the
                                ; media was truly changed or not. we return normally if media change unknown.
                                ; and we pop off the call and jmp to harderr if we see an error.
                                ;
                                ; es:di -> bds

                                checkio:
00001BEF 80FC06                         cmp     ah, 6
00001BF2 75D1                           jnz     short checkfatret
00001BF4 E825FF                         call    chkopcnt
00001BF7 74CC                           jz      short checkfatret
00001BF9 E8DAEA                         call    GetBp
00001BFC 7212                           jb      short no_error_map
00001BFE E889FF                         call    checkfatvid
00001C01 7209                           jb      short checkioret ; disk      error trying to       read in.
00001C03 09F6                           or      si, si          ; is changed for sure?
00001C05 7802                           js      short checkioerr ; yes changed
00001C07 45                             inc     bp              ; allow a retry
00001C08 C3                             retn
                                ; ---------------------------------------------------------------------------

                                checkioerr:
00001C09 E80700                         call    returnvid

                                checkioret:
00001C0C F9                             stc                     ; make sure carry gets passed through
00001C0D E955F1                         jmp     harderr
                                ; ---------------------------------------------------------------------------

                                no_error_map:
00001C10 E955F1                         jmp     harderr2

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

                                ; return vid sets up the vid for a return to dos.
                                ;  es:di -> bds, returns pointer in packet to bds_volid
                                ;  **** trashes si! ****

                                returnvid:
00001C13 BE1600                         mov     si, 22          ; extra
                                                                ; offset into pointer to return      value
00001C16 E80700                         call    vid_into_packet
00001C19 B406                           mov     ah, 6
00001C1B F9                             stc
00001C1C C3                             retn

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

                                ; moves the pointer to the volid for the drive into the original request packet
                                ; no attempt is made to preserve registers.
                                ;
                                ; assumes es:di -> bds
                                ; **trashes si**

                                media_set_vid:
00001C1D BE0F00                         mov     si, 15          ; trans+1
                                                                ; return the value here      in packet

                                        ; fall into vid_into_packet

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

                                ; return pointer to vid in bds at es:di in packet[si]

                                        ; 14/04/2024 - Retro DOS v5.0 (PCDOS 7.1 IBMBIO.COM)
                                        ; 19/10/2022
                                vid_into_packet:
00001C20 1E                             push    ds              ; return pointer to vid      in bds at es:di       in packet[si]
00001C21 C51E[1200]                     lds     bx, [ptrsav]
                                        ;add    di, 75          ; BDS.volid
                                        ; 14/04/2024
00001C25 83C77D                         add     di, 125 ; (PCDOS 7.1)
00001C28 8938                           mov     [bx+si], di
                                        ;sub    di, 75          ; BDS.volid
00001C2A 83EF7D                         sub     di, 125
00001C2D 8C4002                         mov     [bx+si+2], es
00001C30 1F                             pop     ds
                                dofloppy:   ; 18/12/2022
00001C31 C3                             retn

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

                                ;---------------------------------------------------------------------------
                                ;   hidensity - examine a drive/media descriptor to set the media type. if
                                ;   the media descriptor is not f9 (not 96tpi or 3 1/2), we return and let the
                                ;   caller do the rest. otherwise, we pop off the return and jump to the tail
```

```
                                ;   of getbp. for 3.5" media, we just return.
                                ;
                                ;   inputs: es:di point to correct bds for this drive
                                ;           ah has media byte
                                ;
                                ;   outputs:       carry clear
                                ;               no registers modified
                                ;           carry set
                                ;               al = sectors/fat
                                ;               bh = number of root directory entries
                                ;               bl = sectors per track
                                ;               cx = number of sectors
                                ;               dh = sectors per allocation unit
                                ;               dl = number of heads
                                ;
                                ;--------------------------------------------------------------------------

                                            ; 26/12/2023 - Retro DOS v5.0
                                hidensity:

                                ; check for correct drive

                                            ; 26/12/2023
00001C32 26F6453F02                         test    byte [es:di+3Fh], 2 ; is it special?
                                            ; 12/12/2022
                                            ;test  byte [es:di+23h], 2
                                            ;;test word [es:di+23h], 2 ; is it special?
                                                                    ; [es:di+BDS.flags], fchangeline
00001C37 74F8                               jz      short dofloppy ; no, do normal floppy test

                                ; we have a media byte that is pretty complex. examine drive information
                                ; table to see what kind it is.

                                            ; 26/12/2023
00001C39 26807D3E02                         cmp     byte [es:di+3Eh], 2 ; is it single-media?
                                            ;cmp    byte [es:di+22h], 2 ; is it single-media?
00001C3E 74F1                               jz      short dofloppy ; [es:di+BDS.formfactor], ffSmall
                                                                    ; yes, use fatid...
                                ; 96 tpi drive?
00001C40 80FCF9                             cmp     ah, 0F9h
00001C43 75EC                               jnz     short dofloppy

                                ;------ If formfactor of drive = ffother or ff288 it has to be
                                ;------ a 720K diskette

                                            ; 02/09/2023 (PCDOS 7.1)
                                            ; 26/12/2023
00001C45 268A453E                           mov     al, [es:di+3Eh] ; [es:di+BDS.formfactor]
                                            ;mov    al, [es:di+22h]       ; [es:di+BDS.formfactor]
00001C49 3C07                               cmp     al, 7
                                            ;cmp    byte [es:di+22h], 7 ; [es:di+BDS.formfactor]
                                                                    ; ffOther
00001C4B 7413                               jz      short Is720K
00001C4D 3C09                               cmp     al, 9
                                            ;cmp    byte [es:di+22h], 9 ; [es:di+BDS.formfactor]
                                                                    ; ff288
00001C4F 740F                               jz      short Is720K
00001C51 B007                               mov     al, 7           ; seven sectors / fat
00001C53 BB0FE0                             mov     bx, 57359       ; 224*256+0Fh
                                                                    ; 224 root dir entries
                                                                    ; & 0Fh sector max
00001C56 B96009                             mov     cx, 2400        ; 80*15*2
                                                                    ; 80 tracks, 15 sectors/track,
                                                                    ; 2 sides
                                            ; 02/09/2023
00001C59 5A                                 pop     dx              ; pop off return address
00001C5A BA0201                             mov     dx, 258         ; 1*256+2
                                                                    ; sectors/allocation unit
                                                                    ; & head max
                                            ;add    sp, 2           ; pop off return address
00001C5D E9EAEA                             jmp     Has1            ; return to tail of getbp
                                ; ---------------------------------------------------------------------------

                                Is720K:
                                            ; 02/09/2023
00001C60 5B                                 pop     bx              ; pop off return address
                                            ;add    sp, 2           ; pop off return address
00001C61 E9A9EA                             jmp     Has720K         ; return to 720K code
                                ; ---------------------------------------------------------------------------

                                            ; 18/12/2022
                                ;dofloppy:
                                            ;retn

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

                                ; 16/10/2022

                                ;--------------------------------------------------------------------------
                                ; set_changed_dl - sets flag bits according to bits set in bx.
                                ;             essentially used to indicate changeline, or format.
                                ;
                                ;   inputs: dl contains physical drive number
                                ;           bx contains bits to set in the flag field in the bdss
                                ;   outputs:       none
                                ;   registers modified: flags
                                ;
                                ;   called from int13 hooker.  Must preserve ALL registers!!!
                                ;
                                ; in the virtual drive system we *must* flag the other drives as being changed
                                ;--------------------------------------------------------------------------

                                            ; 26/12/2023 - Retro DOS v5.0
                                set_changed_dl:
00001C64 06                                 push    es
00001C65 57                                 push    di
                                            ;les    di, ds:start_bds
                                            ; 19/10/2022
00001C66 C43E[1901]                         les     di, [start_bds]

                                ; note: we assume that the list is non-empty

                                scan_bds:
00001C6A 26385504                           cmp     [es:di+4], dl  ; [es:di+BDS.drivenum]
00001C6E 7504                               jnz     short get_next_bds

                                ; someone may complain, but this *always* must be done when a disk change is
                                ; noted. there are *no* other compromising circumstances.

                                            ; 26/12/2023
00001C70 26095D3F                           or      [es:di+3Fh], bx       ; [es:di+BDS.flags]
                                            ;or     [es:di+23h], bx       ; [es:di+BDS.flags]
                                                                    ; signal change       on other drive
                                get_next_bds:
```

```
19220 00001C74 26C43D                          les     di, [es:di]     ; [es:di+BDS.link]
19221                                                                  ; go to next bds
19222 00001C77 83FFFF                          cmp     di, 0FFFFh
19223 00001C7A 75EE                            jnz     short scan_bds ; loop unless we hit end of chain
19224 00001C7C 5F                              pop     di
19225 00001C7D 07                              pop     es
19226 00001C7E C3                              retn
19227
19228                                  ; =============== S U B   R O U T I N E =======================================
19229
19230                                  ;---------------------------------------------------------------------------
19231                                  ; checkromchange - see if external program has diddled rom change line.
19232                                  ;
19233                                  ;   inputs: es:di points to current bds.
19234                                  ;   outputs:       zero set - no change
19235                                  ;                  zero reset - change
19236                                  ;   registers modified: none
19237                                  ;---------------------------------------------------------------------------
19238
19239                                                  ; 26/12/2023 - Retro DOS v5.0
19240                                  checkromchange:
19241                                                  ;test   word [es:di+BDS.flags], fchanged ; 40h
19242                                                  ; 26/12/2023
19243 00001C7F 26F6453F40                              test    byte [es:di+3Fh], 40h
19244                                                  ; 10/12/2022
19245                                                  ;test   byte [es:di+23h], 40h
19246                                                  ;;test  word [es:di+23h], 40h ; [es:di+BDS.flags]
19247                                                                         ; fchanged
19248 00001C84 C3                                      retn
19249
19250                                  ; =============== S U B   R O U T I N E =======================================
19251
19252                                  ;---------------------------------------------------------------------------
19253                                  ; resetchanged - restore value of change line
19254                                  ;
19255                                  ;   inputs: es:di points to current bds
19256                                  ;   outputs:       none
19257                                  ;   registers modified: none
19258                                  ;---------------------------------------------------------------------------
19259
19260                                                  ; 26/12/2023 - Retro DOS v5.0
19261                                  resetchanged:
19262                                                  ;and    word [es:di+BDS.flags], ~fchanged ; 0FFBFh
19263                                                  ; 26/12/2023
19264 00001C85 2680653FBF                              and     byte [es:di+3Fh], 0BFh
19265                                                  ; 10/12/2022
19266                                                  ;and    byte [es:di+23h], 0BFh
19267                                                  ;;and   word [es:di+23h], 0FFBFh ; [es:di+BDS.flags]
19268                                                                         ; ~fchanged
19269 00001C8A C3                                      retn
19270
19271                                  ; =============== S U B   R O U T I N E =======================================
19272
19273                                  ;---------------------------------------------------------------------------
19274                                  ; haschange - see if drive can supply change line
19275                                  ;
19276                                  ;   inputs: es:di points to current bds
19277                                  ;   outputs:       zero set - no change line available
19278                                  ;                  zero reset - change line available
19279                                  ;   registers modified: none
19280                                  ;---------------------------------------------------------------------------
19281
19282                                                  ; 26/12/2023 - Retro DOS v5.0
19283                                  haschange:
19284                                                  ;test   word [es:di+BDS.flags], fchangeline ; 2
19285                                                  ; 26/12/2023
19286 00001C8B 26F6453F02                              test    byte [es:di+3Fh], 2
19287                                                  ; 10/12/2022
19288                                                  ;test   byte [es:di+23h], 2
19289                                                  ;;test  word [es:di+23h], 2 ; [es:di+BDS.flags]
19290                                                                         ; fchangeline
19291 00001C90 C3                                      retn
19292
19293                                  ; ---------------------------------------------------------------------------
19294
19295                                  ; 16/10/2022
19296
19297                                  ;---------------------------------------------------------------------------
19298                                  ; set_volume_id      -    main routine, calls other routines.
19299                                  ; read_volume_id     -    read the volume id and tells if it has been changed.
19300                                  ; transfer_volume_id -    copy the volume id from tmp to special drive.
19301                                  ; check_volume_id    -    compare volume id in tmp area with one expected for drive.
19302                                  ; fat_check          -    see of the fatid has changed in the specified drive.
19303                                  ;---------------------------------------------------------------------------
19304
19305                                  ; set_volume_id
19306                                  ;   if drive has changeline support, read in and set the volume_id
19307                                  ; and the last fat_id byte. if no change line support then do nothing.
19308                                  ;
19309                                  ;   on entry:
19310                                  ;   es:di points to the bds for this disk.
19311                                  ;   ah contains media byte
19312                                  ;
19313                                  ;   on exit:
19314                                  ;   carry clear:
19315                                  ;      successful call
19316                                  ;   carry set
19317                                  ;      error and ax has error code
19318
19319                                  set_volume_id:
19320 00001C91 52                                      push    dx              ; save registers
19321 00001C92 50                                      push    ax
19322 00001C93 E8F5FF                                  call    haschange       ; does drive have changeline support?
19323 00001C96 740B                                    jz      short setvret   ; no, get out
19324 00001C98 E81000                                  call    read_volume_id
19325 00001C9B 7209                                    jb      short seterr
19326 00001C9D E8A900                                  call    transfer_volume_id ; copy the volume id      to special drive
19327 00001CA0 E8E2FF                                  call    resetchanged    ; restore value       of change line
19328                                  setvret:
19329                                                  ; 10/12/2022
19330                                                  ; cf = 0
19331                                                  ;clc                    ; no error, clear carry flag
19332 00001CA3 58                                      pop     ax              ; restore registers
19333 00001CA4 5A                                      pop     dx
19334 00001CA5 C3                                      retn
19335                                  ; ---------------------------------------------------------------------------
19336
19337                                  seterr:
19338 00001CA6 5A                                      pop     dx              ; pop stack but don't overwrite ax
19339 00001CA7 5A                                      pop     dx              ; restore dx
19340 00001CA8 C3                                      retn
19341                                  ; ---------------------------------------------------------------------------
19342 00001CA9 0000                    root_sec:       dw 0                    ; root sector #
19343
```

```
                                ; 16/10/2022
                                ;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
                        ;over  the  system  initialization  after  the  OEM  bios   has
                        ;performed any  initialization  it  needs  to  do.   Control is
                        ;transfered with a long jump to the external  variable  SYSINIT
                        ;
                        ;
                        ;   The OEM  has  the  following  variables declared external:
                        ;
                        ;   CURRENT_DOS_LOCATION    WORD
                        ;
                        ;This word  contains  the  segment  number of the DOS before it
                        ;is relocated.  The OEM bios must set this value.
                        ;
                        ;   FINAL_DOS_LOCATION      WORD
                        ;
                        ;This word contains the segment number of the DOS after SYSINIT
                        ;moves it.  The OEM bios must set this value.
                        ;
                        ;   DEVICE_LIST             DWORD
                        ;
                        ;This  double  word  pointer  points  to  the  linked  list  of
                        ;character and block device drivers.  The  OEM  must  set  this
                        ;value.
                        ;
                        ;   MEMORY_SIZE             WORD
                        ;
                        ;This word  contains  the  number  of  RAM  paragraphs.  If the
                        ;bios doesn't set  this  variable  SYSINIT  will  automatically
                        ;calculate it.   NOTE:  systems with PARITY checked memory must
                        ;size memory in the BIOS.  SYSINITs method is to  write  memory
                        ;and read it back until it gets a mismatch.
                        ;
                        ;   DEFAULT_DRIVE           BYTE
                        ;
                        ;This is  the initial default drive when the system first comes
                        ;up.  drive a=0, drive b=1,  etc.   If  the  bios  doesn't  set
                        ;it then drive a is assumed.
                        ;
                        ;   BUFFERS                 BYTE
                        ;
                        ;This is  the  default  number of buffers for the system.  This
                        ;value may be overridden by the user in  the  CONFIG.SYS  file.
                        ;It is DBed to 2 in SYSINIT it should be greater than 1.
                        ;
                        ;   FILES                   BYTE
                        ;
                        ;This is  the  default  number  of  files for the system.  This
                        ;value may be overridden by the user in  the  CONFIG.SYS  file.
                        ;It is  DBed  to  8 in SYSINIT, values less than 5 are ignored.
                        ;
                        ;   SYSINIT                 FAR
                        ;
                        ;The entry  point  of  the  SYSINIT  module.  OEM BIOS jumps to
                        ;this label at the end of its INIT code.
                        ;
                        ;   The OEM  has  the  following  variables declared public:
                        ;
                        ;   RE_INIT                 FAR
                        ;
                        ;This is an entry point which allows the BIOS to do some INIT
                        ;work  after  the  DOS is initialized.  ALL REGISTERS MUST BE
                        ;PRESERVED.  On entry DS points to the first available memory
                        ;(after  the DOS).  DS:0 points to a 100H byte program header
                        ;prefix which represents  the  "program"  currently  running.
                        ;This  program  should  be  thought  of  as  the OEM BIOS and
                        ;SYSINIT taken together.  This is not  a  normal  program  in
                        ;that  no  memory  is  allocated to it, it is running in free
                        ;memory.
                        ;NOTES:
                        ;     At the time this routine is called SYSINIT occupies the
                        ;highest 10K of memory ("highest" is determined by the  value
                        ;of the MEMORY_SIZE variable), DO NOT DO WRITES THERE.
                        ;     Since this is called AFTER DOS is initialized, you can
                        ;make system calls.  This also implies that the code for this
                        ;routine    CANNOT   be   thrown   away   by   use   of   the
                        ;FINAL_DOS_LOCATION since the DOS has already been moved.
                        ;     If you don't want  anything done just set this to point
                        ;at a FAR RET instruction.

                        ; --------------------------------------------------------------------------
                        ; TITLE   BIOS SYSTEM INITIALIZATION
                        ; --------------------------------------------------------------------------

                        ;include version.inc
                        ; --------------------------------------------------------------------------

                        ;FALSE   EQU     0
                        ;TRUE    EQU     0FFFFh

                        ;IBMVER         EQU     TRUE
                        ;IBMCOPYRIGHT EQU   FALSE
                        ;STACKSW    EQU        TRUE             ;Include Switchable Hardware Stacks
                        ;IBMJAPVER  EQU     FALSE        ; If TRUE set KANJI true also
                        ;MSVER      EQU     FALSE
                        ;ALTVECT    EQU     FALSE        ; Switch to build ALTVECT version
                        ;KANJI      EQU     FALSE

                        ;(MSDOS 6.0, versiona.inc, 1991)
                        ; --------------------------------------------------------------------------
                        ;MAJOR_VERSION  EQU       6
                        ;;MINOR_VERSION    EQU    0       ;6.00
                        ;MINOR_VERSION  EQU       21      ;6.21  ; 21/03/2019 - Retro DOS v4.0

                        ; 22/10/2022 - Retro DOS v4.0 (Modified MSDOS 5.0)
                        ; --------------------------------------------------------------------------
                        ;MAJOR_VERSION   EQU      5
                        ;MINOR_VERSION   EQU      0

                        ; 01/01/2023 - Retro DOS v4.2 (Modified MSDOS 6.21)
                        ;MAJOR_VERSION   EQU      6
                        ;MINOR_VERSION   EQU      22

                        ; 21/02/2024 - Retro DOS v5.0 (Modified PCDOS 7.1)
                        MAJOR_VERSION   EQU       7
                        MINOR_VERSION   EQU       10

                        expected_version equ      (MINOR_VERSION<<8)+MAJOR_VERSION

                        ;DOSREVNM equ      00000000b       ; m037 - bits 0-2 = revision number of DOS
                                                           ; currently 0.
                        DOSREVNM equ       00000111b      ; [[[ 7 for Retro DOS v4.0 ]]] (21/03/2019)
                        DOSINROM equ       00001000B       ; bit 3 of ver flags returned in BH
                        DOSINHMA equ       00010000B       ; bit 4 of ver flags
```

```
19840
19841                                 ;       if1
19842                                 ;       %OUT  ... for DOS Version 5.00 ...
19843                                 ;       endif
19844
19845                                         ;******************************
19846                                         ;Each assembler program should:
19847                                         ;  mov ah,030h                  ;DOS Get Version function
19848                                         ;  int 021h                     ;Version ret. in AX,minor version first
19849                                         ;  cmp ax,expected_version      ;ALL utilities should check for an
19850                                         ;  jne error_handler           ; EXACT version match.
19851                                         ;******************************
19852
19853                                 ; --------------------------------------------------------------------
19854                                 ; device definitions
19855
19856                                 ;Attribute bit masks
19857                                 DEVTYP  EQU     8000h           ;Bit 15 - 1  if Char, 0 if block
19858                                 DEVIOCTL EQU    4000h           ;Bit 14 - CONTROL mode bit
19859                                 ISFATBYDEV EQU  2000h           ;Bit 13 - Device uses FAT ID bytes, comp media.
19860                                 ISCIN   EQU     0001h           ;Bit 0 - This device is the console input.
19861                                 ISCOUT  EQU     0002h           ;Bit 1 - This device is the console output.
19862                                 ISNULL  EQU     0004h           ;Bit 2 - This device is the null device.
19863                                 ISCLOCK EQU     0008h           ;Bit 3 - This device is the clock device.
19864                                 ISIBM   EQU     0010h           ;Bit 4 - This device is special
19865
19866                                 ; The device table list has the form:
19867                                 struc       SYSDEV
19868 00000000 ????????            .NEXT:              resd 1          ;Pointer to next device header
19869 00000004 ????                .ATT:               resw 1          ;Attributes of the device
19870 00000006 ????                .STRAT:             resw 1          ;Strategy entry point
19871 00000008 ????                .INT:               resw 1          ;Interrupt entry point
19872 0000000A ??????????????      .NAME:              resb 8          ;Name of device (only first byte used for block)
19873                               .size:
19874                               endstruc
19875
19876                               ;Static Reguest Header
19877                               struc       SRHEAD
19878 00000000 ??                  .REQLEN:    resb 1          ;Length in bytes of request block
19879 00000001 ??                  .REQUNIT:   resb 1          ;Device unit number
19880 00000002 ??                  .REQFUNC:   resb 1          ;Type of request
19881 00000003 ????                .REQSTAT:   resw 1          ;Status Word
19882 00000005 ??????????????                  resb 8          ;Reserved for queue links
19883                               .size:
19884                               endstruc
19885
19886                               ;Status word masks
19887                               STERR   EQU     8000H           ;Bit 15 - Error
19888                               STBUI   EQU     0200H           ;Bit 9 - Buisy
19889                               STDON   EQU     0100H           ;Bit 8 - Done
19890                               STECODE EQU     00FFH           ;Error code
19891                               WRECODE EQU     0
19892
19893                               ;Function codes
19894                               DEVINIT EQU     0               ;Initialization
19895                               DINITHL EQU     26              ;Size of init header
19896                               DEVMDCH EQU     1               ;Media check
19897                               DMEDHL  EQU     15              ;Size of media check header
19898                               DEVBPB  EQU     2               ;Get BPB
19899                               DEVRDIOCTL EQU  3               ;IOCTL read
19900                               DBPBHL  EQU     22              ;Size of Get BPB header
19901                               DEVRD   EQU     4               ;Read
19902                               DRDWRHL EQU     22              ;Size of RD/WR header
19903                               DEVRDND EQU     5               ;Non destructive read no wait (character devs)
19904                               DRDNDHL EQU     14              ;Size of non destructive read header
19905                               DEVIST  EQU     6               ;Input status
19906                               DSTATHL EQU     13              ;Size of status header
19907                               DEVIFL  EQU     7               ;Input flush
19908                               ; 21/02/2024
19909                               ;DFLSHL EQU     15              ;Size of flush header
19910                               DFLSHL      equ    13      ; PCDOS 7.1 IBMDOS.COM  ; 21/02/2024
19911                               DEVWRT  EQU     8               ;Write
19912                               DEVWRTV EQU     9               ;Write with verify
19913                               DEVOST  EQU     10              ;Output status
19914                               DEVOFL  EQU     11              ;Output flush
19915                               DEVWRIOCTL EQU  12              ;IOCTL write
19916
19917                               ; --------------------------------------------------------------------
19918                               struc       SYS_FCB
19919 00000000 ??                  .fcb_drive: resb 1
19920 00000001 ??????????????      .fcb_name:  resb 8
19921 00000009 ??????              .fcb_ext:   resb 3
19922 0000000C ????                .fcb_EXTENT:       resw 1
19923 0000000E ????                .fcb_RECSIZ:       resw 1  ; Size of record (user settable)
19924 00000010 ????                .fcb_FILSIZ:       resw 1  ; Size of file in bytes; used with the following
19925                                                          ; word
19926 00000012 ????                .fcb_DRVBP: resw 1  ; BP for SEARCH FIRST and SEARCH NEXT
19927 00000014 ????                .fcb_FDATE: resw 1  ; Date of last writing
19928 00000016 ????                .fcb_FTIME: resw 1  ; Time of last writing
19929 00000018 ??                  .fcb_DEVID: resb 1  ; Device ID number, bits 0-5 if file.
19930                                                          ; bit 7=0 for file, bit 7=1 for I/O device
19931                                                          ; If file, bit 6=0 if dirty
19932                                                          ; If I/O device, bit 6=0 if EOF (input)
19933                                                          ;               Bit 5=1 if Raw mode
19934                                                          ;               Bit 0=1 if console input device
19935                                                          ;               Bit 1=1 if console output device
19936                                                          ;               Bit 2=1 if null device
19937                                                          ;               Bit 3=1 if clock device
19938 00000019 ????                .fcb_FIRCLUS:      resw 1  ; First cluster of file
19939 0000001B ????                .fcb_CLUSPOS:      resw 1  ; Position of last cluster accessed
19940 0000001D ????                .fcb_LSTCLUS:      resw 1  ; Last cluster accessed and directory
19941 0000001F ??                                     resb 1  ; pack 2 12 bit numbers into 24 bits...
19942 00000020 ??                  .fcb_NR:    resb 1  ; Next record
19943 00000021 ????????            .fcb_RR:    resb 4  ; Random record
19944                               .size:
19945                               endstruc
19946
19947                               ; --------------------------------------------------------------------
19948                               ; Field definition for I/O buffer information
19949
19950                               ; 22/03/2019 - Retro DOS v4.0 (MSDOS 6.0, BUFFER.INC, 1991)
19951
19952                                   ; 03/01/2024 - Retro DOS v5.0 (PCDOS 7.1 IBMDOS.COM)
19953
19954                               struc BUFFINFO
19955 00000000 ????                .buf_next:  resw 1          ; Pointer to next buffer in list
19956 00000002 ????                .buf_prev:  resw 1          ; Pointer to prev buffer in list
19957 00000004 ??                  .buf_ID:    resb 1          ; Drive of buffer (bit 7 = 0)
19958                                                          ; SFT table index (bit 7 = 1)
19959                                                          ; = FFH if buffer free
19960 00000005 ??                  .buf_flags: resb 1          ; Bit 7 = 1 if Remote file buffer
19961                                                          ;       = 0 if Local device buffer
19962                                                          ; Bit 6 = 1 if buffer dirty
19963                                                          ; 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
```

```
20088                                   ; FAT32 fs ; 01/01/2024
20089                                   ; ref: https://en.wikibooks.org/wiki/
20090                                   ;      First_steps_towards_system_programming_under_MS-DOS_7/Appendix
20091                                   ;   -- A.03-1. Structure of Drive Parameters Blocks (DPB) ---
20092 00000021 ????                    .FREE_CNT_HW:      resw 1  ; 33   ; High word of free cluster count
20093 00000023 ????                    .EXT_FLAGS: resw 1  ; 35   ; FAT32 extended flags (active FAT number)
20094 00000025 ????                    .FSINFO_SECTOR:    resw 1  ; 37   ; (FAT32 fs) FSINFO structure sector address
20095 00000027 ????                    .BKBOOT_SECTOR:    resw 1  ; 39   ; (FAT32 fs) Backup Boot Sector address
20096 00000029 ????????                .FCLUS_FSECTOR: resd 1     ; 41   ; The first cluster's first sector address
20097 0000002D ????????                .LAST_CLUSTER:     resd 1  ; 45   ; The last cluster number
20098 00000031 ????????                .FAT32_SIZE:       resd 1  ; 49   ; Number of FAT sectors (for FAT32 fs)
20099 00000035 ????????                .ROOT_CLUSTER:     resd 1  ; 53   ; Root directory's cluster number (FAT32 fs)
20100                                   ; 01/01/2024 - Retro DOS v5.0
20101 00000039 ????????                .FAT32_NXTFREE:    resd 1  ; 57   ; The next free cluster (for FAT32 fs)
20102                                   .size:             ; 61 bytes ; 01/01/2024 (PCDOS 7.1)
20103                                   endstruc
20104
20105                                   %endif
20106
20107                                   DPBSIZ  EQU     DPB.size  ; Size of the structure in bytes
20108
20109                                   DSKSIZ  EQU DPB.MAX_CLUSTER       ; Size of disk (temp used during init only)
20110
20111                                   ; --------------------------------------------------------------------------
20112                                   ; 26/03/2018
20113
20114                                   ; IOCTL SUB-FUNCTIONS
20115                                   IOCTL_GET_DEVICE_INFO     EQU     0
20116                                   IOCTL_SET_DEVICE_INFO     EQU     1
20117                                   IOCTL_READ_HANDLE  EQU    2
20118                                   IOCTL_WRITE_HANDLE EQU    3
20119                                   IOCTL_READ_DRIVE   EQU    4
20120                                   IOCTL_WRITE_DRIVE  EQU    5
20121                                   IOCTL_GET_INPUT_STATUS    EQU     6
20122                                   IOCTL_GET_OUTPUT_STATUS EQU       7
20123                                   IOCTL_CHANGEABLE?  EQU    8
20124                                   IOCTL_SHARING_RETRY       EQU     11
20125                                   GENERIC_IOCTL_HANDLE      EQU     12
20126                                   GENERIC_IOCTL             EQU     13
20127
20128                                   ; GENERIC IOCTL SUB-FUNCTIONS
20129                                   RAWIO                     EQU     8
20130
20131                                   ; RAWIO SUB-FUNCTIONS
20132                                   GET_DEVICE_PARAMETERS     EQU     60H
20133                                   SET_DEVICE_PARAMETERS     EQU     40H
20134                                   READ_TRACK         EQU    61H
20135                                   WRITE_TRACK        EQU    41H
20136                                   VERIFY_TRACK              EQU     62H
20137                                   FORMAT_TRACK              EQU     42H
20138
20139                                   ; DEVICETYPE VALUES
20140                                   MAX_SECTORS_IN_TRACK      EQU     63
20141                                   DEV_5INCH          EQU    0
20142                                   DEV_5INCH96TPI            EQU     1
20143                                   DEV_3INCH720KB            EQU     2
20144                                   DEV_8INCHSS        EQU    3
20145                                   DEV_8INCHDS        EQU    4
20146                                   DEV_HARDDISK              EQU     5
20147                                   DEV_OTHER          EQU    7
20148                                   ;DEV_3INCH1440KB   EQU    7
20149                                   DEV_3INCH2880KB           EQU     9
20150                                   ; Retro DOS v2.0 - 26/03/2018
20151                                   ;;DEV_TAPE         EQU    6
20152                                   ;;DEV_ERIMO        EQU    8
20153                                   ;DEV_3INCH2880KB   EQU    9
20154                                   DEV_3INCH1440KB           EQU     10
20155
20156                                   ;MAX_DEV_TYPE             EQU     9       ; MAXIMUM DEVICE TYPE THAT WE
20157                                                             ; CURRENTLY SUPPORT.
20158                                   MAX_DEV_TYPE              EQU     10
20159
20160                                   struc A_SECTORTABLE
20161 00000000 ????                    .ST_SECTORNUMBER:  resw    1
20162 00000002 ????                    .ST_SECTORSIZE:           resw    1
20163                                   .size:
20164                                   endstruc
20165
20166                                   ; --------------------------------------------------------------------------
20167                                   ; structure, equates for devmark for mem command.
20168
20169                                   ; 25/03/2019 - Retro DOS v4.0 (MSDOS 6.0, DEVMARK.INC, 1991)
20170
20171                                   struc devmark
20172 00000000 ??                       .id:           resb 1
20173 00000001 ????                     .seg:          resw 1
20174 00000003 ????                     .size:         resw 1
20175 00000005 ??????                   .dum:          resb 3
20176 00000008 ????????????????        .filename: resb 8
20177                                   endstruc
20178
20179                                   devmark_stk equ     'S'
20180                                   devmark_device      equ     'D'
20181                                   devmark_ifs equ     'I'
20182                                   devmark_buf equ     'B'
20183                                   devmark_cds equ     'L' ; lastdrive
20184                                   devmark_files       equ     'F'
20185                                   devmark_fcbs        equ     'X'
20186                                   devmark_inst        equ     'T' ; used for sysinit base for install= command.
20187                                   devmark_ems_stub equ        'E'
20188
20189                                   setbrkdone  equ     00000001b
20190                                   for_devmark equ     00000010b
20191                                   not_for_devmark equ       11111101b
20192
20193                                   ; --------------------------------------------------------------------------
20194                                   ; Memory arena structure
20195
20196                                   ; 24/03/2019 - Retro DOS v4.0
20197                                   ; (MSDOS 6.0, ARENA.INC)
20198
20199                                   ;** Arena Header
20200
20201                                   struc ARENA
20202 00000000 ??                       .SIGNATURE: resb 1         ; 4D for valid item, 5A for last item
20203 00000001 ????                     .OWNER:            resw 1         ; owner of arena item
20204 00000003 ????                     .SIZE:             resw 1         ; size in paragraphs of item
20205 00000005 ??????                   .RESERVED   resb 3         ; reserved
20206 00000008 ????????????????        .NAME:             resb 8         ; owner file name
20207                                   endstruc
20208
20209                                   ; 12/04/2019
20210
20211                                   arena_owner_system EQU 0   ; free block indication
```

```
20212
20213                                   arena_signature_normal    EQU 4Dh ; valid signature, not end of arena
20214                                   arena_signature_end       EQU 5Ah ; valid signature, last block in arena
20215
20216                                   ; --------------------------------------------------------------------
20217                                   ; Process data block (otherwise known as program header)
20218
20219                                   ; 23/03/2019 - Retro DOS v4.0
20220
20221                                   ; (MSDOS 6.0 - PDB.INC, 1991)
20222
20223                                   FILPERPROC  EQU     20
20224
20225                                   struc PDB   ; Process_data_block
20226 00000000 ????                    .EXIT_CALL: resw 1          ; INT int_abort system terminate
20227 00000002 ????                    .BLOCK_LEN: resw 1          ; size of execution block
20228 00000004 ??                                  resb 1
20229 00000005 ??????????              .CPM_CALL:  resb 5          ; ancient call to system
20230 0000000A ????????                .EXIT:              resd 1          ; pointer to exit routine
20231 0000000E ????????                .CTRL_C:    resd 1          ; pointer to ^C routine
20232 00000012 ????????                .FATAL_ABORT:       resd 1          ; pointer to fatal error
20233 00000016 ????                    .PARENT_PID:        resw 1          ; PID of parent (terminate PID)
20234 00000018 <res 14h>               .JFN_TABLE:     resb FILPERPROC ; indices into system table
20235 0000002C ????                    .ENVIRON:   resw 1          ; seg addr of environment
20236 0000002E ????????                .USER_STACK:        resd 1          ; stack of self during system calls
20237 00000032 ????                    .JFN_LENGTH:        resw 1          ; number of handles allowed
20238 00000034 ????????                .JFN_POINTER:       resd 1          ; pointer to JFN table
20239 00000038 ????????                .NEXT_PDB:  resd 1          ; pointer to nested PDB's
20240 0000003C ??                      .INTERCON:  resb 1          ; *** jh-3/28/90 ***
20241 0000003D ??                      .APPEND:    resb 1          ; *** Not sure if still used ***
20242 0000003E ????                    .NOVELL_USED:       resb 2          ; Novell shell (redir) uses these
20243 00000040 ????                    .VERSION:   resw 1          ; DOS version reported to this app
20244 00000042 <res Eh>                .PAD1:              resb 14         ;
20245 00000050 ??????????              .CALL_SYSTEM:       resb 5          ; portable method of system call
20246 00000055 ????????????????        .PAD2:              resb 7          ; reserved so FCB 1 can be used as an extended FCB
20247 0000005C <res 10h>               .FCB1:              resb 16         ; default FCB 1
20248 0000006C <res 10h>               .FCB2:              resb 16         ; default FCB 2
20249 0000007C ????????                .PAD3:              resb 4          ; not sure if this is used by PDB_FCB2
20250 00000080 <res 80h>               .TAIL:              resb 128        ; command tail and default DTA
20251                                   ;.size:
20252                                   endstruc
20253
20254                                   ; --------------------------------------------------------------------
20255                                   ; <system call definitions>
20256
20257                                   ; 23/03/2019 - Retro DOS v4.0
20258
20259                                   ; (MSDOS 6.0 - SYSCALL.INC, 1991)
20260
20261                                   ABORT                           EQU 0   ;  0      0
20262                                   STD_CON_INPUT                   EQU 1   ;  1      1
20263                                   STD_CON_OUTPUT                  EQU 2   ;  2      2
20264                                   STD_AUX_INPUT                   EQU 3   ;  3      3
20265                                   STD_AUX_OUTPUT                  EQU 4   ;  4      4
20266                                   STD_PRINTER_OUTPUT              EQU 5   ;  5      5
20267                                   RAW_CON_IO                      EQU 6   ;  6      6
20268                                   RAW_CON_INPUT                   EQU 7   ;  7      7
20269                                   STD_CON_INPUT_NO_ECHO           EQU 8   ;  8      8
20270                                   STD_CON_STRING_OUTPUT           EQU 9   ;  9      9
20271                                   STD_CON_STRING_INPUT            EQU 10  ; 10      A
20272                                   STD_CON_INPUT_STATUS            EQU 11  ; 11      B
20273                                   STD_CON_INPUT_FLUSH             EQU 12  ; 12      C
20274                                   DISK_RESET                      EQU 13  ; 13      D
20275                                   SET_DEFAULT_DRIVE               EQU 14  ; 14      E
20276                                   FCB_OPEN                        EQU 15  ; 15      F
20277                                   FCB_CLOSE                       EQU 16  ; 16     10
20278                                   DIR_SEARCH_FIRST                EQU 17  ; 17     11
20279                                   DIR_SEARCH_NEXT                 EQU 18  ; 18     12
20280                                   FCB_DELETE                      EQU 19  ; 19     13
20281                                   FCB_SEQ_READ                    EQU 20  ; 20     14
20282                                   FCB_SEQ_WRITE                   EQU 21  ; 21     15
20283                                   FCB_CREATE                      EQU 22  ; 22     16
20284                                   FCB_RENAME                      EQU 23  ; 23     17
20285                                   GET_DEFAULT_DRIVE               EQU 25  ; 25     19
20286                                   SET_DMA                         EQU 26  ; 26     1A
20287                                   GET_DEFAULT_DPB                 EQU 31  ; 31     1F
20288                                   FCB_RANDOM_READ                 EQU 33  ; 33     21
20289                                   FCB_RANDOM_WRITE                EQU 34  ; 34     22
20290                                   GET_FCB_FILE_LENGTH             EQU 35  ; 35     23
20291                                   GET_FCB_POSITION                EQU 36  ; 36     24
20292                                   SET_INTERRUPT_VECTOR            EQU 37  ; 37     25
20293                                   CREATE_PROCESS_DATA_BLOCK       EQU 38  ; 38     26
20294                                   FCB_RANDOM_READ_BLOCK           EQU 39  ; 39     27
20295                                   FCB_RANDOM_WRITE_BLOCK          EQU 40  ; 40     28
20296                                   PARSE_FILE_DESCRIPTOR           EQU 41  ; 41     29
20297                                   GET_DATE                        EQU 42  ; 42     2A
20298                                   SET_DATE                        EQU 43  ; 43     2B
20299                                   GET_TIME                        EQU 44  ; 44     2C
20300                                   SET_TIME                        EQU 45  ; 45     2D
20301                                   SET_VERIFY_ON_WRITE             EQU 46  ; 46     2E
20302                                   ; Extended functionality group
20303                                   GET_DMA                         EQU 47  ; 47     2F
20304                                   GET_VERSION                     EQU 48  ; 48     30
20305                                   KEEP_PROCESS                    EQU 49  ; 49     31
20306                                   GET_DPB                         EQU 50  ; 50     32
20307                                   SET_CTRL_C_TRAPPING             EQU 51  ; 51     33
20308                                   GET_INDOS_FLAG                  EQU 52  ; 52     34
20309                                   GET_INTERRUPT_VECTOR            EQU 53  ; 53     35
20310                                   GET_DRIVE_FREESPACE             EQU 54  ; 54     36
20311                                   CHAR_OPER                       EQU 55  ; 55     37
20312                                   INTERNATIONAL                   EQU 56  ; 56     38
20313                                   ;   Directory Group
20314                                   MKDIR                           EQU 57  ; 57     39
20315                                   RMDIR                           EQU 58  ; 58     3A
20316                                   CHDIR                           EQU 59  ; 59     3B
20317                                   ;   File Group
20318                                   CREAT                           EQU 60  ; 60     3C
20319                                   OPEN                            EQU 61  ; 61     3D
20320                                   CLOSE                           EQU 62  ; 62     3E
20321                                   READ                            EQU 63  ; 63     3F
20322                                   WRITE                           EQU 64  ; 64     40
20323                                   UNLINK                          EQU 65  ; 65     41
20324                                   LSEEK                           EQU 66  ; 66     42
20325                                   CHMOD                           EQU 67  ; 67     43
20326                                   IOCTL                           EQU 68  ; 68     44
20327                                   XDUP                            EQU 69  ; 69     45
20328                                   XDUP2                           EQU 70  ; 70     46
20329                                   CURRENT_DIR                     EQU 71  ; 71     47
20330                                   ;    Memory Group
20331                                   ALLOC                           EQU 72  ; 72     48
20332                                   DEALLOC                         EQU 73  ; 73     49
20333                                   SETBLOCK                        EQU 74  ; 74     4A
20334                                   ;    Process Group
20335                                   EXEC                            EQU 75  ; 75     4B
```

```
20336                                   EXIT                            EQU 76  ; 76     4C
20337                                   WAITPROCESS             EQU 77  ; 77     4D
20338                                   FIND_FIRST                      EQU 78  ; 78     4E
20339                                   ;   Special Group
20340                                   FIND_NEXT                       EQU 79  ; 79     4F
20341                                   ; SPECIAL SYSTEM GROUP
20342                                   SET_CURRENT_PDB                 EQU 80  ; 80     50
20343                                   GET_CURRENT_PDB                 EQU 81  ; 81     51
20344                                   GET_IN_VARS                     EQU 82  ; 82     52
20345                                   SETDPB                          EQU 83  ; 83     53
20346                                   GET_VERIFY_ON_WRITE             EQU 84  ; 84     54
20347                                   DUP_PDB                         EQU 85  ; 85     55
20348                                   RENAME                          EQU 86  ; 86     56
20349                                   FILE_TIMES                      EQU 87  ; 87     57
20350                                   ;
20351                                   ALLOCOPER               EQU 88  ; 88     58
20352                                   ; Network extention system calls
20353                                   GetExtendedError        EQU 89  ; 89     59
20354                                   CreateTempFile                  EQU 90  ; 90     5A
20355                                   CreateNewFile                   EQU 91  ; 91     5B
20356                                   LockOper                EQU 92  ; 92     5C Lock and Unlock
20357                                   ServerCall              EQU 93  ; 93     5D CommitAll, ServerDOSCall,
20358                                                                   ;           CloseByName, CloseUser,
20359                                                                   ;           CloseUserProcess,
20360                                                                   ;           GetOpenFileList
20361                                   UserOper                EQU 94  ; 94     5E Get and Set
20362                                   AssignOper              EQU 95  ; 95     5F On, Off, Get, Set, Cancel
20363                                   xNameTrans              EQU 96  ; 96     60
20364                                   PathParse               EQU 97  ; 97     61
20365                                   GetCurrentPSP                   EQU 98  ; 98     62
20366                                   Hongeul                 EQU 99  ; 99     63
20367                                   ECS_CALL                EQU 99  ; 99     63  ;; DBCS support
20368                                   Set_Printer_Flag        EQU 100 ; 100    64
20369                                   GetExtCntry             EQU 101 ; 101    65
20370                                   GetSetCdPg              EQU 102 ; 102    66
20371                                   ExtHandle               EQU 103 ; 103    67
20372                                   Commit                          EQU 104 ; 104    68
20373                                   GetSetMediaID                   EQU 105 ; 105    69
20374                                   IFS_IOCTL               EQU 107 ; 107    6B
20375                                   ExtOpen                 EQU 108 ; 108    6C
20376                                   ;
20377                                   ;ifdef ROMEXEC
20378                                   ;ROM_FIND_FIRST                 EQU 109 ; 109     6D
20379                                   ;ROM_FIND_NEXT                  EQU 110 ; 110     6E
20380                                   ;ROM_EXCLUDE                    EQU 111 ; 111    6F
20381                                   ;endif
20382                                   ;
20383                                   Set_Oem_Handler         EQU 248 ; 248    F8
20384                                   OEM_C1                          EQU 249 ; 249    F9
20385                                   OEM_C2                          EQU 250 ; 250    FA
20386                                   OEM_C3                          EQU 251 ; 251    FB
20387                                   OEM_C4                          EQU 252 ; 252    FC
20388                                   OEM_C5                          EQU 253 ; 253    FD
20389                                   OEM_C6                          EQU 254 ; 254    FE
20390                                   OEM_C7                          EQU 255 ; 255    FF
20391
20392                                   ; --------------------------------------------------------------------------
20393                                   ; SYSCONF.ASM (MSDOS 3.3 - 24/07/1987)
20394                                   ; --------------------------------------------------------------------------
20395
20396                                   ;;  IF      STACKSW
20397
20398                                   ;;
20399                                   ;; Internal Stack Parameters
20400                                   ;EntrySize          equ     8
20401                                   ;
20402                                   ;MinCount           equ     8
20403                                   ;DefaultCount               equ     9
20404                                   ;MaxCount           equ     64
20405                                   ;
20406                                   ;MinSize            equ     32
20407                                   ;DefaultSize                equ     128
20408                                   ;MaxSize            equ     512
20409
20410                                   ;;  ENDIF
20411
20412                                   ; --------------------------------------------------------------------------
20413                                   ; BIOSTRUC.INC (MSDOS 3.3 - 24/07/1987)
20414                                   ; --------------------------------------------------------------------------
20415                                                                   ;;Rev 3.30 Modification
20416                                   ; ROM BIOS CALL PACKET STRUCTURES
20417
20418                                   ;********************************
20419                                   ;System Service call ( Int 15h )
20420                                   ;********************************
20421                                   ;Function AH = 0C0h, Return system configuration
20422                                   ;For PC and PCJR on return:
20423                                   ;   (AH)    = 80h
20424                                   ;   (CY)    = 1
20425                                   ;For PCXT, PC PORTABLE and PCAT on return:
20426                                   ;   (AH)    = 86h
20427                                   ;   (CY)    = 1
20428                                   ;For all others:
20429                                   ;   (AH)    = 0
20430                                   ;   (CY)    = 0
20431                                   ;   (ES:BX) = pointer to system descriptor vector in ROS
20432                                   ; System descriptor :
20433                                   ;   DW      xxxx            length of descriptor in bytes,
20434                                   ;                           minimum length = 8
20435                                   ;   DB      xx              model byte
20436                                   ;                           0FFh    = PC
20437                                   ;                           0FEh    = PC/XT, Portable
20438                                   ;                           0FDh    = PC/JR
20439                                   ;                           0FCh    = PC/AT
20440                                   ;                           0F9h    = Convertable
20441                                   ;                           0F8h    = Model 80
20442                                   ;                           0E0 thru 0EFh = reserved
20443                                   ;
20444                                   ;   DB      xx              secondary model byte
20445                                   ;                           000h    = PC1
20446                                   ;                           000h    = PC/XT, Portable
20447                                   ;                           000h    = PC/JR
20448                                   ;                           000h    = PC/AT
20449                                   ;                           001h    = PC/AT Model 339
20450                                   ;                           003h    = PC/RT
20451                                   ;                           000h    = Convertable
20452                                   ;
20453                                   ;   DB      xx              bios revision level
20454                                   ;                           00 for first release, subsequent release
20455                                   ;                           of code with same model byte and
20456                                   ;                           secondary model byte require revison level
20457                                   ;                           to increase by one.
20458                                   ;
20459                                   ;   DB      xx              feature information byte 1
```

```
                                ;                       X0000000 = 1, bios use DMA channel 3
                                ;                               = 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
```

```
20584 00000006 ????????                .Cache_ptr:    resd 1      ; pointer to secondary cache
20585 0000000A ????                    .Cache_count:      resw 1  ; number of secondary cache entries
20586
20587 0000000C ??                      .Buff_In_HMA:         resb 1      ; flag to indicate that buffers
20588                                                                      ; are in HMA
20589 0000000D ????????                .Lo_Mem_Buff:         resd 1      ; Ptr to scratch buff in Low Mem
20590                                                                      ;  used to read/write on disks
20591 00000011 ????????                .UU_EMS_FIRST_PAGE:       resw 2
20592 00000015 ????                    .UU_EMS_NPA640:           resw 1
20593 00000017 ??                      .UU_EMS_mode:             resb 1  ; no EMS = -1
20594 00000018 ????                    .UU_EMS_handle:           resw 1  ; EMS handle for buffers
20595 0000001A ????                    .UU_EMS_PageFrame_Number: resw 1 ; EMS page frame number
20596 0000001C ????                    .UU_EMS_Seg_Cnt:   resw 1  ; EMS segment count
20597 0000001E ????                    .UU_EMS_Page_Frame:       resw 1  ; EMS page frame segment address
20598 00000020 ????                    .UU_EMS_reserved:  resw 1  ; EMS segment count
20599 00000022 ??                      .UU_EMS_Map_Buff:  resb 1  ; map buffer
20600                                  .size:
20601                                  endstruc
20602
20603                                  ; --------------------------------------------------------------------------
20604                                  ; CURDIR.INC (MSDOS 6.0 - 1991)
20605                                  ; --------------------------------------------------------------------------
20606                                  ; 22/03/2019 - Retro DOS v4.0
20607
20608                                  ;** CDS - Current Directory Structure
20609                                  ;
20610                                  ; CDS items are used bu the internal routines to store cluster numbers and
20611                                  ; network identifiers for each logical name.  The ID field is used dually,
20612                                  ; both as net ID and for a cluster number for local devices. In the case
20613                                  ; of local devices, the cluster number will be -1 if there is a potential
20614                                  ; of the disk being changed or if the path must be recracked.
20615                                  ;
20616                                  ;   Some pathnames have special preambles, such as
20617                                  ;
20618                                  ;           \\machine\sharename\...
20619                                  ;   For these pathnames we can't allow ".." processing to back us
20620                                  ;   up into the special front part of the name.  The CURDIR_END field
20621                                  ;   holds the address of the seperator character which marks
20622                                  ;   the split between the special preamble and the regular
20623                                  ;   path list; ".." processing isn't allowed to back us up past
20624                                  ;   (i.e., before) CURDIR_END
20625                                  ;   For the root, it points at the leading /. For net
20626                                  ;   assignments it points at the end (nul) of the initial assignment:
20627                                  ;   A:/    \\foo\bar          \\foo\bar\blech\bozo
20628                                  ;     ^           ^                   ^
20629
20630                                  DIRSTRLEN   EQU    64+3          ; Max length in bytes of directory strings
20631                                  TEMPLEN     EQU    DIRSTRLEN*2
20632
20633                                  struc              curdir_list
20634                                  ; MSDOS 3.3
20635 00000000 <res 43h>               .cdir_text  resb   DIRSTRLEN     ; text of assignment and curdir
20636 00000043 ????                    .cdir_flags resw   1             ; various flags
20637 00000045 ????????                .cdir_devptr       resd   1            ; local pointer to DPB or net device
20638 00000049 ????????                .cdir_ID    resw   2             ; cluster of current dir (net ID)
20639 0000004D ????                    .cdir_usr_word     resw   1
20640 0000004F ????                    .cdir_end   resw   1             ; end of assignment
20641                                  ; MSDOS 6.0
20642 00000051 ??                      .cdir_type: resb   1             ; IFS drive (2=ifs, 4=netuse)
20643 00000052 ????????                .cdir_ifd_hdr:     resd   1            ; Ptr to File System Header
20644 00000056 ????                    .cdir_fsda: resb   2             ; File System Dependent Data Area
20645                                  .size:
20646                                  endstruc
20647
20648                                  curdirlen   EQU    curdir_list.size      ; Needed for screwed up
20649                                                                           ; ASM87 which doesn't allow
20650                                                                           ; Size directive as a macro
20651                                                                           ; argument
20652                                  %define curdir_netID      dword [curdir_list.cdir_ID]
20653
20654                                  ;** Flag values for CURDIR_FLAGS
20655
20656                                  ;Flag word masks
20657                                  curdir_isnet       EQU    1000000000000000B
20658                                  curdir_isifs       EQU    1000000000000000B
20659                                  curdir_inuse       EQU    0100000000000000B
20660                                  curdir_splice      EQU    0010000000000000B
20661                                  curdir_local       EQU    0001000000000000B
20662
20663                                  ; --------------------------------------------------------------------------
20664                                  ; SF.INC (MSDOS 6.0 - 1991)
20665                                  ; --------------------------------------------------------------------------
20666                                  ; 25/03/2019 - Retro DOS v4.0
20667
20668                                  ; 09/04/2024 - Retro DOS v4.2 (BugFix)
20669                                  ; 09/04/2024 - Retro DOS v5.0
20670
20671                                  ; system file table
20672
20673                                  ;** System File Table SuperStructure
20674                                  ;
20675                                  ;   The system file table entries are allocated in contiguous groups.
20676                                  ;   There may be more than one such groups; the SF "superstructure"
20677                                  ;   tracks the groups.
20678
20679                                  struc      SF
20680 00000000 ????????                .SFLink:   resd   1
20681 00000004 ????                    .SFCount:  resw   1              ; number of entries
20682 00000006 ????                    .SFTable:  resw   1              ; beginning of array of the following
20683                                  .size:
20684                                  endstruc
20685
20686                                  ;** System file table entry
20687                                  ;
20688                                  ;   These are the structures which are at SFTABLE in the SF structure.
20689
20690                                  struc      SF_ENTRY
20691 00000000 ????                    .sf_ref_count:     resw   1              ; number of processes sharing entry
20692                                                                          ;   if FCB then ref count
20693 00000002 ????                    .sf_mode:  resw   1              ; mode of access or high bit on if FCB
20694 00000004 ??                      .sf_attr:  resb   1              ; attribute of file
20695 00000005 ????                    .sf_flags: resw   1              ;Bits 8-15
20696                                                                          ; Bit 15 = 1 if remote file
20697                                                                          ;       = 0 if local file or device
20698                                                                          ; Bit 14 = 1 if date/time is not to be
20699                                                                          ;   set from clock at CLOSE.  Set by
20700                                                                          ;   FILETIMES and FCB_CLOSE.  Reset by
20701                                                                          ;   other reseters of the dirty bit
20702                                                                          ;   (WRITE)
20703                                                                          ; Bit 13 = Pipe bit (reserved)
20704                                                                          ;
20705                                                                          ; Bits 0-7 (old FCB_devid bits)
20706                                                                          ; If remote file or local file, bit
20707                                                                          ; 6=0 if dirty Device ID number, bits
```

```
20708                                                   ; 0-5 if local file.
20709                                                   ; bit 7=0 for local file, bit 7
20710                                                   ;     =1 for local I/O device
20711                                                   ; If local I/O device, bit 6=0 if EOF (input)
20712                                                   ;             Bit 5=1 if Raw mode
20713                                                   ;             Bit 0=1 if console input device
20714                                                   ;             Bit 1=1 if console output device
20715                                                   ;             Bit 2=1 if null device
20716                                                   ;             Bit 3=1 if clock device
20717 00000007 ????????              .sf_devptr: resd   1             ; Points to DPB if local file, points
20718                                                   ; to device header if local device,
20719                                                   ; points to net device header if
20720                                                   ; remote
20721 0000000B ????                  .sf_firclus:       resw   1            ; First cluster of file (bit 15 = 0)
20722                                ;.sf_lstclus:      resw   1 ; *
20723 0000000D ????                  .sf_time:   resw   1             ; Time associated with file
20724 0000000F ????                  .sf_date:   resw   1             ; Date associated with file
20725 00000011 ????????              .sf_size:   resd   1             ; Size associated with file
20726 00000015 ????????              .sf_position:      resd   1            ; Read/Write pointer or LRU count for FCBs
20727                                ;
20728                                ; Starting here, the next 7 bytes may be used by the file system to store an
20729                                ; ID
20730                                ;
20731 00000019 ????                  .sf_cluspos:       resw   1            ; Position of last cluster accessed
20732 0000001B ????????              .sf_dirsec: resd   1 ; 09/04/2024 ; Sector number of directory sector for this file
20733 0000001F ??                    .sf_dirpos: resb   1             ; Offset of this entry in the above
20734                                ;
20735                                ; End of 7 bytes of file-system specific info.
20736                                ;
20737 00000020 <res Bh>              .sf_name:   resb   11            ; 11 character name that is in the
20738                                                   ; directory entry.  This is used by
20739                                                   ; close to detect file deleted and
20740                                                   ; disk changed errors.
20741                                ; SHARING INFO
20742 0000002B ????????              .sf_chain:  resd   1             ; link to next SF
20743 0000002F ????                  .sf_UID:    resw   1
20744 00000031 ????                  .sf_PID:    resw   1
20745 00000033 ????                  .sf_MFT:    resw   1
20746 00000035 ????                  .sf_lstclus:       resw   1 ; *         ; Last cluster accessed
20747 00000037 ????????              .sf_IFS_HDR:       resd   1 ; **
20748                                .size:
20749                                endstruc
20750
20751                                ; --------------------------------------------------------------------
20752                                ; DOSCNTRY.INC (MSDOS 3.3 - 24/07/1987)
20753                                ; --------------------------------------------------------------------
20754                                ; 11/06/2018 - Retro DOS v3.0
20755
20756                                ;Equates for COUNTRY INFORMATION.
20757                                SetCountryInfo             EQU     1       ;country info
20758                                SetUcase           EQU     2       ;uppercase table
20759                                SetLcase           EQU     3       ;lowercase table (Reserved)
20760                                SetUcaseFile               EQU     4       ;uppercase file spec table
20761                                SetFileList        EQU     5       ;valid file character list
20762                                SetCollate         EQU     6       ;collating sequence
20763                                SetDBCS            EQU     7       ;double byte character set
20764                                SetALL                     EQU     -1      ;all the entries
20765
20766                                ;DOS country and code page information table structure.
20767                                ;Internally, IBMDOS gives a pointer to this table.
20768                                ;IBMBIO, MODE and NLSFUNC modules communicate with IBMDOS through
20769                                ;this structure.
20770
20771                                struc country_cdpg_info ; DOS_country_cdpg_info
20772 00000000 ??????????????????        .ccInfo_reserved : resb    8       ;reserved for internal use
20773 00000008 <res 40h>             .ccPath_CountrySys:        resb    64      ;path and filename for country info
20774 00000048 ????                  .ccSysCodePage:            resw    1       ;system code page id
20775 0000004A ????                  .ccNumber_of_entries:      resw    1 ; dw 5
20776 0000004C ??                    .ccSetUcase:               resb    1 ; db SetUcase ; = 2
20777 0000004D ????????              .ccUcase_ptr:              resd    1       ;pointer to Ucase table
20778
20779 00000051 ??                    .ccSetUcaseFile:   resb    1 ; db SetUcaseFile ; = 4
20780 00000052 ????????              .ccFileUcase_ptr:  resd    1       ;pointer to File Ucase table
20781
20782 00000056 ??                    .ccSetFileList:            resb    1 ; db SetFileList ; = 5
20783 00000057 ????????              .ccFileChar_ptr:   resd    1       ;pointer to File char list table
20784
20785 0000005B ??                    .ccSetCollate:             resb    1 ; db SetCollate ; = 6
20786 0000005C ????????              .ccCollate_ptr:            resd    1       ;pointer to collate table
20787
20788 00000060 ??                    .ccSetCountryInfo: resb    1 ; db SetCountryInfo ; = 1
20789 00000061 ????                  .ccCountryInfoLen: resw    1       ;length of country info
20790 00000063 ????                  .ccDosCountry:             resw    1       ;system country code id
20791 00000065 ????                  .ccDosCodePage:            resw    1       ;system code page id
20792 00000067 ????                  .ccDFormat:        resw    1       ;date format
20793 00000069 ??????????            .ccCurSymbol:              resb    5 ; db "    ",0
20794                                                                   ;5 byte of (currency symbol+0)
20795 0000006E ????                  .cc1000Sep:        resb    2 ; db " ",0 ;2 byte of (1000 sep. + 0)
20796 00000070 ????                  .ccDecSep:         resb    2 ; db " ",0 ;2 byte of (Decimal sep. + 0)
20797 00000072 ????                  .ccDateSep:        resb    2 ; db " ",0 ;2 byte of (date sep. + 0)
20798 00000074 ????                  .ccTimeSep:        resb    2 ; db " ",0 ;2 byte of (time sep. + 0)
20799 00000076 ??                    .ccCFormat:        resb    1       ;currency format flags
20800 00000077 ??                    .ccCSigDigits:             resb    1       ;# of digits in currency
20801 00000078 ??                    .ccTFormat:        resb    1       ;time format
20802 00000079 ????????              .ccMono_Ptr:               resd    1       ;monocase routine entry point
20803 0000007D ????                  .ccListSep:        resb    2 ; db " ",0 ;data list separator
20804 0000007F <res Ah>              .ccReserved_area:  resw    5 ; dw 5 dup(?) ;reserved
20805                                .size:
20806                                endstruc
20807
20808                                NEW_COUNTRY_SIZE    equ  country_cdpg_info.size - country_cdpg_info.ccDosCountry
20809
20810                                ; ====================================================================
20811                                ; retrodos4.s (offset addresses in MSDOS.SYS or RETRODOS.SYS)
20812                                ; ====================================================================
20813                                ; 21/03/2019 - Retro DOS v4.0
20814                                ; 21/10/2022 - Retro DOS v4.0 (MOdified MSDOS 5.0 IO.SYS)
20815
20816                                ;KERNEL_SEGMENT    equ 0070h  ; (IO.SYS loading segment, BIOS_DATA segment)
20817                                ; 21/10/2022
20818                                DOSBIODATASEG equ 0070h    ; (IO.SYS loading segment, BIOS_DATA segment)
20819                                ; 22/10/2022
20820                                ;DOSBIOCODESEG equ 02C7h ; (MSDOS 5.0 IO.SYS, BIOS_CODE segment)
20821                                ; 09/12/2022
20822                                DOSBIOCODESEG equ IOSYSCODESEG
20823
20824                                ; Note: These offset addresses must be changed when the code
20825                                ;   in retrodos4.s (MSDOS.SYS) file will be changed.
20826
20827                                ; (following addresses can be verified by searching them in retrodos4.lst)
20828
20829                                ; 09/12/2022
20830                                %if 0
20831
```

```
                        ; 13/05/2019
                        
                        ;IsWin386         equ 08CFh
                        ;V86_Crit_SetFocus equ 08D0h
                        ; 21/10/2022
                        IsWin386          equ 08D0h
                        V86_Crit_SetFocus equ 08D1h
                        
                        ;seg_reinit  equ 0772h ; not used in Retro DOS v4.0
                        ; 21/10/2022 - Retro DOS v4.0 (MOdified MSDOS 5.0 IO.SYS)
                        seg_reinit   equ 0032h ; DOSBIOCODESEG:0032h
                        
                        ;SysinitPresent      equ 08FCh
                        ; 21/10/2022
                        SysinitPresent       equ 08FDh
                        
                        inHMA                equ 000Dh
                        xms           equ 000Eh
                        ;FreeHMAPtr   equ 08F6h
                        ;multrk_flag         equ 0533h
                        ;ec35_flag    equ 0535h
                        ;EOT          equ 012Eh
                        ; 21/10/2022
                        FreeHMAPtr    equ 08F7h
                        multrk_flag   equ 052Fh
                        ec35_flag     equ 0531h
                        EOT           equ 012Ch
                        
                        ;NextStack    equ 08BFh
                        ;IT_StackLoc         equ 08C5h
                        ;IT_StackSize        equ 08C9h
                        ; 21/10/2022
                        NextStack     equ 08C0h
                        IT_StackLoc   equ 08C6h
                        IT_StackSize         equ 08CAh
                        
                        ;MoveDOSIntoHMA      equ 08F8h
                        ; 21/10/2022
                        MoveDOSIntoHMA       equ 08F9h
                        
                        ;INT19SEM equ 0644h ; 01/05/2019 - retrodos4.lst
                        ;I19_LST  equ 0645h ; 27/03/2019 - retrodos4.lst
                        ; 21/10/2022
                        INT19SEM equ 0640h ; (iosys5.txt)
                        I19_LST  equ 0641h ; (iosys5.txt)
                        
                        %endif
                        
                        ; 09/12/2022
                        seg_reinit equ _seg_reinit
                        ec35_flag  equ ec35flag
                        INT19SEM   equ int19sem
                        I19_LST    equ i19_lst
                        
                        INT19OLD02 equ I19_LST+1 ; 0642h ; 21/10/2022
                        INT19OLD08 equ I19_LST+6
                        INT19OLD09 equ I19_LST+11
                        INT19OLD0A equ I19_LST+16
                        INT19OLD0B equ I19_LST+21
                        INT19OLD0C equ I19_LST+26
                        INT19OLD0D equ I19_LST+31
                        INT19OLD0E equ I19_LST+36
                        INT19OLD70 equ I19_LST+41
                        INT19OLD72 equ I19_LST+46
                        INT19OLD73 equ I19_LST+51
                        INT19OLD74 equ I19_LST+56
                        INT19OLD76 equ I19_LST+61
                        INT19OLD77 equ I19_LST+66 ; 0683h ; 21/10/2022
                        
                        ; 09/12/2022
                        %if 0
                        
                        ;keyrd_func equ 04E9h
                        ;keysts_func       equ 04EAh
                        ;t_switch   equ 04F6h
                        ; 21/10/2022
                        keyrd_func  equ 04E5h
                        keysts_func equ 04E6h
                        t_switch    equ 04F2h
                        
                        ; 22/10/2022
                        SYSINITSEG  equ 046Dh  ; SYSINIT segment
                        BCODE_END   equ (SYSINITSEG-DOSBIOCODESEG)*16 ; = 1A60h
                        BCODE_START equ 30h  ; (offset BiosDataWord in DOSBIOCODESEG)
                        RE_INIT            equ 089Bh ; (re_init offset in DOSBIODATASEG)
                        
                        %endif
                        
                        ; 09/12/2022
                        BCODESTART  equ BIOSDATAWORD
                        RE_INIT            equ re_init
                        
                        ; ----------------------------------------------------------------------
                        ; CONFIG.INC (MSDOS 6.0 - 1991)
                        ; ----------------------------------------------------------------------
                        ; 15/04/2019 - Retro DOS v4.0
                        
                        CONFIG_BEGIN        equ  '['
                        CONFIG_BREAK        equ  'C'
                        CONFIG_BUFFERS      equ  'B'
                        CONFIG_COMMENT      equ  'Y'
                        CONFIG_COUNTRY      equ  'Q'
                        CONFIG_DEVICE       equ  'D'
                        CONFIG_DEVICEHIGH   equ  'U'
                        CONFIG_DOS          equ  'H'
                        CONFIG_DRIVPARM     equ  'P'
                        CONFIG_FCBS         equ  'X'
                        CONFIG_FILES        equ  'F'
                        CONFIG_INCLUDE      equ  'J'
                        CONFIG_INSTALL      equ  'I'
                        CONFIG_INSTALLHIGH  equ  'W'
                        CONFIG_LASTDRIVE    equ  'L'
                        CONFIG_MENUCOLOR    equ  'R'
                        CONFIG_MENUDEFAULT  equ  'A'
                        CONFIG_MENUITEM     equ  'E'
                        CONFIG_MULTITRACK   equ  'M'
                        CONFIG_NUMLOCK      equ  'N'
                        CONFIG_REM          equ  'O'
                        CONFIG_SEMICOLON    equ  ';'
                        CONFIG_SET          equ  'V'
                        CONFIG_SHELL        equ  'S'
                        CONFIG_STACKS       equ  'K'
                        CONFIG_SUBMENU      equ  'O'
                        CONFIG_SWITCHES     equ  '1'
```

```
20956
20957                           CONFIG_UNKNOWN      equ  'Z'
20958
20959                           ; 13/05/2024 - Retro DOS v5.0 (PCDOS 71 IBMBIO.COM)
20960                           CONFIG_DOSDATA      equ  'T'
20961
20962                           CONFIG_OPTION_QUERY equ 80h
20963
20964                           ; ----------------------------------------------------------------------
20965                           ; SYSINIT1.ASM (MSDOS 6.0 - 1991)
20966                           ; ----------------------------------------------------------------------
20967                           ; 21/03/2019 - Retro DOS v4.0
20968
20969                           trueequ    0FFFFh
20970                           false      equ    0
20971                           cr  equ    13
20972                           lf  equ    10
20973                           tab equ    9
20974
20975                           multMULT      equ 4Ah
20976                           multMULTGETHMAPTR  equ    1
20977                           multMULTALLOCHMA   equ    2
20978
20979                           ;NOEXEC    equ     FALSE
20980
20981                           stacksw    equ     true   ;include switchable hardware stacks
20982                           mycds_size equ     88     ;size of curdir_list. if it is not
20983                                              ;the same, then will generate compile error.
20984
20985                           entrysize   equ     8
20986
20987                           mincount    equ     8
20988                           defaultcount equ    9
20989                           maxcount    equ     64
20990
20991                           minsize     equ     32
20992                           defaultsize equ     128
20993                           maxsize     equ     512
20994
20995                           ;%define allocbyte  byte [es:bp+0]
20996                           ;%define intlevel   byte [es:bp+1]
20997                           ;%define savedsp    word [es:bp+2]
20998                           ;%define savedss    word [es:bp+4]
20999                           ;%define newsp        word [es:bp+6]
21000
21001                           allocbyte   equ     0
21002                           intlevel    equ     1
21003                           savedsp     equ     2
21004                           savedss     equ     4
21005                           newsp       equ     6
21006
21007                           free    equ     0
21008                           allocated   equ     1
21009                           overflowed  equ     2
21010                           clobbered   equ     3
21011
21012                           ;---------------------------------------
21013                           ; external variable defined in ibmbio module for multi-track
21014
21015                           multrk_on equ      10000000b ;user specified mutitrack=on,or system turns
21016                                                ; it on after handling config.sys file as a
21017                                                ; default value,if multrk_flag = multrk_off1.
21018                           multrk_off1 equ 00000000b ;initial value. no "multitrack=" command entered.
21019                           multrk_off2 equ 00000001b ;user specified multitrack=off.
21020
21021                           ; SYSINITSEG       SEGMENT PUBLIC 'SYSTEM_INIT'
21022
21023                           SYSINIT$:
21024                               ;IF    STACKSW
21025                               ; include MSSTACK.INC ;Main stack program and data definitions
21026                               ; include STKMES.INC  ;Fatal stack error message
21027                               ;   public Endstackcode
21028                           ;Endstackcode      label byte
21029                               ;ENDIF
21030
21031                           ; 05/07/2018
21032                           ; ----------------------------------------------------------------------
21033                           ; 04/06/2018 - Retro DOS v3.0
21034
21035                           ; ----------------------------------------------------------------------
21036                           ; 21/10/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 IO.SYS - SYSINIT)
21037                           ; ----------------------------------------------------------------------
21038
21039                           ;   MSStack.inc
21040                           ;
21041                           ;   Interrupt level 2, 3, 4, 5, 6, 7,(10, 11, 12, 14, 15 - AT level)
21042                           ;   should follow the standard Interrupt Sharing Scheme which has
21043                           ;   a standard header structure.
21044                           ;   Fyi, the following shows the relations between
21045                           ;   the interrupt vector and interrupt level.
21046                           ; VEC(Hex)    2  8  9  A  B  C    D  E  70  72  73  74  76  77
21047                           ; LVL(Deci)   9  0  1  2  3  4    5  6   8  10  11  12  14  15
21048                           ;   MSSTACK module modifies the following interrupt vectors
21049                           ;   to meet the standard Interrupt Sharing standard;
21050                           ;     A, B, C, D, E, 72, 73, 74, 76, 77.
21051                           ;   Also, for interrupt level 7 and 15, the FirstFlag in a standard header
21052                           ;   should be initialized to indicat whether this interrupt handler is
21053                           ;   the first (= 80h) or not.  The FirstFlag entry of INT77h's
21054                           ;   program header is initialized in STKINIT.INC module.
21055                           ;   FirstFlag is only meaningful for interrupt level 7 and 15.
21056                           ;
21057
21058                           ;  User specifies the number of stack elements - default = 9
21059                           ;                                         minimum = 8
21060                           ;                                         maximum = 64
21061                           ;
21062                           ;  Intercepts Asynchronous Hardware Interrupts only
21063                           ;
21064                           ;  Picks a stack from pool of stacks and switches to it
21065                           ;
21066                           ;  Calls the previously saved interrupt vector after pushing flags
21067                           ;
21068                           ;  On return, returns the stack to the stack pool
21069                           ;
21070
21071                           ; This is a modification of STACKS:
21072                           ; 1. To fix a bug which was causing the program to take up too much space.
21073                           ; 2. To dispense stack space from hi-mem first rather than low-mem first.
21074                           ;    . Clobbers the stack that got too big instead of innocent stack
21075                           ;    . Allows system to work if the only stack that got too big was the most
21076                           ;      deeply nested one
21077                           ; 3. Disables NMI interrupts while setting the NMI vector.
21078                           ; 4. Does not intercept any interrupts on a PCjr.
21079                           ; 5. Double checks that a nested interrupt didn't get the same stack.
```

```
                                ; 6. Intercepts Ints 70, 72-77 for PC-ATs and other future products
                                
                                        ;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
                                ;int&a      endp
                                ;   endm

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

                                int0A:
00000057 EB10                       jmp     short entry_int0A_stk
00000059 00000000               old0A:      dd      0
0000005D 4B42                       dw      424Bh
                                firstflag0A:
0000005F 00                         db      0
00000060 EB0C                       jmp     short intret_0A
00000062 00<rep 7h>                 times   7 db 0

                                entry_int0A_stk:
00000069 E8DB00                     call    do_int_stacks
0000006C [5900]                     dw      old0A
                                intret_0A:
0000006E CF                         iret

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

                                int0B:
0000006F EB10                       jmp     short entry_int0B_stk
00000071 00000000               old0B:      dd      0
00000075 4B42                       dw      424Bh
                                firstflag0B:
00000077 00                         db      0
00000078 EB0C                       jmp     short intret_0B
0000007A 00<rep 7h>                 times   7 db 0

                                entry_int0B_stk:
00000081 E8C300                     call    do_int_stacks
00000084 [7100]                     dw      old0B
                                intret_0B:
00000086 CF                         iret

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

                                int0C:
00000087 EB10                       jmp     short entry_int0C_stk
00000089 00000000               old0C:      dd      0
0000008D 4B42                       dw      424Bh
                                firstflag0C:
0000008F 00                         db      0
00000090 EB0C                       jmp     short intret_0C
00000092 00<rep 7h>                 times   7 db 0

                                entry_int0C_stk:
00000099 E8AB00                     call    do_int_stacks
0000009C [8900]                     dw      old0C
                                intret_0C:
0000009E CF                         iret

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

                                int0D:
0000009F EB10                       jmp     short entry_int0D_stk
000000A1 00000000               old0D:      dd      0
000000A5 4B42                       dw      424Bh
                                firstflag0D:
000000A7 00                         db      0
000000A8 EB0C                       jmp     short intret_0D
000000AA 00<rep 7h>                 times   7 db 0

                                entry_int0D_stk:
000000B1 E89300                     call    do_int_stacks
000000B4 [A100]                     dw      old0D
                                intret_0D:
000000B6 CF                         iret

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

                                int0E:
000000B7 EB10                       jmp     short entry_int0E_stk
000000B9 00000000               old0E:      dd      0
000000BD 4B42                       dw      424Bh
                                firstflag0E:
000000BF 00                         db      0
000000C0 EB0C                       jmp     short intret_0E
000000C2 00<rep 7h>                 times   7 db 0

                                entry_int0E_stk:
000000C9 E87B00                     call    do_int_stacks
000000CC [B900]                     dw      old0E
                                intret_0E:
000000CE CF                         iret

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

                                int72:
000000CF EB10                       jmp     short entry_int72_stk
000000D1 00000000               old72:      dd      0
000000D5 4B42                       dw      424Bh
                                firstflag72:
000000D7 00                         db      0
000000D8 EB0C                       jmp     short intret_72
000000DA 00<rep 7h>                 times   7 db 0

                                entry_int72_stk:
000000E1 E86300                     call    do_int_stacks
000000E4 [D100]                     dw      old72
                                intret_72:
000000E6 CF                         iret

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

                                int73:
000000E7 EB10                       jmp     short entry_int73_stk
000000E9 00000000               old73:      dd      0
000000ED 4B42                       dw      424Bh
                                firstflag73:
000000EF 00                         db      0
000000F0 EB0C                       jmp     short intret_73
000000F2 00<rep 7h>                 times   7 db 0

                                entry_int73_stk:
000000F9 E84B00                     call    do_int_stacks
000000FC [E900]                     dw      old73
                                intret_73:
000000FE CF                         iret

```

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

                                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]

0000018B 9C                         pushf                           ; go execute the real interrupt handler
                                    ; 11/12/2022
0000018C 2EFF5E00                   call    far [cs:bp]             ;  which will iret back to here
                                    ; 21/10/2022
                                    ;call   far [cs:bp+0]

00000190 89E5                       mov     bp,sp                   ; retrieve the table offset for us
                                    ; 11/12/2022
00000192 268B6E00                   mov     bp,[es:bp]              ;  but leave it on the stack
                                    ; 21/10/2022
                                    ;mov    bp,[es:bp+0]
00000196 268E5604                   mov     ss,[es:bp+savedss]      ; get old stack back
0000019A 268B6602                   mov     sp,[es:bp+savedsp]

                                    ; 11/12/2022
                                    ;mov    byte [es:bp+allocbyte],free ; free the entry
                                    ; 21/10/2022
0000019E 26C6460000                 mov     byte [es:bp],free ; 0
000001A3 2E892E[1000]               mov     [cs:nextentry],bp       ; setup to use next time

000001A8 07                         pop     es
000001A9 5D                         pop     bp                      ; saved on entry
000001AA 58                         pop     ax                      ; saved on entry
000001AB 83C402                     add     sp,2
000001AE CF                         iret                            ; done with this interrupt

                                notfree02:
000001AF 3C01                       cmp     al,allocated            ; error flag
000001B1 7404                       je      short findnext02        ;  no, continue
                                    ; 11/12/2022
                                    ;xchg   [es:bp+allocbyte],al    ;  yes, restore error value
                                    ; 21/10/2022
000001B3 26864600                   xchg    [es:bp],al

                                findnext02:
000001B7 E81200                     call    longpath
000001BA EBA8                       jmp     short found02

                                foundbad02:
000001BC 2E3B2E[0C00]               cmp     bp,[cs:firstentry]
000001C1 72F4                       jc      short findnext02
000001C3 89C5                       mov     bp,ax                   ; flag this entry
                                    ; 11/12/2022
                                    ;mov    byte [es:bp+allocbyte],clobbered
                                    ; 21/10/2022
000001C5 26C6460003                 mov     byte [es:bp],clobbered ; 3
000001CA EBEB                       jmp     short findnext02        ; keep looking

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

                                ; Common routines

                                longpath:
                                    ; 21/03/2019
000001CC 2E8B2E[0E00]               mov     bp,[cs:lastentry]       ; start with last entry in table
                                lploopp:
                                    ; 11/12/2022
                                    ;cmp    byte [es:bp+allocbyte],free ; is entry free?
                                    ; 21/10/2022
000001D1 26807E0000                 cmp     byte [es:bp],free
000001D6 7512                       jne     short inuse             ;  no, try next one

000001D8 B001                       mov     al,allocated
                                    ; 11/12/2022
                                    ;xchg   [es:bp+allocbyte],al    ; allocate entry
                                    ; 21/10/2022
000001DA 26864600                   xchg    [es:bp],al
000001DE 3C00                       cmp     al,free                 ; is it still free?
000001E0 7414                       je      short found             ;  yes, go use it

000001E2 3C01                       cmp     al,allocated            ; is it other than Allocated or Free?
000001E4 7404                       je      short inuse             ;  no, check the next one

                                    ; 11/12/2022
                                    ;mov    [es:bp+allocbyte],al    ;  yes, put back the error state
                                    ; 21/10/2022
000001E6 26884600                   mov     [es:bp],al
                                inuse:
000001EA 2E3B2E[0C00]               cmp     bp,[cs:firstentry]
000001EF 7406                       je      short fatal
000001F1 83ED08                     sub     bp,entrysize
000001F4 EBDB                       jmp     short lploopp
                                found:
000001F6 C3                         retn
                                fatal:
000001F7 1E                         push    ds
000001F8 B800F0                     mov     ax,0F000h               ;look at the model byte
000001FB 8ED8                       mov     ds,ax
000001FD 803EFEFFF9                 cmp     byte [0FFFEh],0F9h ; mdl_convert ; convertible?
00000202 1F                         pop     ds
00000203 7504                       jne     short skip_nmis

00000205 B007                       mov     al,07h                  ; disable pc convertible nmis
00000207 E672                       out     72h,al

                                skip_nmis:
00000209 FA                         cli                             ; disable and mask
0000020A B0FF                       mov     al,0FFh                 ;   all other ints
0000020C E621                       out     021h,al
0000020E E6A1                       out     0A1h,al

00000210 8CCE                       mov     si,cs
00000212 8EDE                       mov     ds,si
00000214 BE[3B02]                   mov     si,fatal_msg
                                ;SR;
                                ;   We set all foci to this VM to issue the stack failure message
                                ;
00000217 50                         push    ax
00000218 1E                         push    ds
                                    ;;mov   ax,Bios_Data ; 0070h
                                    ;mov    ax,KERNEL_SEGMENT ; 0070h
                                    ; 21/10/2022
00000219 B87000                     mov     ax,DOSBIODATASEG
0000021C 8ED8                       mov     ds,ax

                                    ;test   byte [08D0h],1          ; (MSDOS 6.21, IO.SYS - SYSINIT:021Eh)
0000021E F606[1208]01               test    byte [IsWin386],1 ; (retrodos4.sys, offset: ****h)
00000223 1F                         pop     ds
00000224 58                         pop     ax
00000225 7405                       jz      short fatal_loop        ; win386 not present, continue

                                    ;;call  far ptr 0070h:08D1h ; (MSDOS 621, IO.SYS - SYSINIT:0227h)
```

```
                                    ;call  KERNEL_SEGMENT:V86_Crit_SetFocus ; set focus to this VM
                                    ; 21/10/2022
00000227 9A[1308]7000               call   DOSBIODATASEG:V86_Crit_SetFocus ; 0070h:08D1h
                                ;
                                ;SR; We do not bother about the returned status of this call.
                                ;
                                fatal_loop:
0000022C AC                         lodsb
0000022D 3C24                       cmp    al,'$'
0000022F 7408                       je     short fatal_done

00000231 B307                       mov    bl,7
00000233 B40E                       mov    ah,14
00000235 CD10                       int    10h                 ; whoops, this enables ints
00000237 EBF3                       jmp    short fatal_loop

                                fatal_done:
00000239 EBFE                       jmp    short fatal_done


                                ; 21/03/2019 - Retro DOS v4.0 (MSDOS 6.0, SYSINIT1.ASM, 1991)
                                ; ---------------------------------------------------------------------------
                                ;   include msbio.cl5           ; fatal stack error message

                                ; MSDOS 6.21, IO.SYS, SYSINIT:023Bh

                                ; STKMES.INC - MSDOS 3.3 (24/07/1987)
                                ; ---------------------------------------------------------------------------
                                ; 04/06/2018 - Retro DOS v3.0

                                fatal_msg:
0000023B 0D0A                       db     0Dh,0Ah
0000023D 070D0A                     db     7,0Dh,0Ah
00000240 496E7465726E616C20-        db     "Internal stack overflow",0Dh,0Ah
00000249 737461636B206F7665-
00000252 72666C6F770D0A
00000259 53797374656D206861-        db     "System halted",0Dh,0Ah,"$"
00000262 6C7465640D0A24

                                endstackcode:

                                ; ---------------------------------------------------------------------------
                                ; SYINIT1.ASM (MSDOS 6.0, 1991) 'SYSINIT' jump addr from 'MSINIT.ASM'
                                ; ---------------------------------------------------------------------------
                                ; 04/06/2018 - Retro DOS v3.0 (MSDOS 3.3, SYSINIT1.ASM, 24/07/1987)

                                ; 22/03/2019 - Retro DOS v4.0

                                ; SYSINIT:0269h (MSDOS 6.21 IO.SYS, SYSINIT segment, offset: 0269h)

                                ; ('SYSINIT:' location/address is used in 'retrodos4.s'. If following
                                ; address will be changed, it must also be changed in 'retrodos4.s'.)

                                ; 21/10/2022- Retro DOS v4.0 (Modified MSDOS 5.0 IO.SYS)
                                ; ---------------------------------------------------------------------------
                                ; SYSINITSEG:0267h (MSDOS 5.0 IO.SYS, SYSINIT segment, offset: 0267h)

                                ; SYSINIT:0269h (MSDOS 6.22 IO.SYS, SYSINIT segment, offset: 0269h)

                                ; 29/12/2023- Retro DOS v5.0 (Modified PCDOS 7.1 IBMBIO.COM)
                                ; ---------------------------------------------------------------------------
                                ; SYSINITSEG:0269h (PCDOS 7.1 IBMBIO.COM, SYSINIT segment, offset: 0269h)

                                SYSINIT:
00000269 E9AD01                         JMP GOINIT
                                    ;JMP    SYSIN ; 25/02/2018 - Retro DOS 2.0 modification

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

                                struc DDHighInfo
00000000 ????????                .ddhigh_CSegPtr resd 1    ; pointer to code segment to be relocated
00000004 ????                    .ddhigh_CSegLen resw 1    ; length of code segment to be relocated
00000006 ????????                .ddhigh_CallBak resd 1    ; pointer to the call back routine
                                endstruc

                                ; 22/03/2019 - Retro DOS v4.0

0000026C 00                     runhigh: db 0

                                ; 02/11/2022
                                ;align 4

                                DOSINFO:
0000026D 00000000                   dd     0      ; address of the DOS Sysini Variables
                                ;MSDOS:
                                dos_temp_location: ; dword ; MSDOS 6.0
                                dosinit:           ; MSDOS 6.0
00000271 0000                       dw     0

                                ; 02/11/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
                                ;FINAL_DOS_LOCATION: ; 20/04/2019 - Retro DOS v4.0
                                ;   dw     0
                                ;MSDOS 5.0 IO.SYS - SYSINIT:0271h

                                CURRENT_DOS_LOCATION:
00000273 0000                       dw     0

                                ;DOSSIZE: ; Retro DOS 2.0 feature - 25/02/2018
                                ;   dw     0   ; 'MSDOS.BIN' kernel size in words

                                ; 22/10/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 IO.SYS)
                                ; (MSDOS 5.0 MSDOS.SYS size is 37394 bytes)
                                ;DOSSIZE equ       0A000h  ; (MSDOS 6.0 - SYSINIT1.ASM - 1991)
                                ; 30/12/2023 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMBIO.COM)
                                ; 03/09/2023 (PCDOS 7.1 IBMDOS.COM size is 42566 bytes, 04/12/2003)
                                DOSSIZE equ 0B000h  ; (PCDOS 7.1 - SYSINIT)

                                DEVICE_LIST:
00000275 00000000                   dd     0

                                ; 04/06/2018 - Retro DOS v3.0
                                ; 28/03/2018
                                ;; MSDOS 3.3 - SYSINIT1.ASM - 24/07/1987
                                ;
                                sysi_country:
00000279 00000000                   dd     0 ; 5/29/86 Pointer to country table in DOS

                                ; MSDOS 6.0
0000027D 00000000               dos_segreinit:     dw     0,0    ; room for dword

00000281 0000                   lo_doscod_size:    dw     0      ; dos code size when in low mem
00000283 0000                   hi_doscod_size:    dw     0      ; dos code size when in HMA

00000285 0000                   def_php:    dw     0
```

```
                                ; 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
```

```
21821 000002D6 00000000               sysinit_ptr:       dd     0              ; returning addr. from sysinit_base
21822 000002DA 0000                   checksum:   dw     0              ; used by sum_up
21823
21824 000002DC 20<rep 14h>            ldexec_fcb: times 20 db 20h ; db 20 dup (' ') ;big enough
21825 000002F0 00                     ldexec_line:       db     0              ;# of parm characters
21826 000002F1 20                     ldexec_start:      db     ' '
21827 000002F2 00<rep 50h>            ldexec_parm:       times 80 db 0  ; db 80 dup (0)
21828
21829                                 ;instexe exec0     <0,ldexec_line,ldexec_fcb,ldexec_fcb>
21830
21831                                 instexe:
21832 00000342 0000                   iexec.environ:     dw     0              ; seg addr of environment
21833 00000344 [F002]                 iexec.ldexec_line: dw     ldexec_line ; pointer to asciz command line
21834 00000346 0000                               dw     0      ; SYSINIT segment (0473h for MSDOS 6.21 IO.SYS)
21835                                                            ; SYSINIT segment (0544h for PCDOS 7.1 IBMBIO.COM)
21836 00000348 [DC02]                 iexec.ldexec_5c_fcb: dw   ldexec_fcb     ; default fcb at 5C
21837 0000034A 0000                               dw     0      ; SYSINIT segment (0473h for MSDOS 6.22 IO.SYS)
21838 0000034C [DC02]                 iexec.ldexec_6c_fcb: dw   ldexec_fcb     ; default fcb at 6C
21839 0000034E 0000                               dw     0
21840
21841                                 ; variables for comment=
21842
21843 00000350 00                     com_level:  db     0              ; level of " " in command line
21844 00000351 00                     cmmt:              db     0              ; length of comment string token
21845 00000352 00                     cmmt1:             db     0              ; token
21846 00000353 00                     cmmt2:             db     0              ; token
21847 00000354 00                     cmd_indicator:     db     0
21848 00000355 00                     donotshownum:      db     0
21849
21850 00000356 0000                   count:             dw     0
21851 00000358 0000                   org_count:  dw     0
21852 0000035A 0000                   chrptr:            dw     0
21853 0000035C 0000                   cntryfilehandle: dw       0
21854 0000035E 0000                   old_area:   dw     0
21855 00000360 0000                   impossible_owner_size: dw 0              ; paragraph
21856
21857                                 bucketptr: ; label dword
21858                                 bufptr:       ; label dword              ; leave this stuff in order!
21859 00000362 0000                   memlo:      dw     0
21860                                 prmblk:       ; label word
21861 00000364 0000                   memhi:      dw     0
21862 00000366 0000                   ldoff:      dw     0
21863 00000368 0000                   area:       dw     0
21864
21865                                 ; 29/12/2023 - PCDOS 7.1 IBMBIO.COM - SYSINIT:036Ah
21866 0000036A 0000                   prev_memhi: dw 0
21867 0000036C 0000                   prev_alloclim:     dw 0
21868 0000036E 00                     dosdata_umb:       db 0
21869
21870                                 ; Following is the request packet used to call INIT routines for
21871                                 ; all device drivers. Some fields may be accessed individually in
21872                                 ; the code, and hence have individual labels, but they should not
21873                                 ; be separated.
21874
21875 0000036F 19                     packet:     db     25                     ; PCDOS 7.1 IBMBIO.COM
21876                                     ;db     24                     ; was 22
21877 00000370 00                       db     0
21878 00000371 00                       db     0                      ; initialize code
21879 00000372 0000                     dw     0
21880 00000374 00<rep 8h>               times  8 db 0  ; db 8 dup (?)
21881
21882 0000037C 00                     unitcount:  db     0
21883 0000037D 00000000               break_addr: dd     0
21884 00000381 00000000               bpb_addr:   dd     0
21885                                 drivenumber:       ; 22/10/2022
21886 00000385 00                     devdrivenum:       db     0
21887 00000386 0000                   configmsgflag:     dw     0  ; used to control "error in config.sys line #" message
21888
21889                                 ; end of request packet
21890
21891                                 ;drivenumber:      db     0  ; 22/03/2019
21892
21893                                 toomanydrivesflag:
21894 00000388 00                               db     0  ; >24 fixed disk partitions flag ; M029
21895 00000389 90                     align 2
21896
21897                                 BCodeSeg:   ; 21/10/2022
21898 0000038A 0203                     dw     DOSBIOCODESEG ; (02C7h for MSDOS 5.0 IO.SYS)
21899                                                    ; 0364h for PCDOS 7.1 IBMBIO.COM - 29/12/2023
21900                                     ;dw     Bios_Code  ; = KERNEL_SEGMENT = 0070h (for Retro DOS v4.0)
21901                                                      ; BCodeSeg = 2CCh (for MSDOS 6.21 IO.SYS)
21902
21903                                 ; 30/12/2022
21904                                 ; MSDOS 6.21 IO.SYS, SYSINIT:0387h
21905                                 ;
21906                                 ; Magicbackdoor: dd 0
21907                                 ; NullBackdoor:
21908                                 ;           retf
21909
21910                                 ; 02/11/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 IO.SYS)
21911                                 ; 01/01/2023 - Retro DOS v4.2 (Modified MSDOS 6.21 IO.SYS)
21912                                 ; 19/04/2019
21913                                 _timer_lw_:
21914 0000038C 0000                     dw     0  ; MSDOS 6.21 IO.SYS - SYSINIT:038Ch
21915
21916                                 ; 29/12/2023 - Retro DOS v5.0
21917                                 ; PCDOS 7.1 IBMBIO.COM - SYSINIT:038Eh
21918
21919 0000038E 00                     F5_key:     db 0
21920 0000038F 00                     F8_key:     db 0
21921 00000390 00000000               MagicBackdoor:     dd 0
21922                                 NullBackdoor:
21923 00000394 CB                               retf
21924
21925                                 ;SR;
21926                                 ; This is the communication block between the DOS and the BIOS. It starts at
21927                                 ;the SysinitPresent flag. Any other data that needs to be communicated
21928                                 ;to the DOS should be added after SysinitPresent. The pointer to this block
21929                                 ;is passed to DOS as part of the DOSINIT call.
21930                                 ;
21931
21932                                 BiosComBlock:
21933                                     ;dd     Bios_Data:SysinitPresent
21934                                             ; 0070h:08FDh for MSDOS 6.21 IO.SYS
21935 00000395 [DD07]                   dw     SysinitPresent  ; (retrodos4.sys, offset: ****h)
21936                                     ;dw     KERNEL_SEGMENT ; 0070h
21937                                     ; 21/10/2022
21938 00000397 7000                     dw     DOSBIODATASEG ; 0070h
21939
21940                                 ;align 2
21941
21942                                     ; 22/10/2022 - (MSDOS 5.0 IO.SYS, SYSINIT:0406h)
21943                                     ; 30/12/2022 - (MSDOS 6.21 IO.SYS, SYSINIT:0392h)
21944                                 tempstack:
```

```
21945 00000399 00<rep 80h>                  times  128 db 0  ; db 80h dup (?)
21946
21947                                   ; ----------------------------------------------------------------------------
21948
21949                                       ; 29/12/2023 - Retro DOS v5.0
21950                                       ; 22/10/2022 - Retro DOS v4.0
21951                                       ;      ; (MSDOS 5.0 IO.SYS, SYSINIT:0486h)
21952                                   GOINIT:            ; (MSDOS 6.22 IO.SYS, SYSINIT:0412h)
21953                                       ; (PCDOS 7.1 IBMBIO.COM, SYSINIT:0419h)
21954                                       ; 12/12/2023
21955 00000419 0E                           push    cs
21956 0000041A 1F                           pop     ds
21957
21958                                       ; 12/12/2022
21959                                       ; 22/03/2019 - Retro DOS v4.0
21960                                       ; 06/07/2018
21961                                       ; 04/06/2018 - Retro DOS v3.0
21962                                   ; before doing anything else, let's set the model byte
21963 0000041B B4C0                         mov     ah,0C0h                 ; get system configuration
21964 0000041D CD15                         int     15h                     ;
21965 0000041F 7214                         jc      short no_rom_config
21966
21967                                       ;cmp    ah,0                    ; double check
21968                                       ;jne    short no_rom_config
21969                                       ; 03/09/2023
21970 00000421 08E4                         or      ah,ah
21971 00000423 7510                         jnz     short no_rom_config
21972
21973                                       ; 12/12/2023 ; *
21974                                       ; ds = cs
21975
21976 00000425 268A4702                     mov     al,[es:bx+ROMBIOS_DESC.bios_sd_modelbyte]
21977                                       ;mov    [cs:sys_model_byte],al
21978 00000429 A2[BB02]                     mov     [sys_model_byte],al ; *
21979 0000042C 268A4703                     mov     al,[es:bx+ROMBIOS_DESC.bios_sd_scnd_modelbyte]
21980                                       ;mov    [cs:sys_scnd_model_byte],al
21981 00000430 A2[BC02]                     mov     [sys_scnd_model_byte],al ; *
21982                                       ;jmp    short SYSIN
21983                                       ; 02/11/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
21984 00000433 EB29                         jmp     short move_myself
21985
21986                                   no_rom_config:                          ; Old ROM
21987                                       ; 12/12/2023
21988                                       ;mov    ax,0F000h
21989                                       ;mov    ds,ax
21990                                       ;mov    al,[0FFFEh]
21991                                       ;mov    [cs:sys_model_byte],al; set the model byte.
21992                                       ; 12/12/2023
21993                                       ; ds = cs
21994 00000435 B800F0                       mov     ax,0F000h
21995 00000438 8EC0                         mov     es,ax
21996 0000043A 26A0FEFF                     mov     al,[es:0FFFEh]
21997 0000043E A2[BB02]                     mov     [sys_model_byte],al   ; set the model byte.
21998
21999                                   ; set fake_floppy_drv if there is no diskette drives in this machine.
22000                                   ; execute the equipment determination interrupt and then
22001                                   ; check the returned value to see if we have any floppy drives
22002                                   ; if we have no floppy drive we set cs:fake_floppy_drv to 1
22003                                   ; see the at tech ref bios listings for help on the equipment
22004                                   ; flag interrupt (11h)
22005
22006                                       ; 22/10/2022
22007                                   ;check_for_fake_floppy:                 ; entry point for rom_config above
22008 00000441 CD11                         int     11h                     ; check equipment flag
22009
22010                                       ; 29/12/2023 - Retro DOS v5.0
22011                                       ;jmp    short check_for_fake_floppy
22012                                       ; PCDOS 7.1 IBMBIO.COM - SYSINIT:0446h
22013                                       ;db     52h     ; 'RPS' sign
22014                                       ;db     50h
22015                                       ;db     53h
22016
22017                                   check_for_fake_floppy:
22018                                       ; 29/12/2023
22019                                       ; PCDOS 7.1 IBMBIO.COM - SYSINIT:0449h
22020                                       ;or     ax, 1           ; (nonsense! this may be overwritten/disabled
22021                                       ;                       ; by using 'RPS' sign position)
22022                                       ;                       ;               03/07/2023 - Erdogan Tan
22023                                       ;test   ax, 1           ; have any floppies?
22024
22025                                       ; 12/12/2022
22026 00000443 A801                         test    al,1
22027                                       ;test   ax,1                    ; have any floppies?
22028 00000445 7517                         jnz     short move_myself       ; yes,normal system
22029
22030                                   ; Some ROM BIOSs lie that there are no floppy drives. Lets find out
22031                                   ; whether it is an old ROM BIOS or a new one
22032                                   ;
22033                                   ; WARNING !!!
22034                                   ;
22035                                   ; This sequence of code is present in MSINIT.ASM also. Any modification
22036                                   ; here will require an equivalent modification in MSINIT.ASM also
22037
22038                                       ; 12/12/2023
22039                                       ;push   es  ; not necessary
22040
22041 00000447 30C9                         xor     cl,cl
22042 00000449 B408                         mov     ah,8                    ; get disk parameters
22043 0000044B B200                         mov     dl,0                    ; of drive 0
22044 0000044D CD13                         int     13h
22045
22046                                       ;pop    es  ; 12/12/2023
22047
22048 0000044F 720D                         jc      short move_myself       ; if error lets assume that the
22049                                                                       ;  ROM BIOS lied
22050                                       ;cmp    cl,0                    ; double check (max sec no cannot be 0)
22051                                       ;je     short move_myself
22052                                       ; 03/09/2023
22053 00000451 08C9                         or      cl,cl
22054 00000453 7409                         jz      short move_myself
22055
22056 00000455 08D2                         or      dl,dl                   ; number of flp drvs == 0?
22057 00000457 7505                         jnz     short move_myself       ; no
22058
22059                                       ;mov    byte [cs:fake_floppy_drv],1 ; set fake flag.
22060                                       ; 12/12/2023
22061                                       ; ds = cs
22062 00000459 C606[8B02]01                 mov     byte [fake_floppy_drv],1 ; set fake flag.
22063
22064                                   move_myself:
22065                                       ; 12/12/2023
22066                                       ;cld    ; not necessary             ; set up move
22067                                       ;xor    si,si
22068                                       ;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
                                        
000004B6 CB                             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
```

```
                        add     ax,cx
                        mov     es,ax

                        xor     si,si
                        mov     di,si

                        mov     ds,[cs:CURRENT_DOS_LOCATION] ; where it is (set by msinit)

                        ;mov    ax,cs
                        ;mov    ds,ax

                        ;;;mov  cx,20480  ; MSDOS 6.21 IO.SYS - SYSINIT:04E2h
                        ;;mov   cx,dossize/2 ; MSDOS 6.0
                        ;mov    cx,[DOSSIZE] ; words (not bytes!)  ; Retro DOS v4.0 (3.0, 2.0)
                        ;mov    es,[FINAL_DOS_LOCATION] ; on top of SYSINIT code
                        ;mov    ds,[CURRENT_DOS_LOCATION]

                        ; 22/10/2022
                        mov     cx,DOSSIZE/2 ; 5000h
                                             ; 03/09/2023
                                             ; 5800h (PCDOS 7.1)
                        rep      movsw
                        mov     [cs:CURRENT_DOS_LOCATION],es

                    ; The DOS code is ORGed at a non-zero value to allow it to be located in
                    ; HIMEM. Thus, the DOS segment location must be adjusted accordingly.
                    ; If this is ROMDOS, however, only the init code is loaded into RAM, so
                    ; this ORG is not done. The entry point is at offset zero in the segment.

                        ; 22/04/2019 (MSDOS 6.0 & MSDOS 6.21 kernel address modification)
                        ;mov    ax,cs
                        ; 02/11/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 IO.SYS, SYSINIT)
                        ;mov    ds,ax

                    ; 29/04/2019 - Retro DOS v4.0 ! important MODIFICATION !

                    ;   ; 24/04/2019
                    ;;ifndef ROMDOS
                    ;   mov     ax,[es:3]           ; get offset of dos
                    ;           ; ax = 3DE0h for MSDOS 6.21 kernel (MSDOS.SYS, offset 3)
                    ;   mov     [dosinit],ax        ; that's the entry point offset
                    ;   call    off_to_para         ; subtract this much from segment
                    ;   ; 23/04/2019
                    ;   ;sub    [CURRENT_DOS_LOCATION],ax
                    ;   sub     [FINAL_DOS_LOCATION],ax
                    ;;else
                    ;;  mov     word [dosinit],0    ; entry to init is at zero
                    ;;
                    ;;endif ; ROMDOS

                        ; 29/04/2019 - Retro DOS v4.0 ! important MODIFICATION !
                        ; (! MSDOS6.BIN starts with DOSDATA ! - Retro DOS v4.0 modification)

                        ;mov    ax,[es:0] ; DOSCODE start address = DOSDATA size (= 136Ah)
                        ;               ; (Valid for Retro DOS v4.0 only!)

                        ; 22/10/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 IO.SYS, SYSINIT)
                        ; (SYSINIT:0563h for MSDOS 5.0 IO.SYS SYSINIT)
                        ; 03/09/2023 - Retro DOS 4.2 (Modified MSDOS 6.21 IO.SYS)
                        ; (SYSINIT:04ECh for MSDOS 6.21 IO.SYS SYSINIT)
                        ; (SYSINIT:0540h for PCDOS 7.1 IBMBIO.COM SYSINIT)
                        mov     ax,[3]          ; mov ax, word ptr ds:3
                                                ; 30/12/2023
                                                ; ax = 3F10h for IBMDOS 7.1 kernel
                                                ;  (IBMDOS.SYS, offset 3)

                        mov     [cs:dosinit],ax ; (SYSINIT:0563h for MSDOS 5.0 IO.SYS SYSINIT)
                        ; 02/11/2022
                        call    off_to_para         ; subtract this much from segment
                        sub     [cs:CURRENT_DOS_LOCATION],ax

                        ; Current DOSCODE start address = dword [dosinit]

                    ;; If this is not ROMDOS, then the BIOS code is moved to the top of memory
                    ;; until it is determined whether it will be running in HIMEM or not.

                    ;ifndef ROMDOS

                    ; now put Bios_Code up on top of that. Assume Bios_Code + dossize < 64k

                        ; 22/10/2022
                        mov     ax,es
                        add     ax,DOSSIZE/16       ; get paragraph of end of dos
                        mov     es,ax
                        xchg    ax,[cs:temp_bcode_seg]; swap with original home of Bios_Code
                        mov     ds,ax               ; point to loaded image of Bios_Code

                        ;mov    si,BCODE_START ; mov si,30h
                        ; 09/12/2022
                        mov     si,BCODESTART
                        ; 02/11/2022
                        mov     di,si
                        ; 30/12/2023
                        ;mov    cx,1E00h        ; BCODE_END = (SYSINITSEG-DOSBIOCODESEG)*16
                        ;                       ; (544h-364h)*10h = 1E00h (for PCDOS 7.1 IBMBIO.COM)
                        ;mov    cx,BCODE_END    ; mov cx,1A60h ; mov cx,1A70h ; 30/12/2022
                        ;sub    cx,si
                        ; 31/03/2024
                        BCODESIZE equ BCODEEND-BCODESTART
                        mov     cx,BCODESIZE
                        shr     cx,1
                        rep     movsw               ; move Bios_Code into place

                        mov     ax,es               ; tell it what segment it's in
                        call    far [cs:seg_reinit_ptr]     ; far call to seg_reinit in Bios_Code (M022)

                    ;endif      ; not ROMDOS

                    ; now call dosinit while it's in its temporary home

                        ;mov    ax,cs
                        ;mov    ds,ax

                        ;mov    dx,[MEMORY_SIZE]    ; set for call to dosinit

                        ; 22/10/2022

                        les     di,[cs:BiosComBlock]  ; ptr to BIOS communication block
                                ; es = KERNEL_SEGMENT (70h), di = 'SysInitPresent' address
                        lds     si,[cs:DEVICE_LIST]   ; set for call to dosinit
                                ; ds = KERNEL_SEGMENT (70h), si = 'res_dev_list' address

                        mov     dx,[cs:MEMORY_SIZE]   ; set for call to dosinit
```

```
22441 0000055C FA                       cli
22442 0000055D 8CC8                     mov     ax,cs
22443 0000055F 8ED0                     mov     ss,ax
22444
22445                                 ; 30/12/2023 - Retro DOS v5.0 (PCDOS 7.1 IBMBIO.COM)
22446                                 ; 22/03/2019 - Retro DOS v4.0 (MSDOS 6.0, SYSINIT1.ASM)
22447                                 %define locstack ($ - SYSINIT$) & 0FFFEh  ; 532h in MSDOS 6.21 IO.SYS
22448                                                                          ; 5A6h in MSDOS 5.0 IO.SYS SYSINIT
22449                                                                          ; 586h in PCDOS 7.1 IBMBIO.COM SYSINIT
22450                                 ;SYSINIT:0532h:
22451
22452                                 ; 22/10/2022
22453                                 ; ----------------------------------------------------------------------------
22454                                 ;SYSINIT:05A6h:
22455                                 ;locstack:  ; (at SYSINIT:05A6h for MSDOS 5.0 IO.SYS)
22456
22457                                 ; 03/09/2023
22458                                 ; (locstack at SYSINIT:0586h in PCDOS 7.1 IBMBIO.COM SYSINIT)
22459
22460                                     ;mov    sp,05A6h
22461 00000561 BC6005                   mov     sp,locstack           ; set stack
22462
22463 00000564 FB                       sti
22464
22465                                 ;align 2
22466                                     ; 30/03/2018
22467                                 ;LOCSTACK:
22468                                         ;CALL      FAR [CS:MSDOS] ; FINAL_DOS_LOCATION:0
22469                                                            ;('jmp DOSINIT' in 'MSHEAD.ASM')
22470                                                            ;('DOSINIT:' is in 'MSINIT.ASM')
22471
22472                                     ; 22/10/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 IO.SYS)
22473                                     ; 22/03/2019 - Retro DOS v4.0 (MSDOS 6.0, 6.21)
22474
22475                                 ; This call to DOSINIT will relocate the DOS data from its present location
22476                                 ; at the top of memory, to its final location in low memory just above the
22477                                 ; BIOS data. It will then build important DOS data structures in low
22478                                 ; memory following the DOS data. It returns (among many other things) the
22479                                 ; new starting address of free memory.
22480
22481 00000565 2EFF1E[7102]                  call    far [cs:dosinit]      ; call dosinit
22482                                                         ; es:di -> sysinitvars_ext
22483
22484 0000056A 2E8C1E[8502]               mov     [cs:def_php],ds               ; save pointer to PSP
22485
22486                                     ; 11/12/2022
22487                                     ; 22/03/2019
22488 0000056F 0E                       push    cs
22489 00000570 1F                       pop     ds
22490                                   ; 22/10/2022
22491 00000571 A3[8302]                 mov     [hi_doscod_size],ax
22492 00000574 890E[8102]               mov     [lo_doscod_size],cx
22493 00000578 8916[7D02]               mov     [dos_segreinit],dx
22494
22495                                   ; 11/12/2022
22496                                   ; ds = cs
22497                                   ;mov    [cs:hi_doscod_size],ax; size of doscode (including exepatch)
22498                                   ;mov    [cs:lo_doscod_size],cx; (not including exepatch)
22499                                   ;mov    [cs:dos_segreinit],dx ; save offset of segreinit
22500
22501                                   ; 05/06/2018 - Retro DOS v3.0
22502                                   ; ES:DI = Address of pointer to SYSINITVARS structure (MSDOS 3.3)
22503
22504                                   ; 11/12/2022
22505                                   ; ds = cs
22506                                   ; 22/10/2022
22507                                   ;mov    ax,[es:di+SysInitVars_Ext.SYSI_InitVars] ; 5/29/86
22508 0000057C 268B05                   mov     ax,[es:di] ; 22/03/2019
22509                                   ;mov    [cs:DOSINFO],ax
22510 0000057F A3[6D02]                 mov     [DOSINFO],ax
22511                                   ;mov    ax,[es:di+SysInitVars_Ext.SYSI_InitVars+2]
22512 00000582 268B4502                 mov     ax,[es:di+2]
22513                                   ;mov    [cs:DOSINFO+2],ax
22514 00000586 A3[6F02]                 mov     [DOSINFO+2],ax ; set the sysvar pointer
22515
22516                                   ;mov    ax,[es:di+SysInitVars_Ext.SYSI_Country_Tab]
22517 00000589 268B4504                 mov     ax,[es:di+4]
22518                                   ;mov    [cs:sysi_country],ax
22519 0000058D A3[7902]                 mov     [sysi_country],ax
22520                                   ;mov    ax,[es:di+SysInitVars_Ext.SYSI_Country_Tab+2]
22521 00000590 268B4506                 mov     ax,[es:di+6]
22522                                   ;mov    [cs:sysi_country+2],ax
22523 00000594 A3[7B02]                 mov     [sysi_country+2],ax   ; set the SYSI_Country pointer
22524
22525                                   ; 20/04/2019
22526                                   ;mov    ax,[CURRENT_DOS_LOCATION]
22527                                   ;;mov   es,[CURRENT_DOS_LOCATION]
22528                                   ;mov    ax,[FINAL_DOS_LOCATION] ; give dos its temporary location
22529                                   ; 22/10/2022
22530                                   ;mov    ax,[cs:CURRENT_DOS_LOCATION]
22531                                   ;;;mov  [dos_segreinit+2],es
22532                                   ;;mov   [dos_segreinit+2],ax
22533                                   ;mov    [cs:dos_segreinit+2],ax
22534                                   ; 11/12/2022
22535                                   ; ds = cs
22536 00000597 8E06[7302]                mov     es,[CURRENT_DOS_LOCATION]
22537 0000059B 8C06[7F02]               mov     [dos_segreinit+2],es
22538                                   ; 02/11/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
22539                                   ;mov    es,[cs:CURRENT_DOS_LOCATION]
22540                                   ;mov    [cs:dos_segreinit+2],es
22541
22542                                 ; ----------------------------------------------------------------------------
22543
22544                                 ;SYSINIT:0577h:
22545                                     ; ... RPLArena ... MSDOS 6.21 IO.SYS (SYSINIT:0577h to SYSINIT:05D1h)
22546                                 ;SYSINIT:05D1h:    ; NoRPLArena
22547
22548                                     ; 22/10/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 IO.SYS SYSINIT)
22549                                 ;------ Cover up RPL code with an arena
22550                                 ;SYSINIT:05EBh:
22551                                     ; 11/12/2022
22552                                     ; ds = cs
22553 0000059F 31DB                     xor     bx,bx
22554 000005A1 391E[9602]               cmp     [RPLMemTop],bx ; 0
22555                                   ;cmp    word [RPLMemTop],0
22556                                   ;;cmp   word [cs:RPLMemTop],0
22557 000005A5 7450                     je      short NoRPLArena
22558
22559                                 ;------ alloc all memory
22560
22561                                     ; 11/12/2022
22562                                     ;mov    bx,0FFFFh
22563 000005A7 4B                       dec     bx
22564                                     ; 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
```

```
22813                                   ;   mov     [CONFBOT],ax          ; init this in case no CONFIG.SYS
22814
22815                                       ; 30/12/2022 - Retro DOS v4.2 (MSDOS 6.21 IO.SYS)
22816                                       ; (SYSINIT:064Ch)
22817 00000643 89C1                       mov     cx,ax ; (*)
22818 00000645 2D8F00                     sub     ax,((26*(curdirlen))+15)/16 ; sub ax,143
22819 00000648 A3[A502]                   mov     [ALLOCLIM],ax         ; init top of free memory pointer
22820 0000064B A3[A302]                   mov     [CONFBOT],ax          ; init this in case no CONFIG.SYS
22821
22822                                   ; setup and initialize the temporary buffer at cx
22823
22824                                       ;les    di,[es:di+SYSI_BUF]   ; get the buffer chain entry pointer
22825 0000064E 26C47D12                   les     di,[es:di+12h]
22826                                       ; 11/12/2022
22827 00000652 31DB                       xor     bx,bx
22828                                       ;xor    ax,ax
22829                                       ;mov    [es:di+BUFFINF.Dirty_Buff_Count],ax ; 0
22830                                       ;mov    word [es:di+4],0
22831 00000654 26895D04                   mov     [es:di+4],bx ; 0
22832                                       ;mov    [es:di+BUFFINF.Buff_Queue],ax ; 0
22833                                       ;mov    word [es:di],0
22834 00000658 26891D                     mov     [es:di],bx ; 0
22835                                       ;mov    [es:di+BUFFINF.Buff_Queue+2],cx ; cx = [top_of_cdss] ; 6.21
22836                                       ;;mov   [es:di+BUFFINF.Buff_Queue+2],ax ; ax = [top_of_cdss] ; 5.0
22837                                       ;mov    [es:di+2],ax
22838                                       ;mov    es,ax  ; [top_of_cdss] = [CONFBOT]
22839                                       ; 30/12/2022 - Retro DOS v4.2 (MSDOS 6.21 IO.SYS, SYSINIT)
22840 0000065B 26894D02                   mov     [es:di+2],cx ; [top_of_cdss] ; (*)
22841 0000065F 8EC1                       mov     es,cx
22842
22843                                       ; 11/12/2022
22844                                       ;xor    ax,ax
22845                                       ;mov    di,ax                 ; es:di -> single buffer
22846 00000661 89DF                       mov     di,bx
22847                                       ; di = 0
22848
22849                                       ;mov    [es:di+buffinfo.buf_next],ax ; points to itself
22850                                       ; 11/12/2022
22851                                       ;mov    [es:di],ax ; 0
22852 00000663 26891D                     mov     [es:di],bx ; 0
22853                                       ;mov    [es:di+buffinfo.buf_prev],ax ; points to itself
22854                                       ; 11/12/2022
22855                                       ;mov    [es:di+2],ax ; 0
22856 00000666 26895D02                   mov     [es:di+2],bx ; 0
22857
22858                                       ; 22/10/2022 - Retro DOS 4.0 (Modified MSDOS 5.0 IO.SYS SYINIT)
22859                                       ; MSDOS 5.0 IO.SYS - SYSINIT:06E0h
22860
22861                                       ;mov    word [es:di+buffinfo.buf_ID],00FFh ; free buffer,clear flag
22862 0000066A 26C74504FF00               mov     word [es:di+4],00FFh
22863                                   ;SYSINIT:06E6h
22864                                       ;;mov   [es:di+buffinfo.buf_sector],ax ; 0
22865                                       ;mov    word [es:di+6],0
22866                                       ; 11/12/2022
22867                                       ;mov    [es:di+buffinfo.buf_sector],bx ; 0
22868 00000670 26895D06                   mov     [es:di+6],bx ; 0
22869                                       ;;mov   [es:di+buffinfo.buf_sector+2],ax ; 0
22870                                       ;mov    word [es:di+8],0
22871                                       ; 11/12/2022
22872                                       ;mov    [es:di+buffinfo.buf_sector+2],bx ; 0
22873 00000674 26895D08                   mov     [es:di+8],bx ; 0
22874
22875                                       ; 30/12/2023 (!*)
22876                                       ;pop    di                    ; restore pointer to DOSINFO data
22877                                       ;pop    es
22878
22879                                       ; 11/12/2022
22880                                       ; ds = cs
22881                                       ; 22/10/2022
22882                                       ;push   cs
22883                                       ;pop    ds
22884
22885 00000678 E82807                     call    TempCDS               ; set up cdss so re_init and sysinit
22886                                                                     ;  can make disk system calls
22887                                                                     ; tempcds trashes ds
22888                                       ; 10/05/2019
22889 0000067B 2E8E1E[8502]               mov     ds,[cs:def_php]               ; retrieve pointer to PSP returned by DOSINIT
22890
22891                                       ;if not ibmjapver
22892                                       ;call   far KERNEL_SEGMENT:re_init ; re-call the bios
22893                                       ;endif
22894
22895                                       ; 22/10/2022
22896                                   ;SYSINIT:06FEh:    ; (MSDOS 5.0 IO.SYS, SYSINIT)
22897                                       ; 30/12/2022
22898                                   ;SYSINIT:0697h:    ; (MSDOS 6.21 IO.SYS, SYSINIT)
22899                                       ;call   far ptr 70h:89Bh
22900 00000680 9A[2807]7000               call    DOSBIODATASEG:RE_INIT
22901
22902 00000685 FB                          sti                           ; ints ok
22903 00000686 FC                         cld                           ; make sure
22904
22905                                   ; 23/03/2019
22906
22907                                   ;SYSINIT:069Eh     ; 30/12/2022
22908
22909                                   ; dosinit has set up a default "process" (php) at ds:0. we will move it out
22910                                   ; of the way by putting it just below sysinit at end of memory.
22911
22912 00000687 8CCB                       mov     bx,cs
22913 00000689 83EB10                     sub     bx,10h
22914 0000068C 8EC3                       mov     es,bx
22915 0000068E 31F6                       xor     si,si
22916 00000690 89F7                       mov     di,si
22917 00000692 B98000                     mov     cx,128
22918 00000695 F3A5                       rep     movsw
22919
22920                                       ;mov    [es:PDB.JFN_POINTER+2],es ; Relocate
22921                                       ; 22/10/2022
22922 00000697 268C063600                 mov     [es:36h],es
22923
22924                                       ; Set Process Data Block - Program Segment Prefix address
22925                                       ; BX = PDB/PSP segment
22926 0000069C B450                           mov ah,50h  ; SET_CURRENT_PDB
22927 0000069E CD21                       int     21h                   ; tell DOS we moved it
22928                                                   ; DOS - 2+ internal - SET PSP SEGMENT
22929                                                   ; BX = segment address of new PSP
22930                                       ; 22/10/2022
22931                                       ; 27/03/2019
22932                                       ; 30/12/2023
22933                                       ;push   ds ; */               ; preserve DS returned by DOSINIT
22934
22935 000006A0 0E                         push    cs
22936 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
```

```
23061 00000715 8936[A716]               mov     [MagicDDNamePtr],si
23062 00000719 1E                       push    ds
23063 0000071A 29C0                     sub     ax,ax
23064 0000071C 8ED8                     mov     ds,ax   ; 0             ; ROMBIOS data area
23065 0000071E 8B166C04                 mov     dx,[46Ch]               ; Counter for Interrupt 1Ah
23066                                   wfk2s_1:
23067 00000722 B401                     mov     ah,1
23068 00000724 CD16                     int     16h             ; KEYBOARD - CHECK BUFFER, DO NOT CLEAR
23069                                                           ; Return: ZF clear if character in buffer
23070                                                           ; AH = scan code, AL = character
23071                                                           ; ZF set if no character in buffer
23072 00000726 7511                     jnz     short wfk2s_2
23073 00000728 B402                     mov     ah,2
23074 0000072A CD16                     int     16h                     ; KEYBOARD - GET SHIFT STATUS
23075                                                                   ; AL = shift status bits
23076 0000072C A803                     test    al,3
23077 0000072E 7509                     jnz     short wfk2s_2
23078 00000730 A16C04                   mov     ax,[46Ch]               ; tick count
23079 00000733 29D0                     sub     ax,dx
23080 00000735 3C25                         cmp al,37                   ; 2 seconds
23081 00000737 72E9                     jb      short wfk2s_1           ; wait for user's key press
23082                                   wfk2s_2:
23083 00000739 1F                       pop     ds                      ; read/check the pressed key
23084 0000073A 29DB                     sub     bx,bx   ; bx = 0
23085 0000073C B402                     mov     ah,2
23086 0000073E CD16                     int     16h                     ; KEYBOARD - GET SHIFT STATUS
23087                                                                   ; AL = shift status bits
23088 00000740 A803                     test    al,3                    ; Left or Right SHIFT key pressed ?
23089 00000742 7402                     jz      short wfk2s_3           ; no
23090 00000744 43                       inc     bx
23091 00000745 43                       inc     bx      ; bx = 2
23092                                   wfk2s_3:
23093 00000746 B401                     mov     ah,1
23094 00000748 CD16                     int     16h             ; KEYBOARD - CHECK BUFFER, DO NOT CLEAR
23095                                                           ; Return: ZF clear if character in buffer
23096                                                           ; AH = scan code, AL = character
23097                                                           ; ZF set if no character in buffer
23098 0000074A 7418                     jz      short wfk2s_6
23099 0000074C 80FC65                   cmp     ah,65h                  ; F8 key pressed ?
23100 0000074F 740C                     jz      short wfk2s_5
23101 00000751 80FC62                   cmp     ah,62h                  ; F5 key pressed ?
23102 00000754 750E                     jnz     short wfk2s_6
23103                                   ;mov    byte [cs:F5_key],1
23104 00000756 C606[8E03]01             mov     byte [F5_key],1
23105                                   wfk2s_4:
23106 0000075B EB49                     jmp     short ProcessConfig   ; continue (as normal/default state)
23107
23108                                   wfk2s_5:
23109                                   ;mov    byte [cs:F8_key],1
23110 0000075D C606[8F03]01             mov     byte [F8_key],1
23111 00000762 EB42                     jmp     short ProcessConfig
23112
23113                                   wfk2s_6:
23114 00000764 E8AA02                   call    AllocFreeMem            ; get the largest free block from DOS
23115 00000767 E8700F                   call    MagicPreload            ; **** PRE-LOAD MAGICDRV!!! ****
23116
23117                                   ; 07/04/2024 - Retro DOS v5.0
23118                                   ; (DS may not be same with CS here!)
23119 0000076A 0E                       push    cs
23120 0000076B 1F                       pop     ds ; *
23121 0000076C 8E06[6803]               mov     es,[area]
23122
23123 00000770 09C0                     or      ax,ax                   ; error?
23124 00000772 7406                     jz      short wfk2s_7
23125                                   PreloadFailed:
23126 00000774 B449                     mov     ah,49h                  ; Dealloc ; free the block if no load
23127                                   ;;mov   es,[cs:area]
23128                                   ;mov    es,[area]
23129 00000776 CD21                     int     21h             ; DOS - 2+ - FREE MEMORY
23130                                                           ; ES = segment address of area to be freed
23131 00000778 EB2C                     jmp     short ProcessConfig
23132
23133                                   wfk2s_7:
23134                                   ;mov    bx,[cs:memhi]
23135                                   ;mov    es,[cs:area]
23136                                   ;sub    bx,[cs:area]            ; get desired block size in paras
23137                                   ; 07/04/2024 - Retro DOS v5.0
23138                                   ; ds = cs ; *
23139 0000077A 8CC3                     mov     bx,es
23140 0000077C F7DB                     neg     bx              ; bx = - [cs:area]
23141 0000077E 031E[6403]               add     bx,[memhi]      ; bx = [cs:memhi] - [cs:area]
23142
23143 00000782 B44A                     mov     ah,4Ah
23144 00000784 CD21                     int     21h             ; DOS - 2+ - ADJUST MEMORY BLOCK SIZE (SETBLOCK)
23145                                                           ; ES = segment address of block to change
23146                                                           ; BX = new size in paragraphs
23147 00000786 8CC0                     mov     ax,es
23148 00000788 48                       dec     ax
23149 00000789 8EC0                     mov     es,ax                   ; get Magicdrv arena
23150
23151 0000078B 26C70601000800           mov     word [es:1],8  ; [es:arena_owner]
23152                                   ;mov    word [es:ARENA.OWNER],8       ; set impossible owner
23153 00000792 26C70608005344           mov     word [es:8],4453h ; [es:arena_name],'SD' ; System Data
23154                                   ;mov    word [es:ARENA.NAME],'SD' ; 4453h
23155 00000799 2603060300                   add ax,[es:3]               ; get MCB length
23156                                   ;add    ax,[es:ARENA.SIZE]
23157
23158                                   ;lds    si,[cs:DOSINFO]                 ; get to arena header
23159 0000079E C536[6D02]               lds     si,[DOSINFO]
23160 000007A2 40                           inc     ax                  ; get addr of next MCB
23161 000007A3 8944FE                       mov     [si-2], ax          ; store that
23162
23163                                   ; ----------------------------------------------------------------------------
23164
23165                                   ; MSDOS 6.21 IO.SYS, SYSINIT:0744h
23166
23167                                   ; 23/03/2019 - Retro DOS v4.0 (MSDOS 6.0, SYSINIT1.ASM, 1991)
23168                                   ; ----------------------------------------------------------------------------
23169                                   ; 22/10/2022 - Retro DOS v4.0 (MSDOS 5.0 IO.SYS SYSINIT)
23170                                   ; ----------------------------------------------------------------------------
23171                                   ; 30/12/2022 - Retro DOS v4.2 (MSDOS 6.21 IO.SYS SYSINIT)
23172                                   ; ----------------------------------------------------------------------------
23173                                   ; 07/04/2024 - Retro DOS v5.0 (PCDOS 7.1 IBMBIO.COM SYSINIT)
23174
23175                                   ; (MSDOS 6.22 IO.SYS - SYSINIT:0744h)
23176
23177                                   ; PCDOS 7.1 IBMBIO.COM - SYSINIT:0820h
23178
23179                                   ProcessConfig:
23180                                       ;; ds = cs ; 27/03/2019
23181                                       ; 11/12/2022
23182                                       ; ds <> cs
23183
23184                                   ; (MSDOS 5.0 IO.SYS - SYSINIT:0746h)
```

```
23185
23186 000007A6 E8BF1C                    call    doconf                  ; do pre-scan for dos=high/low
23187
23188                                    ; 11/12/2022
23189                                    ; 27/03/2019
23190                                    ; ds = cs (at return from doconf)
23191
23192                                ; Now, if this is not romdos, we decide what to do with the DOS code.
23193                                ; It will either be relocated to low memory, above the DOS data structures,
23194                                ; or else it will be located in HiMem, in which case a stub with the DOS
23195                                ; code entry points will be located in low memory. Dos_segreinit is used
23196                                ; to tell the DOS data where the code has been placed, and to install the
23197                                ; low memory stub if necessary. If the DOS is going to go into HiMem, we
23198                                ; must first initialize it in its present location and load the installable
23199                                ; device drivers. Then, if a HiMem driver has been located, we can actually
23200                                ; relocate the DOS code into HiMem.
23201                                ;
23202                                ; For ROMDOS, if DOS=HIGH is indicated, then we need to call dos_segreinit
23203                                ; to install the low memory stub (this must be done before allowing any
23204                                ; device drivers to hook interrupt vectors). Otherwise, we don't need to
23205                                ; call dos_segreinit at all, since the interrupt vector table has already
23206                                ; been patched.
23207
23208                                    ; 22/10/2022 - Retro DOS v4.0
23209                                    ; (MSDOS 5.0 IO.SYS - SYSINIT:0749h)
23210                                    ;cmp    byte [cs:runhigh],0   ; Did user choose to run low ?
23211                                    ; 11/12/2022
23212 000007A9 803E[6C02]00            cmp     byte [runhigh],0
23213 000007AE 7404                    je      short dont_install_stub      ; yes, don't install dos low mem stub
23214
23215                                ;------ user chose to load high
23216
23217                                    ; 22/10/2022
23218                                    ;mov    es,[cs:CURRENT_DOS_LOCATION] ; MSDOS 6.21 (& MSDOS 6.0)
23219                                    ; 11/12/2022
23220                                    ; ds = cs
23221                                ; 13/04/2024
23222                                %if 0
23223                                    mov     es,[CURRENT_DOS_LOCATION]
23224                                %endif
23225                                    ;mov    es,[cs:FINAL_DOS_LOCATION] ; Retro DOS v4.0
23226                                    ; 27/03/2019
23227                                    ;;mov   es,[FINAL_DOS_LOCATION]
23228
23229 000007B0 31C0                    xor     ax,ax                   ; ax = 0 ---> install stub
23230
23231                                ; 13/04/2024
23232                                %if 0
23233                                    ; 11/12/2022
23234                                    ; ds = cs
23235                                    ;call   far [cs:dos_segreinit]; call dos segreinit
23236                                    call    far [dos_segreinit]
23237                                %endif
23238 000007B2 EB08                    jmp     short do_multi_pass
23239
23240                                ;------ User chose to load dos low
23241
23242                                dont_install_stub:
23243                                    ; 22/10/2022
23244 000007B4 31DB                    xor     bx,bx                   ; M012
23245                                                                    ; don't use int 21 call to alloc mem
23246 000007B6 E80E03                  call    MovDOSLo                ; move it !
23247
23248 000007B9 B80100                  mov     ax,1                    ; dont install stub
23249
23250                                ; 13/04/2024
23251                                %if 1
23252                                do_multi_pass:
23253                                %endif
23254                                    ; 11/12/2022
23255                                    ; ds = cs
23256 000007BC 8E06[7302]              mov     es,[CURRENT_DOS_LOCATION]
23257                                    ;mov    es,[cs:CURRENT_DOS_LOCATION] ; set_dos_final_position set it up
23258                                    ;;mov   es,[cs:FINAL_DOS_LOCATION]   ; Retro DOS v4.0
23259                                    ; 27/03/2019
23260                                ;do_multi_pass:
23261                                    ;;mov   es,[FINAL_DOS_LOCATION]
23262
23263                                    ; 11/12/2022
23264                                    ; ds =cs
23265                                    ;call   far [cs:dos_segreinit]; inform dos about new seg
23266 000007C0 FF1E[7D02]              call    far [dos_segreinit]
23267
23268                                ; 13/04/2024
23269                                %if 0
23270                                do_multi_pass:
23271                                %endif
23272
23273 000007C4 E84A02                  call    AllocFreeMem            ; allocate all the free mem
23274                                                                    ; & update [memhi] & [area]
23275                                                                    ; start of free memory.
23276                                    ;ifdef dblspace_hooks
23277                                    ;mov    bx,0                    ; magic backdoor to place int hooks
23278                                    ;call   cs:MagicBackdoor
23279                                    ;endif
23280
23281                                    ; 07/04/2024 - Retro DOS v5.0
23282                                    ; (PCDOS 7.1 IBMBIO.COM)
23283                                    ;cmp    byte [cs:F5_key],1
23284 000007C7 803E[8E03]01            cmp     byte [F5_key],1
23285 000007CC 740D                    je      short skip_magicbackdoor
23286                                    ;cmp    byte [cs:F8_key],1
23287 000007CE 803E[8F03]01            cmp     byte [F8_key],1
23288 000007D3 7406                    je      short skip_magicbackdoor
23289 000007D5 31DB                    xor     bx,bx   ; bx = 0        ; magic backdoor to place int hooks
23290                                    ;call   far [cs:MagicBackdoor]
23291 000007D7 FF1E[9003]              call    far [MagicBackdoor]
23292
23293                                skip_magicbackdoor:
23294
23295                                ; Now, process config.sys some more.
23296                                ; Load the device drivers and install programs
23297
23298                                    ; 22/10/2022
23299                                    ;inc    byte [cs:multi_pass_id]      ; multi_pass_id = 1
23300                                    ; 11/12/2022
23301                                    ; ds = cs
23302 000007DB FE06[CD02]              inc     byte [multi_pass_id]
23303 000007DF E8221D                  call    multi_pass              ; load device drivers
23304 000007E2 E8EA31                  call    ShrinkUMB
23305 000007E5 E80E32                  call    UnlinkUMB               ; unlink all UMBs     ;M002
23306                                    ; 02/11/2022
23307                                    ;inc    byte [cs:multi_pass_id]      ; multi_pass_id = 2
23308                                    ; 11/12/2022
```

```
                                    ; ds = cs
000007E8 FE06[CD02]                 inc     byte [multi_pass_id]
000007EC E8151D                     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?
```

```
23433                                   ;test   byte [cs:install_flag],has_installed
23434                                   ; 11/12/2022
23435                                   ; ds = cs
23436 00000845 F606[CE02]02            test    byte [install_flag],2 ; has_installed
23437                                   ;test   byte [install_flag],has_installed
23438 0000084A 741F                     jz      short skip_free_sysinitbase ; no.
23439
23440                                 ; set block from the old_area with impossible_owner_size.
23441                                 ; this will free the unnecessary sysinit_base that had been put in memory to
23442                                 ; handle install= command.
23443
23444                                   ; 12/12/2022
23445                                      ;push      es              ; BUGBUG 3-30-92 JeffPar: no reason to save ES
23446                                   ;push   bx
23447
23448                                   ; 22/10/2022
23449                                   ;mov    es,[cs:old_area]
23450                                   ;mov    bx,[cs:impossible_owner_size]
23451                                   ; 12/12/2022
23452                                   ; ds = cs
23453 0000084C 8E06[5E03]              mov     es,[old_area]
23454 00000850 8B1E[6003]              mov     bx,[impossible_owner_size]
23455
23456 00000854 B44A                    mov     ah,4Ah ; SETBLOCK
23457 00000856 CD21                    int     21h
23458                                                  ; DOS - 2+ - ADJUST MEMORY BLOCK SIZE (SETBLOCK)
23459                                                  ; ES = segment address of block to change
23460                                                  ; BX = new size in paragraphs
23461 00000858 8CC0                    mov     ax,es
23462 0000085A 48                      dec     ax
23463 0000085B 8EC0                    mov     es,ax                   ; point to arena
23464                                   ;mov    word [es:ARENA.OWNER],8       ; set impossible owner
23465 0000085D 26C70601000800          mov     word [es:1],8
23466                                   ;mov    word [es:ARENA.NAME],'SD' ; 4453h ; System Data
23467 00000864 26C70608005344          mov     word [es:8],'SD'
23468
23469                                   ; 12/12/2022
23470                                   ;pop    bx
23471                                      ;pop       es              ; BUGBUG 3-30-92 JeffPar: no reason to save ES
23472
23473                                 skip_free_sysinitbase:
23474                                   ; 22/10/2022
23475                                   ;cmp    byte [cs:runhigh],0
23476                                   ; 12/12/2022
23477                                   ; ds = cs
23478 0000086B 803E[6C02]00            cmp     byte [runhigh],0
23479 00000870 7403                    je      short _@@@_ ; 04/07/2023
23480
23481 00000872 E8DF03                  call    InstVDiskHeader         ; Install VDISK header (allocates some mem from DOS)
23482
23483                                 ; ----------------------------------------------------------------------------
23484
23485                                 _@@@_:
23486                                   ; 12/12/2022
23487                                   ; ds = cs
23488                                   ; 22/10/2022
23489                                   ; 27/03/2019
23490                                   ;push   cs
23491                                   ;pop    ds                      ; point DS to sysinitseg
23492
23493                                 ; set up the parameters for command
23494
23495                                 ;   ; 22/10/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 IO.SYS SYSINIT)
23496                                 ;;ifdef     MULTI_CONFIG
23497                                 ;   mov     byte [config_cmd],0   ; set special code for query_user
23498                                 ;       call    query_user        ; to issue the AUTOEXEC prompt
23499                                 ;   jnc     short process_autoexec; we should process autoexec normally
23500                                 ;   ; !!!
23501                                 ;   or      byte [bQueryOpt],4 ; MSDOS 6.21 IO.SYS - SYSINIT:081Fh
23502                                 ;       ; !!!
23503                                 ;   call    disable_autoexec        ; no, we should disable it
23504                                 ;process_autoexec:
23505                                 ;;endif     ; !!!
23506                                 ;   call    CheckQueryOpt  ; MSDOS 6.21 IO.SYS - SYSINIT:0827h
23507                                 ;   ; !!!
23508
23509                                   ; 22/10/2022
23510                                   ;mov     cl,[command_line]
23511                                      ;mov     ch,0
23512                                      ;inc     cx
23513                                      ;mov     si,command_line
23514                                   ;add     si,cx
23515                                      ;mov     byte [si],cr     ; cr-terminate command line
23516
23517                                   ; 22/10/2022 - Retro DOS 4.0 (Modified MSDOS 5.0 IO.SYS)
23518                                   ; (SYSINIT:0809h)
23519
23520                                   ;;;;
23521
23522                                   ; 30/12/2022 - Retro DOS 4.2 (Modified MSDOS 6.21 IO.SYS)
23523                                   ; (SYSINIT:0813h)
23524                                   ; ds = cs
23525                                   ; push  cs
23526                                   ; pop   ds
23527
23528 00000875 C606[6419]00            mov     byte [config_cmd],0   ; set special code for query_user
23529 0000087A E89F3D                  call    query_user            ; to issue the AUTOEXEC prompt
23530                                   ; 07/04/2024
23531                                   ;jnc    short process_autoexec; we should process autoexec normally
23532
23533                                   ; 07/04/2024 - Retro DOS v5.0 (PCDOS 7.1 IBMBIO.COM)
23534                                   ;;;
23535 0000087D 9C                      pushf
23536 0000087E F606[814C]01            test    byte [bDisableUI],1
23537 00000883 7507                    jnz     short _@@@@_   ; F5 clean/interactive boot option (has been) disabled
23538 00000885 803E[8E03]01            cmp     byte [F5_key],1
23539 0000088A 7405                    je      short _@@@@@_  ; F5 key pressed, bypass AUTOEXEC.BAT (clean boot)
23540                                 _@@@@_:
23541 0000088C 9D                      popf
23542 0000088D 730B                    jnc     short process_autoexec; we should process autoexec normally
23543 0000088F EB01                    jmp     short bypass_autoexec
23544                                 _@@@@@_:
23545 00000891 9D                      popf                    ; cf status at the return from 'query_user' call
23546                                 bypass_autoexec:
23547                                   ;;;
23548
23549                                   ; !!!
23550 00000892 800E[854C]04            or      byte [bQueryOpt],4 ; MSDOS 6.21 IO.SYS - SYSINIT:081Fh
23551                                   ; !!!
23552 00000897 E87D3E                  call    disable_autoexec        ; no, we should disable it
23553                                 process_autoexec:
23554                                   ; !!!
23555 0000089A E8C53E                  call    CheckQueryOpt  ; MSDOS 6.21 IO.SYS - SYSINIT:0827h
23556
```

```
23557                                   ;mov     cl,[command_line]
23558                                   ; 30/12/2022
23559 0000089D BE[BB4B]                 mov     si,command_line
23560 000008A0 8A0C                     mov     cl,[si]
23561 000008A2 B500                     mov     ch,0
23562 000008A4 41                       inc     cx
23563                                   ;mov    si,command_line
23564 000008A5 01CE                     add     si,cx
23565 000008A7 C6040D                   mov     byte [si],cr ; 0Dh    ; cr-terminate command line
23566
23567                                   ;;;;
23568
23569                                 ; 30/12/2022 - Retro DOS v4.2
23570                                 %if 0
23571                                     ;mov    si,(offset command_line+1)
23572                                     mov     si,command_line+1
23573                                     push    ds
23574                                     pop     es
23575                                     mov     di,si
23576                                     mov     cl,0FFh ; -1
23577                                 _@_loop:
23578                                     inc     cl ; +1
23579                                     lodsb
23580                                     stosb
23581                                     or      al,al
23582                                     jnz     short _@_loop
23583                                     dec     di
23584                                     mov     al,0Dh
23585                                     stosb                   ; cr-terminate command line
23586                                     mov     [command_line],cl ; command line length (except CR)
23587                                 %endif
23588
23589                                 ; ----------------------------------------------------------------------------
23590
23591                                 ;   Once we get to this point, the above code, which is below "retry"
23592                                 ;   in memory, can be trashed (and in fact is -- see references to retry
23593                                 ;   which follow....)
23594
23595                                 retry:              ; PCDOS 7.1 IBMBIO.COM - SYSINIT:094Ch ; 07/04/2024
23596 000008AA BA[2D4B]                 mov     dx,commnd       ; now pointing to file description
23597
23598                                 ; we are going to open the command interpreter and size it as is done in
23599                                 ; ldfil. the reason we must do this is that sysinit is in free memory. if
23600                                 ; there is not enough room for the command interpreter,exec will probably
23601                                 ; overlay our stack and code so when it returns with an error sysinit won't be
23602                                 ; here to catch it. this code is not perfect (for instance .exe command
23603                                 ; interpreters are possible) because it does its sizing based on the
23604                                 ; assumption that the file being loaded is a .com file. it is close enough to
23605                                 ; correctness to be usable.
23606
23607                                 ; first, find out where the command interpreter is going to go.
23608
23609 000008AD 52                        push    dx              ; save pointer to name
23610 000008AE BBFFFF                   mov     bx,0FFFFh
23611 000008B1 B448                     mov     ah,48h  ; ALLOC
23612 000008B3 CD21                         int     21h             ; get biggest piece
23613 000008B5 B448                     mov     ah,48h  ; ALLOC
23614 000008B7 CD21                     int     21h             ; second time gets it
23615 000008B9 726B                     jc      short memerrjx ; oooops
23616
23617 000008BB 8EC0                     mov     es,ax
23618 000008BD B449                     mov     ah,49h  ; DEALLOC
23619 000008BF CD21                     int     21h             ; give it right back
23620 000008C1 89DD                     mov     bp,bx
23621
23622                                 ; es:0 points to block,and bp is the size of the block in para.
23623
23624                                 ; we will now adjust the size in bp down by the size of sysinit.
23625                                 ; we need to do this because exec might get upset if some of the exec
23626                                 ; data in sysinit is overlayed during the exec.
23627
23628                                     ; 22/10/2022
23629                                     ; (MSDOS 5.0 IO.SYS SYSINIT:083Bh)
23630 000008C3 8B1E[9402]                   mov     bx,[MEMORY_SIZE] ; get location of end of memory
23631 000008C7 8CC8                     mov     ax,cs           ; get location of beginning of sysinit
23632
23633                                 ; Note that the "config_wrkseg" environment data is a segment in
23634                                 ; unallocated memory (as of the Dealloc of [area], above). This is ideal
23635                                 ; in one sense, because Exec is going to make a copy of it for COMMAND.COM
23636                                 ; anyway, and no one has responsibility for freeing "config_wrkseg". But
23637                                 ; we need to make sure that there's no way Exec will stomp on that data
23638                                 ; before it can copy it, and one way to do that is to make the available
23639                                 ; memory calculation even more "paranoid", by subtracting "config_wrkseg"
23640                                 ; from the "memory_size" segment value (which is typically A000h) instead
23641                                 ; of the current sysinit CS....
23642                                 ;
23643                                 ; The reason I use the term "paranoid" is because this code should have
23644                                 ; slid the data required by Exec up to the very top of memory, because as
23645                                 ; it stands, you have to have sizeof(COMMAND.COM) PLUS 64K to load just
23646                                 ; COMMAND.COM (64k is about what sysinit, and all the goop above sysinit,
23647                                 ; consumes). Now it's just a little worse (65K or more, depending on
23648                                 ; the size of your CONFIG.SYS, since the size of the environment workspace
23649                                 ; is determined by the size of CONFIG.SYS.... -JTP
23650
23651                                     ; 30/12/2022 - Retro DOS v4.2 (Modified MSDOS 6.21, IO.SYS)
23652                                     ; (SYSINIT:0858h)
23653 000008C9 8B0E[6019]                 mov     cx,[config_envlen]
23654 000008CD E303                         jcxz    no_env          ; use config_wrkseg only if there's env data
23655 000008CF A1[6219]                     mov ax,[config_wrkseg]
23656
23657                                     ; 22/10/2022
23658                                     ;mov    cx,[config_envlen]
23659                                         ;jcxz   no_env          ; use config_wrkseg only if there's env data
23660                                         ;mov    ax,[config_wrkseg]
23661                                 ;no_env:
23662                                     ; 22/10/2022
23663                                     ; (MSDOS 5.0 IO.SYS SYSINIT:0841h)
23664                                 no_env:
23665                                     ; 30/12/2022
23666                                     ; (MSDOS 6.21 IO.SYS SYSINIT:0861h)
23667 000008D2 29C3                     sub     bx,ax           ; bx is size of sysinit in para
23668 000008D4 83C311                   add     bx,11h          ; add the sysinit php
23669 000008D7 29DD                     sub     bp,bx           ; sub sysinit size from amount of free memory
23670 000008D9 724B                     jc      short memerrjx ; if there isn't even this much memory, give up
23671
23672                                         ;mov    ax,(OPEN<<8)   ; open the file being execed
23673 000008DB B8003D                       mov ax,3D00h
23674 000008DE F9                       stc                     ; in case of int 24
23675 000008DF CD21                     int     21h
23676 000008E1 7271                     jc      short comerr   ; ooops
23677                                                   ; DOS - 2+ - OPEN DISK FILE WITH HANDLE
23678                                                   ; DS:DX -> ASCIZ filename
23679                                                   ; AL = access mode
23680                                                   ; 0 - read
```

```
                                    ; 22/10/2022
                                    ; (MSDOS 5.0 IO.SYS SYSINIT:0852h)
000008E3 89C3                           mov     bx,ax           ; handle in bx

                                ;   If the standard command interpreter is being used, verify it is correct

                                    ; 30/12/2022 - Retro DOS v4.2
                                    ; (MSDOS 6.21 IO.SYS, SYSINIT:0874h)
000008E5 803E[2A4B]00               cmp     byte [newcmd],0       ; was a new shell selected?
000008EA 7518                       jne     short skip_validation ; yes
                                    ; 07/04/2024 - Retro DOS v5.0
                                    ; (PCDOS 7.1 IBMBIO.COM, SYSINIT:098Eh)
000008EC BA[A608]                   mov     dx,retry-4      ; PCDOS 7.1 IBMBIO.COM - SYSINIT:0948h
000008EF B90400                     mov     cx,4            ;
000008F2 B43F                       mov     ah,READ         ;
000008F4 CD21                       int     21h             ;
000008F6 803E[A608]E9               cmp     byte [retry-4],0E9h
000008FB 7557                       jne     short comerr
                                    ; 20/04/2019 - Retro DOS v4.0
                                    ; 30/12/2022
                                    ;cmp    byte [retry-1],64h ; MSDOS 6.21 IO.SYS - SYSINIT:088Ch
                                                               ; .. COMMAND.COM Version 6.20 (14h&0Fh)
                                    ; 07/04/2024 - Retro DOS v5.0
                                    ;;cmp   byte [retry-1],66h ; .. COMMAND.COM Version 6.22 (16h&0Fh)
                                    ;cmp    byte [retry-1],7Ah ; PCDOS 7.1 IBMBIO.COM - SYSINIT:099Fh
                                                               ; .. COMMAND.COM Version 7.10 (0Ah&0Fh)
000008FD 803E[A908]7A               cmp     byte [retry-1],((MAJOR_VERSION&0Fh)<<4)|(MINOR_VERSION&0Fh)
00000902 7550                       jne     short comerr    ;

                                    ; 22/10/2022
                                    ;cmp    byte [newcmd],0       ; was a new shell selected?
                                    ;jne    short skip_validation ; yes
                                    ;mov    dx,retry-4
                                    ;mov    cx,4            ;
                                    ;mov    ah,READ         ;
                                    ;int    21h             ;
                                    ;cmp    byte [retry-4],0E9h
                                    ;jne    short comerr
                                    ;; 20/04/2019 - Retro DOS v4.0
                                    ;cmp    byte [retry-1],64h ; MSDOS 6.21 IO.SYS - SYSINIT:088Ch
                                    ;;cmp   byte [retry-1],((MAJOR_VERSION&0Fh)<<4)|(MINOR_VERSION&0Fh)
                                    ;jne    short comerr    ;

                                ;skip_validation:
                                    ; 22/10/2022
                                    ; (MSDOS 5.0 IO.SYS SYSINIT:0854h)
                                skip_validation:
                                    ; 30/12/2022
                                    ; (MSDOS 6.21 IO.SYS SYSINIT:0893h)
00000904 31C9                       xor     cx,cx
00000906 31D2                       xor     dx,dx
                                    ;mov    ax,(LSEEK<<8)|2
00000908 B80242                     mov     ax,4202h
0000090B F9                         stc                     ; in case of int 24
0000090C CD21                       int     21h             ; get file size in dx:ax
0000090E 7244                       jc      short comerr
                                                            ; convert size in dx:ax to para in ax
00000910 83C00F                     add     ax,15           ; round up size for conversion to para
00000913 83D200                     adc     dx,0
00000916 E88104                     call    off_to_para
00000919 B10C                       mov     cl,12
0000091B D3E2                       shl     dx,cl           ; low nibble of dx to high nibble
0000091D 09D0                       or      ax,dx           ; ax is now # of para for file
0000091F 83C010                     add     ax,10h          ; 100h byte php
00000922 39E8                       cmp     ax,bp           ; will command fit in available mem?
00000924 7208                       jb      short okld      ; jump if yes.

                                ; 30/12/2022
                                %if 0
                                    ; 22/10/2022
                                memerrjx:   ; (MSDOS 5.0 IO.SYS SYSINIT:0876h)
                                    ;jmp    memerr  ; (MSDOS 5.0 IO.SYS SYSINIT:34D5h)
                                    ; 02/11/2022
                                    ;jmp    mem_err
                                    ; 11/12/2022
                                    ; ds = cs
                                    jmp     mem_err2
                                %endif
                                    ; 30/12/2022
                                    ; (MSDOS 6.21, IO.SYS, SYSINIT:08B5h)
                                memerrjx:
00000926 BA[4951]                   mov     dx,badmem       ; "Configuration too large for memory"
00000929 E82841                     call    print
0000092C EB3A                       jmp     short continue

                                okld:
0000092E B43E                       mov     ah,3Eh ; CLOSE
00000930 CD21                       int     21h             ; close file

                                    ; 22/10/2022
00000932 5A                         pop     dx      ; (MSDOS 5.0 IO.SYS SYSINIT:087Dh)

                                    ; 24/03/2019

00000933 0E                         push    cs              ; point es to sysinitseg
00000934 07                         pop     es
00000935 BB[BF02]                       mov     bx,COMEXE ; point to exec block
                                    ; 22/10/2022
                                    ;pop    dx               ; recover pointer to name

                                ;;ifdef     MULTI_CONFIG

                                ;   If there's any environment data in "config_wrkseg", pass it to shell;
                                ;   there will be data if there were any valid SET commands and/or if a menu
                                ;   selection was made (in which case the CONFIG environment variable will be
                                ;   set to that selection).

                                    ; 23/10/2022
                                    ;mov    cx,[config_envlen]
                                    ;jcxz   no_envdata
                                        ;mov        cx,[config_wrkseg]
                                ;no_envdata:
                                    ;;mov   [bx+EXEC0.ENVIRON],cx
                                    ;mov    [bx],cx

                                ;;endif     ;MULTI_CONFIG

                                    ; 30/12/2022 - Retro DOS v4.2
                                    ; (MSDOS 6.21 IO.SYS SYSINIT:08C7h)
00000938 8B0E[6019]                 mov     cx,[config_envlen]
0000093C E304                       jcxz    no_envdata
0000093E 8B0E[6219]                     mov cx,[config_wrkseg]
                                no_envdata:
                                    ;mov    [bx+EXEC0.ENVIRON],cx
```

```
23805 00000942 890F                     mov     [bx],cx
23806
23807                                   ; 23/10/2022
23808                                   ; (MSDOS 5.0 IO.SYS SYSINIT:0883h)
23809
23810                                   ;mov    [bx+EXEC0.COM_LINE+2],cs ; set segments
23811 00000944 8C4F04                   mov     [bx+4],cs
23812                                   ;mov    [bx+EXEC0.5C_FCB+2],cs
23813 00000947 8C4F08                   mov     [bx+8],cs
23814                                   ;mov    [bx+EXEC0.6C_FCB+2],cs
23815 0000094A 8C4F0C                   mov     [bx+12],cs
23816
23817                                   ;mov    ax,(EXEC<<8) + 0
23818                                   ; 23/10/2022
23819                                   ;xor    ax,ax
23820                                   ;mov    ah,4Bh
23821                                   ; 04/07/2023
23822                                   ;mov    ax,4B00h
23823 0000094D B8004B                   mov     ax,(EXEC<<8)
23824
23825 00000950 F9                       stc                     ; in case of int 24
23826 00000951 CD21                         int     21h             ; go start up command
23827                                                               ; DOS - 2+ - LOAD OR EXECUTE (EXEC)
23828                                                               ; DS:DX -> ASCIZ filename
23829                                                               ; ES:BX -> parameter block
23830                                                               ; AL = subfunc: load & execute program
23831                                   ;push   cs
23832                                   ;pop    ds
23833
23834                                   ; 13/04/2024
23835                                   ; 23/10/2022
23836 00000953 52                       push    dx              ; push to balance fall-through pop
23837
23838                                 ; note fall through if exec returns (an error)
23839                                 comerr:
23840                                     ; 23/10/2022
23841                                 ;;ifdef     MULTI_CONFIG
23842                                     ;cmp    byte [commnd4],0
23843                                     ;je     short comerr2  ; all defaults exhausted, print err msg
23844                                     ;cmp    byte [newcmd],0
23845                                     ;je     short continue ; don't print err msg for defaults just yet
23846                                 ;comerr2:
23847                                 ;;endif
23848
23849                                     ; 30/12/2022 - Retro DOS v4.2
23850                                     ;push   cs
23851                                     ;pop    ds
23852                                     ; 07/04/2024
23853                                     ; ds = cs
23854
23855 00000954 803E[9E4B]00              cmp     byte [commnd4],0
23856 00000959 7407                      je      short comerr2  ; all defaults exhausted, print err msg
23857 0000095B 803E[2A4B]00              cmp     byte [newcmd],0
23858 00000960 7406                      je      short continue ; don't print err msg for defaults just yet
23859                                 comerr2:
23860                                     ; 07/04/2024
23861                                     ;push   dx ; 30/12/2022
23862
23863                                     ; 23/10/2022
23864 00000962 BA[C550]                     mov     dx,badcom ; want to print command error
23865 00000965 E8C040                   call    badfil
23866
23867                                     ; 07/04/2024
23868                                     ;pop    dx  ; 30/12/2022
23869                                 continue:
23870                                     ; 13/04/2024
23871                                     ; 23/10/2022
23872 00000968 5A                       pop     dx
23873
23874                                 ; 30/12/2022
23875                                 %if 0
23876
23877                                 ;;ifndef MULTI_CONFIG
23878                                     ;jmp    stall
23879                                     ; 24/10/2022
23880                                 stall:              ; (MSDOS 5.0 IO.SYS, SYSINIT:0899h)
23881                                     jmp     short stall
23882                                 ;;else
23883
23884                                 %endif
23885
23886                                 ; 30/12/2022 (MSDOS 6.21 SYSINIT, Retro DOS v4.2)
23887                                 ;%if 1
23888                                 ; 23/10/2022 (MSDOS 5.0 SYSINIT, Retrodos v4.0)
23889                                 ;%if 0
23890 00000969 B419                     mov     ah,GET_DEFAULT_DRIVE ; 19h
23891 0000096B CD21                     int     21h             ;
23892 0000096D 0441                     add     al,'A'          ;
23893 0000096F 88C2                     mov     dl,al           ; DL == default drive letter
23894 00000971 BE[6D4B]                 mov     si,commnd2
23895 00000974 803E[2A4B]00             cmp     byte [newcmd],0 ; if a SHELL= was given
23896 00000979 7505                     jne     short do_def2  ; then try the 2nd alternate;
23897 0000097B C60400                   mov     byte [si],0    ; otherwise, the default SHELL= was tried,
23898 0000097E EB05                     jmp     short do_def3   ; which is the same as our 2nd alt, so skip it
23899                                 do_def2:
23900 00000980 803C00                   cmp     byte [si],0    ; has 2nd alternate been tried?
23901 00000983 7554                        jne short do_alt    ; no
23902                                 do_def3:
23903 00000985 BE[7E4B]                 mov     si,commnd3
23904 00000988 803C00                   cmp     byte [si],0    ; has 3rd alternate been tried?
23905 0000098B 754C                     jne     short do_alt   ; no
23906 0000098D BE[9E4B]                 mov     si,commnd4
23907 00000990 803C00                   cmp     byte [si],0    ; has 4th alternate been tried?
23908 00000993 7544                     jne     short do_alt   ; no
23909 00000995 52                       push    dx              ;
23910 00000996 BA[3853]                 mov     dx,badcomprmpt
23911 00000999 E8B840                   call    print          ;
23912 0000099C 5A                       pop     dx              ; recover default drive letter in DL
23913                                 request_input:                ;
23914 0000099D B402                     mov     ah,STD_CON_OUTPUT
23915 0000099F CD21                     int     21h             ;
23916 000009A1 52                       push    dx              ;
23917 000009A2 B23E                     mov     dl,'>'          ;
23918 000009A4 CD21                     int     21h             ;
23919 000009A6 8A1E[2C4B]               mov     bl,[tmplate+1] ; [tmplate+1] = 12
23920 000009AA B700                     mov     bh,0            ;
23921 000009AC C687[2D4B]0D             mov     byte [commnd+bx],0Dh
23922 000009B1 BA[2B4B]                 mov     dx,tmplate
23923 000009B4 B40A                     mov     ah,STD_CON_STRING_INPUT
23924 000009B6 CD21                     int     21h             ; read a line of input
23925 000009B8 BA[7050]                 mov     dx,crlfm       ;
23926 000009BB E89640                   call    print          ;
23927 000009BE 5A                       pop     dx              ;
23928 000009BF 8A1E[2C4B]               mov     bl,[tmplate+1] ;
```

```
23929 000009C3 08DB                     or      bl,bl           ; was anything typed?
23930 000009C5 74D6                     jz      short request_input ;
23931 000009C7 C606[2A4B]01             mov     byte [newcmd],1 ; disable validation for user-specified binaries
23932 000009CC C687[2D4B]00             mov     byte [commnd+bx],0 ; NULL-terminate it before execing it
23933 000009D1 C706[BB4B]000D           mov     word [command_line],0D00h
23934 000009D7 EB35                     jmp     short do_exec   ;
23935                                 do_alt:
23936 000009D9 1E                       push    ds
23937 000009DA 07                       pop     es
23938 000009DB C606[2A4B]00             mov     byte [newcmd],0 ; force validation for alternate binaries
23939 000009E0 BF[2D4B]                 mov     di,commnd       ;
23940                                 do_alt1:
23941 000009E3 AC                       lodsb                   ; copy the alternate, zapping it as we go,
23942 000009E4 C644FF00                 mov     byte [si-1],0   ; so that we know it's been tried
23943 000009E8 AA                       stosb                   ;
23944 000009E9 08C0                     or      al,al           ;
23945 000009EB 75F6                     jnz     short do_alt1   ;
23946 000009ED BF[BB4B]                 mov     di,command_line
23947 000009F0 807C023A                 cmp     byte [si+2],':'
23948 000009F4 7503                     jne     short do_alt2   ;
23949 000009F6 885401                   mov     [si+1],dl       ; stuff default drive into alt. command line
23950                                 do_alt2:                  ;
23951 000009F9 AC                       lodsb                   ;
23952 000009FA AA                       stosb                   ;
23953 000009FB 08C0                     or      al,al           ;
23954 000009FD 75FA                     jnz     short do_alt2   ;
23955 000009FF C645FF0D                 mov     byte [di-1],cr
23956
23957                                 ;;   Last but not least, see if we need to call disable_autoexec
23958
23959                                     ; MSDOS 6.0 (SYSINIT1.ASM)
23960                                     ;cmp   [command_line-1],0
23961                                         ;jne       short do_exec   ;
23962                                         ;mov       [command_line-1],'/'
23963                                     ;call  disable_autoexec ;
23964
23965                                     ; MSDOS 6.21 IO.SYS (SYSINIT:0994h)
23966 00000A03 C606[7B4C]00             mov     byte [dae_flag],0 ; 24/03/2019 - Retro DOS v4.0
23967 00000A08 E80C3D                   call    disable_autoexec
23968 00000A0B E8543D                   call    CheckQueryOpt  ; 24/03/2019 - Retro DOS v4.0
23969                                 do_exec:
23970 00000A0E E999FE                   jmp     retry           ;
23971
23972                                 ;;endif    ;MULTI_CONFIG
23973
23974                                 ;%endif ; 23/10/2022 (MSDOS 5.0 SYSINIT)
23975                                 ;%endif ; 30/12/2022 (MSDOS 6.21 SYSINIT)
23976
23977                                 ; 24/03/2019 - Retro DOS v4.0
23978
23979                                 ; ----------------------------------------------------------------------
23980                                 ; procedure : AllocFreeMem
23981                                 ;
23982                                 ; Allocate Max memory from DOS to find out where to load DOS.
23983                                 ; DOS is at temporary location when this call is being made
23984                                 ;
23985                                 ; Inputs : None
23986                                 ; Outputs: The biggest chunk of memory is allocated (all mem at init time)
23987                                 ;      [area] & [memhi] set to the para value of the start of the
23988                                 ;      free memory.
23989                                 ;
23990                                 ; Uses   : AX, BX
23991                                 ;
23992                                 ; ----------------------------------------------------------------------
23993
23994                                     ; 30/12/2022 - Retro DOS v4.2
23995                                     ; (MSDOS 6.21 IO.SYS, SYSINIT:09A2h)
23996
23997                                     ; 08/04/2024 - Retro DOS v5.0
23998                                     ; (PCDOS 7.1 IBMBIO.COM, SYSINIT:0AB5h)
23999
24000                                     ; 23/10/2022
24001                                 AllocFreeMem:
24002 00000A11 BBFFFF                   mov     bx,0FFFFh
24003 00000A14 B448                     mov     ah,48h ; ALLOC
24004 00000A16 CD21                     int     21h                     ; first time fails
24005 00000A18 B448                     mov     ah,48h ; ALLOC
24006 00000A1A CD21                     int     21h                     ; second time gets it
24007                                   ; 11/12/2022
24008                                   ; ds = cs
24009                                   ;mov    [cs:area],ax
24010                                   ;mov    [cs:memhi],ax           ; memhi:memlo now points to
24011 00000A1C A3[6803]                 mov     [area],ax
24012 00000A1F A3[6403]                 mov     [memhi],ax              ; memhi:memlo now points to
24013 00000A22 C3                       retn                            ; start of free memory
24014
24015                                   ; include msbio.cl6
24016                                 ; ----------------------------------------------------------------------
24017                                 DOSLOMSG:
24018 00000A23 484D41206E6F742061-      db      'HMA not available: Loading DOS low',0Dh,0Ah,'$'
24018 00000A2C 7661696C61626C653A-
24018 00000A35 204C6F6164696E6720-
24018 00000A3E 444F53206C6F770D0A-
24018 00000A47 24
24019                                 FEmsg:
24020 00000A48 466174616C20457272-      db      'Fatal Error: Cannot allocate Memory for DOS',0Dh,0Ah,'$'
24020 00000A51 6F723A2043616E6E6F-
24020 00000A5A 7420616C6C6F636174-
24020 00000A63 65204D656D6F727920-
24020 00000A6C 666F7220444F530D0A-
24020 00000A75 24
24021
24022                                 ; ----------------------------------------------------------------------
24023                                 ;
24024                                 ; procedure : LoadDOSHiOrLo
24025                                 ;
24026                                 ;           Tries to move DOS into HMA. If it fails then loads
24027                                 ;           DOS into Low memory. For ROMDOS, nothing is actually
24028                                 ;           moved; this just tries to allocate the HMA, and prints
24029                                 ;           a message if this is not possible.
24030                                 ;
24031                                 ; ----------------------------------------------------------------------
24032
24033                                     ; 23/10/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 IO.SYS)
24034                                 LoadDOSHiOrLo:
24035                                     ; 27/03/2019 - Retro DOS v4.0
24036                                     ; ds = cs
24037 00000A76 E81F00                   call    TryToMovDOSHi           ; Try moving it into HMA (M024)
24038                                   ;jc     short LdngLo            ; If that don't work...
24039                                   ;retn
24040                                     ; 18/12/2022
24041 00000A79 731C                     jnc     short LoadDosHi_ok
24042                                 LdngLo:
24043                                     ; 23/10/2022
```

```
                                    ;push   cs
                                    ;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
```

```
24168                                   ; ES:DI points to free HMA after DOS
24169                                 ;else
24170                                 ;   ; allocate space at beginning of HMA to allow for CPMHack
24171                                 ;
24172                                 ;   mov     di,0E0h                     ; room for 5 bytes at ffff:d0
24173                                 ;
24174                                 ;endif ; ROMDOS
24175
24176 00000AC2 E87602                   call    SaveFreeHMAPtr              ; Save the Free HMA ptr
24177 00000AC5 F8                       clc
24178                                 mdhx:
24179 00000AC6 C3                       retn
24180
24181                                 ; ----------------------------------------------------------------------
24182                                 ;
24183                                 ; procedure : MovDOSLo
24184                                 ;
24185                                 ;           Allocates memory from DOS and moves BIOS/DOS code into it
24186                                 ;
24187                                 ; ----------------------------------------------------------------------
24188
24189                                   ; 23/10/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 IO.SYS)
24190
24191                                 ;ifndef ROMDOS
24192
24193                                 MovDOSLo:
24194                                   ; 14/05/2019
24195                                   ; 27/03/2019 - Retro DOS v4.0
24196                                   ; ds = cs
24197 00000AC7 E84500                   call    AllocMemForDOS              ; incestuosly!!!
24198
24199                                   ; 23/10/2022
24200                                   ; 14/05/2019
24201                                   ;inc    ax  ; skip MCB
24202
24203 00000ACA 8EC0                     mov     es,ax                       ; pass the segment to MovBIOS
24204                                   ; NOTE: Retro DOS v4.0 does not move BIOS (IO.SYS) to HMA
24205                                   ; 24/03/2019
24206
24207                                   ; 23/10/2022
24208 00000ACC E81E00                   call    MovBIOS
24209
24210                                 ;------ ES:DI points memory immediately after BIOS
24211
24212                                   ; 14/05/2019
24213                                   ; NOTE:
24214                                   ;     Order of (RETRO) DOS kernel sections at memory:
24215                                   ;       BIOSDATA+BIOSCODE+BIOSDATAINIT+DOSDATA+DOSCODE(LOW)
24216
24217                                   ; 24/03/2019 - Retro DOS v4.0
24218                                   ;xor    di,di
24219
24220                                   ; 23/10/2022
24221                                   ;mov    cx,[cs:lo_doscod_size]      ; DOS code size when loaded
24222                                   ; 11/12/2022
24223                                   ; ds = cs
24224 00000ACF 8B0E[8102]              mov     cx,[lo_doscod_size]         ; low
24225                                   ;call   MovDOS
24226                                   ;retn
24227                                   ; 11/12/2022
24228                                   ;jmp    short MovDOS
24229
24230                                 ;endif ; ROMDOS
24231
24232                                 ; 11/12/2022
24233
24234                                 ; ----------------------------------------------------------------------
24235                                 ;
24236                                 ; procedure : MovDOS
24237                                 ;
24238                                 ;           Moves DOS code into requested area
24239                                 ;
24240                                 ;   In : ES:DI - pointer to memory where DOS is to be moved
24241                                 ;        CX    - size of DOS code to be moved
24242                                 ;
24243                                 ;   Out : ES:DI - pointer to memory immediately after DOS
24244                                 ;
24245                                 ; ----------------------------------------------------------------------
24246
24247                                   ; 11/12/2022
24248                                   ; 23/10/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 IO.SYS)
24249
24250                                 ;ifndef ROMDOS
24251
24252                                 MovDOS:
24253                                   ; 14/05/2019
24254                                   ; 27/03/2019 - Retro DOS v4.0
24255
24256                                   ; 11/12/2022
24257                                   ; ds = cs
24258
24259                                   ; 23/10/2022
24260                                   ;push   ds ; *//
24261
24262 00000AD3 06                       push    es
24263 00000AD4 57                       push    di
24264
24265                                   ; 11/12/2022
24266 00000AD5 1E                       push    ds ; *// ; 11/12/202
24267
24268                                   ; 29/04/2019
24269 00000AD6 C536[7102]               lds     si,[dosinit] ; 11/12/2022
24270                                   ; 23/10/2022
24271                                   ;lds    si,[cs:dosinit]
24272                                   ; 03/09/2023
24273 00000ADA 89F0                     mov     ax,si
24274
24275 00000ADC F3A4                     rep     movsb
24276
24277 00000ADE 1F                       pop     ds ; *// ; 11/12/2022
24278
24279 00000ADF 5B                       pop     bx                          ; get back offset into which
24280                                                                       ;  DOS was moved
24281                                   ; 03/09/2023
24282                                   ;;mov   ax,[cs:dosinit]                     ; get the offset at which DOS
24283                                                                       ;  wants to run
24284                                   ; 03/09/2023
24285                                   ;mov    ax,[dosinit]
24286                                   ; ax = [dosinit]
24287
24288 00000AE0 29D8                     sub     ax,bx
24289 00000AE2 E8B502                   call    off_to_para
24290 00000AE5 5B                       pop     bx                          ; get the segment at which
24291                                                                       ;  we moved DOS into
```

```
24292 00000AE6 29C3                       sub     bx,ax                           ; Adjust segment
24293
24294                                     ; 11/12/2022
24295                                     ; 23/10/2022
24296                                     ;mov    [cs:CURRENT_DOS_LOCATION],bx  ; and save it
24297                                     ;;mov   [cs:FINAL_DOS_LOCATION],bx
24298                                     ; 11/12/2022
24299 00000AE8 891E[7302]                 mov     [CURRENT_DOS_LOCATION],bx
24300
24301                                     ; 27/03/2019
24302                                     ;pop    ds ; *//
24303                                     ; ds = cs
24304                                     ;mov    [FINAL_DOS_LOCATION],bx
24305
24306 00000AEC C3                         retn
24307
24308                                   ;endif ;ROMDOS
24309
24310                                   ; NOTE: Retro DOS v4.0 does not move BIOS (IO.SYS) to HMA
24311                                   ; 24/03/2019
24312                                   ; ---------------------------------------------------------------------------
24313                                   ;
24314                                   ; procedure : MovBIOS
24315                                   ;
24316                                   ;           Moves BIOS code into requested segment
24317                                   ;
24318                                   ;   In : ES - segment to which BIOS is to be moved
24319                                   ;             ( it moves always into offset BCode_Start)
24320                                   ;
24321                                   ;   Out : ES:DI - pointer to memory immediately after BIOS
24322                                   ;
24323                                   ; ---------------------------------------------------------------------------
24324
24325                                       ; 23/10/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 IO.SYS)
24326                                       ; 08/04/2024 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMBIO.COM)
24327
24328                                   ;ifndef ROMDOS
24329
24330                                   MovBIOS: ; proc    near
24331                                       ; 11/12/2022
24332 00000AED 1E                         push    ds ; ds = cs
24333                                     ;
24334                                     ; 23/10/2022
24335                                     ;mov    ds,[cs:temp_bcode_seg]        ; current BIOS code seg
24336                                     ; 17/09/2023 ; 08/04/2024
24337 00000AEE 8E1E[8902]                 mov     ds,[temp_bcode_seg]
24338                                     ;mov    si,BCODE_START ; mov si,30h
24339                                     ; 09/12/2022
24340 00000AF2 BE[3000]                   mov     si,BCODESTART ; 30h
24341 00000AF5 89F7                       mov     di,si
24342                                     ;mov    cx,1E00h ; PCDOS 7.1 IBMBIO.COM ; 08/04/2024
24343 00000AF7 B9701D                     mov     cx,BCODE_END ; mov cx,1A60h
24344 00000AFA 29F1                       sub     cx,si                           ; size of BIOS
24345 00000AFC D1E9                       shr     cx,1                            ; Both the labels are para
24346                                                                             ;  aligned
24347 00000AFE F3A5                       rep     movsw
24348
24349                                     ; 11/12/2022
24350 00000B00 1F                         pop     ds ; ds = cs
24351                                     ;
24352 00000B01 06                         push    es
24353 00000B02 57                         push    di                              ; save end of BIOS
24354 00000B03 8CC0                       mov     ax,es
24355                                     ;
24356                                     ; 11/12/2022
24357                                     ;mov    [cs:BCodeSeg],ax              ; save it for later use
24358                                     ;;call  dword ptr cs:_seg_reinit_ptr
24359                                     ;call   far [cs:seg_reinit_ptr]               ; far call to seg_reinit (M022)
24360                                     ; ds = cs
24361 00000B05 A3[8A03]                   mov     [BCodeSeg],ax
24362 00000B08 FF1E[8702]                 call    far [seg_reinit_ptr]
24363                                     ;
24364 00000B0C 5F                         pop     di
24365 00000B0D 07                         pop     es                              ; get back end of BIOS
24366 00000B0E C3                         retn
24367
24368                                   ;MovBIOS endp
24369
24370                                   ;endif ; ROMDOS
24371
24372                                   ; 11/12/2022
24373                                   %if 0
24374
24375                                   ; 24/03/2019
24376
24377                                   ; ---------------------------------------------------------------------------
24378                                   ;
24379                                   ; procedure : MovDOS
24380                                   ;
24381                                   ;           Moves DOS code into requested area
24382                                   ;
24383                                   ;   In : ES:DI - pointer to memory where DOS is to be moved
24384                                   ;        CX    - size of DOS code to be moved
24385                                   ;
24386                                   ;   Out : ES:DI - pointer to memory immediately after DOS
24387                                   ;
24388                                   ; ---------------------------------------------------------------------------
24389
24390                                       ; 23/10/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 IO.SYS)
24391
24392                                   ;ifndef ROMDOS
24393
24394                                   MovDOS:
24395                                       ; 14/05/2019
24396                                       ; 27/03/2019 - Retro DOS v4.0
24397
24398                                       ; 11/12/2022
24399                                       ; ds = cs
24400
24401                                       ; 23/10/2022
24402                                       ;push   ds ; *//
24403
24404                                       push    es
24405                                       push    di
24406
24407                                       ; 11/12/2022
24408                                       push    ds ; *// ; 11/12/202
24409
24410                                       ; 29/04/2019
24411                                       lds     si,[dosinit] ; 11/12/2022
24412                                       ; 23/10/2022
24413                                       ;lds    si,[cs:dosinit]
24414                                       ; 03/09/2023
24415                                       mov     ax,si
```

```
24416
24417                                   rep     movsb
24418
24419                                   pop     ds ; *// ; 11/12/2022
24420
24421                                   pop     bx                              ; get back offset into which
24422                                                                           ;  DOS was moved
24423                                   ;mov    ax,[dosinit] ; 03/09/2023
24424                                   ;;mov   ax,[cs:dosinit]                         ; get the offset at which DOS
24425                                                                           ;  wants to run
24426                                   sub     ax,bx
24427                                   call    off_to_para
24428                                   pop     bx                              ; get the segment at which
24429                                                                           ;  we moved DOS into
24430                                   sub     bx,ax                           ; Adjust segment
24431
24432                                   ; 11/12/2022
24433                                   ; 23/10/2022
24434                                   ;mov    [cs:CURRENT_DOS_LOCATION],bx  ; and save it
24435                                   ;;mov   [cs:FINAL_DOS_LOCATION],bx
24436                                   ; 11/12/2022
24437                                   mov     [CURRENT_DOS_LOCATION],bx
24438
24439                                   ; 27/03/2019
24440                                   ;pop    ds ; *//
24441                                   ; ds = cs
24442                                   ;mov    [FINAL_DOS_LOCATION],bx
24443
24444                                   retn
24445
24446                                ;endif ;ROMDOS
24447
24448                                %endif
24449
24450                                ; ----------------------------------------------------------------------
24451                                ;
24452                                ; procedure : AllocMemForDOS
24453                                ;
24454                                ;           Allocate memory for DOS/BIOS code from DOS !!!
24455                                ;
24456                                ;   Out : AX - seg of allocated memoryblock
24457                                ;
24458                                ; ----------------------------------------------------------------------
24459
24460                                    ; 23/10/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 IO.SYS)
24461                                    ; 08/04/2024 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMBIO.COM)
24462
24463                                ;ifndef ROMDOS
24464
24465                                AllocMemForDOS:
24466                                    ; 11/12/2022
24467                                    ; 14/05/2019
24468                                    ; 27/03/2019 - Retro DOS v4.0
24469                                    ; ds = cs
24470                                    ;mov    ax,BCode_end
24471                                    ;sub    ax,BCode_start        ; BIOS code size
24472                                    ; 23/10/2022
24473 00000B0F B8701D                  mov     ax,BCODE_END ; 1A60h  ; 1A70h for MSDOS 6.21
24474                                                                  ; 30/12/2022
24475                                   ;mov    ax,1E00h ; PCDOS 7.1 IBMBIO.COM ; 08/04/2024
24476                                   ;sub    ax,BCODE_START ; 30h
24477                                   ; 09/12/2022
24478 00000B12 2D[3000]                 sub     ax,BCODESTART ; sub ax,30h ; 08/04/2024
24479                                   ; 24/03/2019 - Retro DOS v4.0
24480                                   ; 02/11/2022
24481                                   ;add    ax,[cs:lo_doscod_size]; DOS code size
24482                                   ; 11/12/2022
24483                                   ; ds = cs
24484 00000B15 0306[8102]              add     ax,[lo_doscod_size]
24485 00000B19 83C00F                  add     ax,15
24486 00000B1C E87B02                  call    off_to_para                     ; convert to para
24487                                   ; 23/10/2022
24488                                   ; 14/05/2019
24489                                   ;inc    ax ; + 1 paragraph for MCB
24490 00000B1F 09DB                    or      bx,bx                           ; M012
24491 00000B21 89C3                    mov     bx,ax                           ;  can we use int 21 for alloc
24492 00000B23 741A                    jz      short update_arena              ; M012
24493 00000B25 B448                    mov     ah,48h                          ; request DOS
24494 00000B27 CD21                    int     21h
24495 00000B29 7250                    jc      short FatalErr                  ; IF ERR WE ARE HOSED
24496                                   ; 23/10/2022
24497                                   ; 24/03/2019 - Retro DOS v4.0 (ORG 0)
24498 00000B2B 83E803                  sub     ax,3                            ; Take care ORG 30h of
24499                                                                          ;  BIOS code
24500 00000B2E 8EC0                    mov     es,ax
24501                                   ;mov    word [es:20h+ARENA.OWNER],08h ; mark it as system
24502                                   ;mov    word [es:20h+ARENA.NAME],'SC' ;  code area
24503                                   ; 14/05/2019
24504                                   ;mov    word [es:ARENA.OWNER],08h     ; mark it as system
24505                                   ;mov    word [es:ARENA.NAME],'SC'     ;  code area
24506                                   ; 08/04/2024 (PCDOS 7.1 IBMBIO.COM)
24507                                   ; 23/10/2022
24508 00000B30 26C70621000800          mov     word [es:20h+1],08h             ; mark it as system
24509 00000B37 26C70628005343          mov     word [es:20h+8],'SC' ; 4353h  ;  code area
24510
24511 00000B3E C3                      retn
24512
24513                                ; BUGBUG -- 5 Aug 92 -- chuckst -- Allocating space for DOS
24514                                ;     using DOS itself causes an arena to be generated.
24515                                ;     Unfortunately, certain programs (like PROTMAN$)
24516                                ;     assume that the device drivers are loaded into
24517                                ;     the first arena. For this reason, MagicDrv's
24518                                ;     main device driver header arena is manually
24519                                ;     truncated from the arena chain, and the space
24520                                ;     for DOS is allocated using the following
24521                                ;     simple code, which also assumes that the
24522                                ;     first arena is the free one where DOS's low
24523                                ;     stub will go.
24524                                ;
24525                                ; M012 : BEGIN
24526
24527                                    ; 23/10/2022
24528                                update_arena:
24529 00000B3F 1E                      push    ds ; ds = cs
24530 00000B40 57                      push    di
24531 00000B41 51                      push    cx
24532 00000B42 52                      push    dx
24533                                   ; 23/10/2022
24534                                   ;lds    di,[cs:DOSINFO]                         ; get ptr to DOS var
24535                                   ; 11/12/2022
24536                                   ; ds = cs
24537 00000B43 C53E[6D02]              lds     di,[DOSINFO] ; 27/03/2019
24538 00000B47 4F                      dec     di
24539 00000B48 4F                      dec     di                              ; Arena head is immediately
```

```
                                                        ;  before sysvar
00000B49 8E05                   mov     es,[di]         ; es = arena head
                                ;mov    cx,[es:ARENA.SIZE]      ; cx = total low mem size
00000B4B 268B0E0300             mov     cx,[es:3]
00000B50 39D9                   cmp     cx,bx           ; is it sufficient ?
00000B52 7227                   jb      short FatalErr  ; no, fatal error

                                ;mov    dl,[es:ARENA.SIGNATURE]
00000B54 268A160000             mov     dl,[es:0]
00000B59 8CC0                   mov     ax,es
00000B5B 01D8                   add     ax,bx           ; ax = new arena head
00000B5D 8905                   mov     [di],ax         ; store it in DOS data area
00000B5F 8ED8                   mov     ds,ax
                                ;mov    [ARENA.SIGNATURE],dl    ; type of arena
00000B61 88160000               mov     [0],dl
                                ;mov    word [ARENA.OWNER],0    ; free
00000B65 C70601000000           mov     word [1],0
00000B6B 29D9                   sub     cx,bx           ; size of the new block
                                ;mov    [ARENA.SIZE],cx         ; store it in the arena
00000B6D 890E0300               mov     [3],cx
00000B71 8CC0                   mov     ax,es           ; return seg to the caller
                                ; 23/10/2022
                                ;; 24/03/2019 - Retro DOS v4.0 (ORG 0)      ; Take care ORG 30h of
00000B73 83E803                 sub     ax,3            ;  BIOS code
00000B76 5A                     pop     dx
00000B77 59                     pop     cx
00000B78 5F                     pop     di
00000B79 1F                     pop     ds ; ds = cs
00000B7A C3                     retn
                        ;
                        ; M012 : END
                        ;
                        FatalErr:
00000B7B 0E                     push    cs
00000B7C 1F                     pop     ds
00000B7D BA[480A]               mov     dx,FEmsg
00000B80 B409                   mov     ah,9
00000B82 CD21                   int     21h             ; DOS - PRINT STRING
                                        ; DS:DX -> string terminated by "$"
                                ; 30/12/2022 (MSDOS 6.21 SYSINIT)
00000B84 E9C707                 jmp     stall
                                ; 23/10/2022 (MSDOS 5.0 SYSINIT)
                                ;cli
                                ;hlt

                        ;endif ;ROMDOS

                        ; 25/03/2019 - Retro DOS v4.0

                        ; --------------------------------------------------------------------
                        ;
                        ; procedure : AllocHMA
                        ;
                        ;   grab_the_hma tries to enable a20 and make sure there is memory
                        ;     up there. If it gets any sort of error, it will return with
                        ;     carry set so that we can resort to running low.
                        ;
                        ;   It also returns ES: -> 0ffffh if it returns success
                        ;
                        ; --------------------------------------------------------------------

                        AllocHMA:
                        ;   cas note:  The pre-286 check is no longer needed here since the
                        ;              presence of XMS is sufficient. However, this code hasn't
                        ;              been deleted because it can be recycled for skipping the
                        ;              extra pass of CONFIG.SYS and assuming we're running low
                        ;              in the case of a pre-286.

                        ;;  see if we're running on a pre-286. If not, force low.
                        ;
                        ;   xor     ax,ax
                        ;   pushf                   ; save flags (like int)
                        ;   push    ax
                        ;   popf
                        ;   pushf
                        ;   pop     ax
                        ;   popf                    ; restore original flags (like int)
                        ;   and     ax,0F000h
                        ;   cmp     ax,0F000h       ; 8088/8086?
                        ;   jz      short grab_hma_error

                                ; 23/10/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 IO.SYS)
                                ; (SYSINIT:0A26h)

                                ; 13/04/2024 - Retro DOS v5.0
                                ; (PCDOS 7.1 IBMBIO.COM - SYSINIT:0C41h)

00000B87 1E                     push    ds
                                ;;mov   ax,Bios_Data
                                ;mov    ax,KERNEL_SEGMENT
                                ; 21/10/2022
00000B88 B87000                 mov     ax,DOSBIODATASEG ; 70h
00000B8B 8ED8                   mov     ds,ax

00000B8D E84A00                 call    IsXMSLoaded
00000B90 7545                   jnz     short grabhma_error

00000B92 B81043                 mov     ax,4310h
00000B95 CD2F                   int     2Fh             ; get the vector into es:bx
                                        ; - Multiplex - XMS - GET DRIVER ADDRESS
                                        ; Return: ES:BX -> driver entry point

00000B97 891E[0E00]             mov     [xms],bx
                                ;mov    [0Eh], bx
00000B9B 8C06[1000]             mov     [xms+2],es
                                ;mov    [10h],es

00000B9F B401                   mov     ah,1            ; request HMA
00000BA1 BAFFFF                 mov     dx,0FFFFh
                                ;call   dword ptr ds:0Eh
00000BA4 FF1E[0E00]             call    far [xms]
00000BA8 48                     dec     ax
00000BA9 7409                   jz      short allocHMA_1 ; error if not able to allocate HMA

                        ;------ Himem may be lying because it has allocated mem for int 15

00000BAB B488                   mov     ah,88h
00000BAD CD15                   int     15h
                                        ; Get Extended Memory Size
                                        ; Return: CF clear on success
                                        ; AX = size of memory above 1M in K
00000BAF 83F840                 cmp     ax,64           ; less than 64 K of hma ?
                                ;jb     short grabhma_error
                                ; 11/12/2022
```

```
24664 00000BB2 7224                     jb      short grabhma_err ; cf=1
24665                                 allocHMA_1:
24666 00000BB4 B405                     mov     ah,5            ; localenableA20
24667                                   ;call   dword ptr ds:0Eh
24668 00000BB6 FF1E[0E00]               call    far [xms]
24669 00000BBA 48                       dec     ax
24670 00000BBB 751A                     jnz     short grabhma_error ; error if couldn't enable A20
24671
24672 00000BBD E89D01                   call    IsVDiskInstalled
24673 00000BC0 7415                     jz      short grabhma_error ; yes, we cant use HMA
24674
24675 00000BC2 B8FFFF                   mov     ax,0FFFFh
24676 00000BC5 8EC0                     mov     es,ax
24677 00000BC7 26C70610003412           mov     word [es:10h],1234h ; see if we can really read/write there
24678 00000BCE 26813E10003412           cmp     word [es:10h],1234h
24679                                   ;jne    short grabhma_error ; don't try to load there if XMS lied
24680                                   ; 11/12/2022
24681 00000BD5 7401                     je      short allocHMA_ok
24682
24683                                 ; 11/12/2022
24684                                 ;   ; 11/12/2022
24685                                 ;   ; cf=0
24686                                 ;   ;clc
24687                                 ;   pop     ds
24688                                 ;   retn
24689
24690                                 grabhma_error:
24691 00000BD7 F9                       stc
24692                                   ; 11/12/022
24693                                 grabhma_err:        ; cf=1
24694                                 allocHMA_ok:        ; cf=0
24695 00000BD8 1F                       pop     ds
24696 00000BD9 C3                       retn
24697
24698                                 ; ---------------------------------------------------------------------------
24699                                 ;
24700                                 ; procedure : IsXMSLoaded
24701                                 ;
24702                                 ;             Checks whether a XMS driver is loaded
24703                                 ;
24704                                 ; Returns : Z flag set if XMS driver loaded
24705                                 ;           Z flag reset if no XMS drivers are present
24706                                 ;
24707                                 ; ---------------------------------------------------------------------------
24708
24709                                 IsXMSLoaded:
24710 00000BDA B80043                     mov     ax,4300h
24711 00000BDD CD2F                       int     2Fh             ; - Multiplex - XMS - INSTALLATION CHECK
24712                                                             ; Return: AL = 80h XMS driver installed
24713                                                             ; AL <> 80h no driver
24714 00000BDF 3C80                       cmp     al,80h          ; XMS installed?
24715 00000BE1 C3                         retn
24716
24717                                 ; ---------------------------------------------------------------------------
24718                                 ; procedure : FTryToMovDOSHi
24719                                 ;
24720                                 ;           Called from HMA suballoc calls
24721                                 ;
24722                                 ; ---------------------------------------------------------------------------
24723
24724                                     ; 23/10/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 IO.SYS)
24725                                     ; (MSDOS 5.0 IO.SYS - SYSINIT:0A84h)
24726
24727                                     ; 08/04/2024 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMBIO.COM)
24728                                     ; (PCDOS 7.1 IBMBIO.COM - SYSINIT:0C9Fh)
24729
24730                                     ; ((MSDOS 6.22 IO.SYS - SYSINIT:0B8Ch))
24731
24732                                 FTryToMovDOSHi:    ; proc  far
24733
24734 00000BE2 50                         push    ax
24735 00000BE3 53                         push    bx
24736 00000BE4 51                         push    cx
24737 00000BE5 52                         push    dx
24738 00000BE6 56                         push    si
24739 00000BE7 57                         push    di
24740 00000BE8 1E                         push    ds
24741 00000BE9 06                         push    es
24742
24743                                     ; 23/10/2022
24744                                     ; 27/03/2019 - Retro DOS v4.0
24745                                     ; 11/12/2022
24746 00000BEA 0E                         push    cs
24747 00000BEB 1F                         pop     ds
24748
24749                                     ;cmp    byte [cs:runhigh],0FFh
24750                                     ; 11/12/2022
24751 00000BEC 803E[6C02]FF               cmp     byte [runhigh],0FFh
24752 00000BF1 7503                       jne     short _ftymdh_1
24753
24754                                     ; ds = cs
24755 00000BF3 E8A2FE                     call    TryToMovDOSHi
24756                                 _ftymdh_1:
24757 00000BF6 07                         pop     es
24758 00000BF7 1F                         pop     ds
24759 00000BF8 5F                         pop     di
24760 00000BF9 5E                         pop     si
24761 00000BFA 5A                         pop     dx
24762 00000BFB 59                         pop     cx
24763 00000BFC 5B                         pop     bx
24764 00000BFD 58                         pop     ax
24765
24766 00000BFE CB                         retf
24767
24768                                 ; ---------------------------------------------------------------------------
24769                                 ;
24770                                 ; following piece of code will be moved into a para boundary. And the para
24771                                 ; address posted in seg of int 19h vector. Offset of int 19h will point to
24772                                 ; VDint19. This is to protect HMA from apps which use VDISK header method
24773                                 ; to determine free extended memory.
24774                                 ;
24775                                 ; For more details read "power programming" column by Ray Duncan in the
24776                                 ; May 30 1989 issue of PC Magazine (pp 377-388) [USING EXTENDED MEMORY,PART 1]
24777                                 ;
24778                                 ; ---------------------------------------------------------------------------
24779
24780                                     ; 30/12/2023 - Retro DOS 5.0
24781 00000BFF 00                       db      0
24782
24783                                 ; 13/04/2024
24784                                 ;align 2
24785
24786                                     ; 30/12/2023
24787                                     ; PCDOS v7.1 IBMBIO.COM, SYSYINIT:0CBCh
```

```
24788
24789                                   StartVDHead:
24790                                   ;-------------- what follows is a dummy device driver header (not used by DOS)
24791
24792 00000C00 00000000                    dd      0               ; link to next device driver
24793 00000C04 0080                        dw      8000h           ; device attribute
24794 00000C06 0000                        dw      0               ; strategy routine offset
24795 00000C08 0000                        dw      0               ; interrupt routine offset
24796 00000C0A 01                          db      1               ; number of units
24797                                      ;db     7 dup(0)
24798 00000C0B 00<rep 7h>                  times   7 db 0          ; reserved area
24799                                   VDiskSig1:
24800 00000C12 564449534B                  db      'VDISK'
24801
24802                                   VLEN1       equ     ($-VDiskSig1)
24803
24804 00000C17 202056332E33                db      '  V3.3'        ; vdisk label
24805                                      ;db     15 dup (0)      ; pad
24806 00000C1D 00<rep Fh>                  times   15 db 0
24807 00000C2C 0000                        dw      0               ; bits 0-15 of free HMA
24808 00000C2E 11                          db      11h             ; bits 16-23 of free HMA (1M + 64K)
24809                                   VDInt19:
24810 00000C2F EA                          db      0EAh            ; jmp to old vector
24811                                   OldVDInt19:
24812 00000C30 00000000                    dd      0               ; Saved int 19 vector
24813
24814                                   EndVDHead: ; label byte
24815
24816                                   VDiskHMAHead:
24817 00000C34 000000                      db      0,0,0           ; non-bootable disk
24818                                   VDiskSig2:
24819 00000C37 564449534B                  db      'VDISK'
24820
24821                                   VLEN2       equ     ($-VDiskSig2)
24822
24823 00000C3C 332E33                      db      '3.3'           ; OEM - signature
24824 00000C3F 8000                        dw      128             ; number of bytes/sector
24825 00000C41 01                          db      1               ; sectors/cluster
24826 00000C42 0100                        dw      1               ; reserved sectors
24827 00000C44 01                          db      1               ; number of FAT copies
24828 00000C45 4000                        dw      64              ; number of root dir entries
24829 00000C47 0002                        dw      512             ; number of sectors
24830 00000C49 FE                          db      0FEh            ; media descriptor
24831 00000C4A 0600                        dw      6               ; number of sectors/FAT
24832 00000C4C 0800                        dw      8               ; sectors per track
24833 00000C4E 0100                        dw      1               ; number of heads
24834 00000C50 0000                        dw      0               ; number of hidden sectors
24835 00000C52 4004                        dw      440h            ; Start of free HMA in K (1M+64K)
24836
24837                                   EndVDiskHMAHead: ; label byte
24838
24839                                   ; ------------------------------------------------------------------------
24840                                   ;
24841                                   ; procedure : InstVDiskHeader
24842                                   ;
24843                                   ;             Installs the VDISK header to reserve the 64k of HMA
24844                                   ;         It puts a 32 byte header at 10000:0 and
24845                                   ;         another header at (seg of int19):0
24846                                   ;
24847                                   ; Inputs : None
24848                                   ;
24849                                   ; Outputs : None
24850                                   ;
24851                                   ; USES : DS,SI,AX,CX,DX
24852                                   ;
24853                                   ; ------------------------------------------------------------------------
24854
24855                                       ; 23/10/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 IO.SYS)
24856
24857                                   InstVDiskHeader:
24858 00000C54 31C0                        xor     ax,ax
24859 00000C56 8ED8                        mov     ds,ax                   ; seg of int vect table
24860
24861                                   ;-------------- save old int 19 vector
24862
24863                                       ; 23/10/2022
24864 00000C58 A16400                      mov     ax,[19h*4]
24865                                      ;mov    [OldVDInt19],ax
24866 00000C5B 2EA3[300C]                  mov     [cs:OldVDInt19],ax
24867 00000C5F A16600                      mov     ax,[19h*4+2]
24868                                      ;mov    [OldVDInt19+2],ax
24869 00000C62 2EA3[320C]                  mov     [cs:OldVDInt19+2],ax
24870
24871                                   ;-------------- calculate seg of new int 19 handler
24872
24873 00000C66 B448                        mov     ah,48h                  ; allocate memory
24874                                      ;mov    bx,(EndVDHead-StartVDHead+15)>>4
24875                                       ; 23/10/2022
24876 00000C68 BB0400                      mov     bx,4
24877 00000C6B CD21                        int     21h
24878
24879                                   ;   if carry, fatal hanging error!!!!!
24880
24881 00000C6D 48                          dec     ax                      ; point to arena
24882 00000C6E 8EC0                        mov     es,ax
24883                                      ;mov    word [es:ARENA.OWNER],8       ; owner = System
24884 00000C70 26C70601000800              mov     word [es:1],8
24885                                      ;mov    word [es:ARENA.NAME],'SC' ; System Code
24886 00000C77 26C70608005343              mov     word [es:8],'SC' ; 4353h
24887 00000C7E 40                          inc     ax
24888 00000C7F 8EC0                        mov     es,ax                   ; get back to allocated memory
24889
24890                                   ;-------------- install new int 19 vector
24891
24892 00000C81 FA                          cli                             ; no reboots at this time
24893                                      ;mov    word [19h*4],(VDInt19-StartVDHead)
24894 00000C82 C70664002F00                mov     word [19h*4],47
24895 00000C88 A36600                      mov     [19h*4+2],ax
24896
24897                                   ;-------------- move the code into proper place
24898
24899                                      ;mov    cx,(EndVDHead-StartVDHead)
24900 00000C8B B93400                      mov     cx,52
24901 00000C8E BE[000C]                    mov     si,StartVDHead
24902 00000C91 31FF                        xor     di,di
24903 00000C93 0E                          push    cs
24904 00000C94 1F                          pop     ds
24905 00000C95 FC                          cld
24906 00000C96 F3A4                        rep     movsb
24907 00000C98 FB                          sti                             ; BUGBUG is sti OK now?
24908
24909                                   ;-------------- mov the HMA VDisk head into HMA
24910
24911                                       ; 23/10/2022
```

```
00000C99 57                             push    di
00000C9A 06                             push    es

                                        ;mov    ax,0FFFFh
                                        ;mov    es,ax
                                        ; 03/09/2023
00000C9B 49                             dec     cx
                                        ; cx = 0FFFFh
00000C9C 8EC1                           mov     es,cx

00000C9E BF1000                         mov     di,10h
                                        ;mov    cx,(EndVDiskHMAHead-VDiskHMAHead)
00000CA1 B92000                         mov     cx,32
00000CA4 BE[340C]                       mov     si,VDiskHMAHead
00000CA7 F3A4                           rep     movsb                   ; ds already set to cs

00000CA9 5F                             pop     di
00000CAA 07                             pop     es

00000CAB C3                             retn

                                ; ---------------------------------------------------------------------------
                                ; procedure : ClrVDISKHeader
                                ;
                                ;           Clears the first 32 bytes at 1MB boundary
                                ;           So that DOS/HIMEM is not confused about the VDISK header
                                ;           left by previous DOS=HIGH session
                                ;
                                ; ---------------------------------------------------------------------------

                               struc desc
00000000 ????                   .seg_lim:  resw   1             ; seg limit 64K
00000002 ????                   .lo_word:  resw   1             ; 24 bit seg physical
00000004 ??                     .hi_byte:  resb   1             ; address
00000005 ??                     .acc_rights:       resb   1             ; access rights ( CPL0 - R/W )
00000006 ????                   .reserved: resw   1             ;
                                .size:
                               endstruc

                                        ; 23/10/2022
                               bmove:           ;label byte

                               dummy:           ;times desc.size db 0 ; desc <>
00000CAC 00<rep 8h>                     times 8 db 0
                               gdt:     ;times desc.size db 0 ; desc <>
00000CB4 00<rep 8h>                     times 8 db 0
00000CBC FFFF                  src_desc:  dw     0FFFFh          ; desc <0ffffh,0,0,93h,0>
00000CBE 0000                           dw      0
00000CC0 00                             db      0
00000CC1 93                             db      93h
00000CC2 0000                           dw      0
00000CC4 FFFF                  tgt_desc:  dw     0FFFFh          ; desc <0ffffh,0,10h,93h,0>  ; 1MB
00000CC6 0000                           dw      0
00000CC8 10                             db      10h
00000CC9 93                             db      93h
00000CCA 0000                           dw      0

                               rombios_code:    ;times desc.size db 0 ; desc <>
00000CCC 00<rep 8h>                     times 8 db 0
                               temp_stack: ;times desc.size db 0 ; desc <>
00000CD4 00<rep 8h>                     times 8 db 0

00000CDC 00<rep 20h>           ClrdVDISKHead:   times 32 db 0          ; db 32 dup (0)


                               ; 25/03/2019 - Retro DOS v4.0 (MSDOS 6.21 IO.SYS, MSDOS 6.0 SYSINIT1.ASM)

                               ; 30/12/2022 - Retro DOS v4.2 (Modified MSDOS 6.21 IO.SYS)
                               ; (SYSINIT:0CA6h)

                               ClrVDISKHeader:  ; proc  near

                               ;; 04/07/2023 - Retro DOS v4.2 IO.SYS (optimized)
                               ;;-------------------------------------------------------         ;I070
                               ;; The following workaround get around a problem with the         ;I070
                               ;; Tortugas and PS/2 30-286 BIOS when password server mode        ;I070
                               ;; is set. On those machines the INT 15h block move code          ;I070
                               ;; goes through the 8042 to twiddle A20 instead of port 92h.      ;I070
                               ;; In password server mode the 8042 is disabled so the block      ;I070
                               ;; move crashes the system. We can do this because these          ;I070
                               ;; systems clear all of memory on a cold boot.                   ;I070
                               ;                                                       ;I070
                               ;               in      al,64h          ; Test for password servr mode ;I070
                               ;               test    al,10h          ; Is keyboard inhibited?       ;I070
                               ;               jnz     short ClrVDISKok ; No, go do block move.       ;I070
                               ;                                       ; Check for Tortugas...        ;I070
                               ;               cmp     word [cs:sys_model_byte],19F8h                 ;I070
                               ;               je      short ClrVDISKno                               ;I070
                               ;                                       ; Check for mod 30-286         ;I070
                               ;               cmp     word [cs:sys_model_byte],09FCh                 ;I070
                               ;               jne     short ClrVDISKok                       ;I070
                               ;ClrVDISKno:       retn                    ; Return w/o block move.            ;I070
                               ;                                                               ;I070
                               ;ClrVDISKok:                                                    ;I070
                               ;-------------------------------------------------------         ;I070

                                   ; 30/12/2023 - Retro DOS v5.0
                                   ; PCDOS 7.1 IBMBIO.COM - SYSINIT:0DBAh
                               ClrVDISKHeader:
00000CFC E464                      in      al,64h  ; 8042 keyboard controller status register
                                                   ; 7:  PERR    1=parity error in data received from keyboard
                                                   ;   +---------- AT Mode ----------+------------ PS/2 Mode ------------+
                                                   ; 6: |RxTO    receive (Rx) timeout  | TO     general timeout (Rx or Tx) |
                                                   ; 5: |TxTO    transmit (Tx) timeout | MOBF   mouse output buffer full   |
                                                   ;   +------------------------------+-----------------------------------+
                                                   ; 4:  INH     0=keyboard communications inhibited
                                                   ; 3:  A2      0=60h was the port last written to, 1=64h was last
                                                   ; 2:  SYS     distinguishes reset types: 0=cold reboot, 1=warm reboot
                                                   ; 1:  IBF     1=input buffer full (keyboard can't accept data)
                                                   ; 0:  OBF     1=output buffer full (data from keyboard is available)
00000CFE A810                      test    al,10h  ; test bit 4 - Is keyboard inhibited?
00000D00 7511                      jnz     short ClrVDISKok ; No, go do block move
                                   ; 30/12/2023
                                   ; ds = cs
00000D02 813E[BB02]F819            cmp     word [sys_model_byte],19F8h ; check for TORTUGA models
00000D08 7408                      jz      short ClrVDISKno ; do not use INT 15h block move code
                                                            ; (while 8042 is disabled)
00000D0A 813E[BB02]FC09            cmp     word [sys_model_byte],9FCh ; check for PS/2 30-286 model
00000D10 7501                      jnz     short ClrVDISKok
                               ClrVDISKno:
00000D12 C3                        retn
                               ; ---------------------------------------------------------------------------
                                   ; 30/12/2023
                               ClrVDISKok:
```

```
                                        ; 12/12/2022
                                        ; 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
                                
00000DD9 2EA1[AB02]                 mov     ax,[cs:DirStrng+2] ; "\",0
00000DDD AB                         stosw
00000DDE 2EFE06[A902]               inc     byte [cs:DirStrng]
00000DE3 31C0                       xor     ax,ax ; 0
00000DE5 51                         push    cx
                                    ;mov    cx,curdir_list.cdir_flags - 4 ; 63
00000DE6 B93F00                     mov     cx,63   ; 23/10/2022
00000DE9 F3AA                       rep     stosb                   ; zero out rest of CURDIR_TEXTs
                                
                                ;   should handle the system that does not have any floppies.
                                ;   in this case,we are going to pretended there are two dummy floppies
                                ;   in the system. still they have dpb and cds,but we are going to
                                ;   0 out curdir_flags,curdir_devptr of cds so ibmdos can issue
                                ;   "invalid drive specification" message when the user try to
                                ;   access them.
                                ;
                                ;   (ax) = 0
                                ;   (es:di) = CURDIR_FLAGS in the CDS records
                                ;   (ds:si) = Next DPB (-1 if none)
                                
00000DEB 83FEFF                     cmp     si,-1   ; cmp si,0FFFFh
00000DEE 740C                       je      short fooset_zero       ; don't have any physical drive.
                                
                                ;   check to see if we are faking floppy drives. if not go to normcds.
                                ;   if we are faking floppy drives then see if this cds being initialised
                                ;   is for drive a: or b: by checking the appropriate field in the dpb
                                ;   pointed to by ds:si. if not for a: or b: then go to normcds. if
                                ;   for a: or b: then execute the code given below starting at fooset_zero.
                                ;   for dpb offsets look at inc\dpb.inc.
                                
                                    ; 03/09/2023
00000DF0 41                         inc     cx  ; cx = 1
                                
00000DF1 2E380E[8B02]               cmp     [cs:fake_floppy_drv],cl ; 1 ; 03/09/2023
                                    ;cmp    byte [cs:fake_floppy_drv],1
00000DF6 750A                       jne     short normcds           ; machine has floppy drives
                                    ;cmp    byte [si+DPB.drive],1 ; if dpb_drive = 0 (a) or 1 (b).
                                    ;cmp    byte [si],1
00000DF8 380C                       cmp     [si],cl ; 1 ; 03/09/2023
00000DFA 7706                       ja      short normcds
                                
                                    ; 30/12/2023
                                    ; ax = 0
                                fooset_zero:
00000DFC B103                       mov     cl,3                    ; the next dbp pointer
                                                                    ; AX should be zero here
00000DFE F3AB                       rep     stosw
                                    ; 30/12/2023
                                    ;pop    cx
00000E00 EB0F                       jmp     short get_next_dpb ; findcds
                                
                                ;   (ax) = 0
                                
                                    ; 30/12/2023
                                ;fooset_zero:
                                    ;mov    cl,3
                                    ;rep    stosw
                                    ;pop    cx
                                    ;jmp    short fincds
                                
                                ;*  We have a "normal" DPB and thus a normal CDS.
                                ;
                                ;   (ax) = 0
                                ;   (es:di) = CURDIR_FLAGS in the CDS records
                                ;   (ds:si) = Next DPB (-1 if none)
                                
                                normcds:
                                    ; 30/12/2023
                                    ;pop    cx
                                
                                ;   if a non-fat based media is detected (by dpb.numberoffat == 0), then
                                ;   set curdir_flags to 0. this is for signaling ibmdos and ifsfunc that
                                ;   this media is a non-fat based one.
                                
                                    ;cmp    byte [si+DPB.FAT_COUNT],0 ; non fat system?
                                    ; 23/10/2022
                                    ;cmp    byte [si+8],0
                                    ; 03/09/2023 (ax=0)
00000E02 384408                     cmp     [si+8],al ; 0
00000E05 7403                       je      short setnormcds        ; yes. set curdir_flags to 0. ax = 0 now.
00000E07 B80040                     mov     ax,curdir_inuse ; 4000h         ; else,fat system. set the flag to curdir_inuse.
                                    ;mov    ax,4000h
                                setnormcds:
00000E0A AB                         stosw                           ; curdir_flags
00000E0B 89F0                       mov     ax,si
00000E0D AB                         stosw                           ; curdir_devptr
00000E0E 8CD8                       mov     ax,ds
00000E10 AB                         stosw
                                
                                get_next_dpb:                               ; entry point for fake_fooset_zero
                                    ; 30/12/2022
                                    ; (MSDOS 6.21 SYSINIT:0DD1h)
                                    ; 23/10/2022
                                    ;lds    si,[si+19h] ; (MSDOS 5.0 SYSINIT)
                                    ;;lds   si,[si+DPB.NEXT_DPB] ; [si+19h]
                                fincds:     ; get_next_dpb
                                    ; 30/12/2023
00000E11 59                         pop     cx
                                    ; 30/12/2022
                                    ; (MSDOS 6.21 SYSINIT:0DD1h)
00000E12 B8FFFF                     mov     ax,-1   ; mov ax,0FFFFh
00000E15 AB                         stosw                           ; curdir_id
00000E16 AB                         stosw                           ; curdir_id
00000E17 AB                         stosw                           ; curdir_user_word
00000E18 B80200                     mov     ax,2
00000E1B AB                         stosw                           ; curdir_end
00000E1C B000                       mov     al,0                    ; clear out 7 bytes (curdir_type,
00000E1E AA                         stosb
00000E1F AB                         stosw                           ;  curdir_ifs_hdr,curdir_fsda)
00000E20 AB                         stosw
00000E21 AB                         stosw
                                
00000E22 E2AD                       loop    fooset
                                
00000E24 2EC606[A902]41             mov     byte [cs:DirStrng],"A"; "A:\"
                                
00000E2A C3                         retn
                                
                                ; --------------------------------------------------------------------------
                                ;***get_dpb_for_drive_al -- lookup the DPB for drive in al
                                ;
                                ;   entry:
```

```
25408                                   ;       al == ASCII CAPS drive letter
25409                                   ;
25410                                   ;   exit:
25411                                   ;       ds:si -> DPB, or si = -1 if not found
25412                                   ; ----------------------------------------------------------------------
25413
25414                                   ; 30/12/2023
25415                                   ; PCDOS 7.1 IBMBIO.COM - SYSINIT:0EFEh
25416
25417                                       ; 30/12/2022
25418                                       ; (MSDOS 6.21 SYSINIT:0DEAh)
25419                                       ; 23/10/2022
25420                                   get_dpb_for_drive_al:
25421 00000E2B 2EC536[6D02]               lds     si,[cs:DOSINFO]             ; point to first DPB
25422                                     ;lds    si,[si+SYSI_DPB]      ; (ds:si) = address of first DPB
25423 00000E30 C534                       lds     si,[si]
25424 00000E32 2C41                       sub     al,'A'
25425
25426                                   get_dpb_for_drive_1:
25427                                       ;cmp    al,[si+DPB.DRIVE]     ; match?
25428 00000E34 3A04                       cmp     al,[si]
25429 00000E36 7408                       je      short got_dpb_for_drive       ;  done if so
25430
25431 00000E38 C57419                     lds     si,[si+DPB.NEXT_DPB] ; [si+19h]
25432 00000E3B 83FEFF                     cmp     si,-1
25433 00000E3E 75F4                       jne     short get_dpb_for_drive_1 ; loop until hit end of DPBs
25434
25435                                   got_dpb_for_drive:
25436 00000E40 C3                         retn
25437
25438                                   ;============================================================================
25439
25440                                   ;** EndFile - Build DOS structures
25441                                   ;
25442                                   ; This procedure is called after the config.sys has been processed and
25443                                   ; installable device drivers have been loaded (but before "install="
25444                                   ; programs are loaded) to create the dos structures such as SFTs, buffers,
25445                                   ; FCBs, CDSs, etc. It also loads the sysinit_base module in low memory
25446                                   ; to allow for the safe EXECing of "install=" programs. All memory
25447                                   ; above these structures is deallocated back to DOS.
25448                                   ;
25449                                   ;   ENTRY   ?? BUGBUG
25450                                   ;   EXIT    ?? BUGBUG
25451                                   ;   USES    ?? BUGBUG
25452
25453                                   ;============================================================================
25454                                   ; allocate files
25455                                   ; ----------------------------------------------------------------------
25456
25457                                       ; 23/10/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 IO.SYS)
25458                                       ; (SYSINIT:0CCDh)
25459
25460                                       ; 30/12/2022 - Retro DOS v4.2 (Modified MSDOS 6.21 IO.SYS)
25461                                       ; (SYSINIT:0E00h)
25462
25463                                       ; 09/04/2024 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMBIO.COM)
25464                                       ; (PCDOS 7.1 IBMBIO.COM - SYSINIT:0F14h)
25465
25466                                       ; ((MSDOS 6.22 IO.SYS - SYSINIT:0E00h))
25467
25468                                   endfile:
25469                                   ; we are now setting up final cdss,buffers,files,fcss strings etc. we no
25470                                   ; longer need the space taken by the temp stuff below confbot,so set alloclim
25471                                   ; to confbot.
25472
25473                                   ;   if this procedure has been called to take care of install= command,
25474                                   ;      then we have to save es,si registers.
25475
25476                                       ; 11/12/2022
25477                                       ; ds = cs
25478
25479                                       ; 23/10/2022
25480                                       ; 31/03/2019
25481 00000E41 1E                         push    ds
25482
25483                                       ;;mov   ax,Bios_Data ; 0070h
25484                                       ;mov    ax,KERNEL_SEGMENT ; 0070h
25485                                       ; 21/10/2022
25486 00000E42 B87000                     mov     ax,DOSBIODATASEG ; 0070h
25487 00000E45 8ED8                       mov     ds,ax
25488
25489                                       ;cmp    word [052Fh],0
25490 00000E47 833E[A004]00               cmp     word [multrk_flag],multrk_off1 ;=0,multrack= command entered?
25491 00000E4C 7505                       jne     short multrk_flag_done
25492                                     ; 02/11/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
25493                                     ;or     word [multrk_flag],multrk_on ; 80h  ; default will be on.
25494                                     ; 12/12/2022
25495 00000E4E 800E[A004]80               or      byte [multrk_flag],multrk_on ; 80h
25496                                   multrk_flag_done:
25497                                       ; 23/10/2022
25498                                       ; 31/03/2019
25499 00000E53 1F                         pop     ds
25500
25501                                       ; 11/12/2022
25502                                       ; ds = cs
25503                                       ;mov    ax,[top_of_cdss] ; mov ax,[CONFBOT]
25504                                       ; 31/12/2022 - Retro DOS v4.2 (Modified MSDOS 6.21 IO.SYS)
25505                                       ; (SYSINIT:0E14h)
25506 00000E54 A1[A302]                   mov     ax,[CONFBOT]
25507 00000E57 A3[A502]                   mov     [ALLOCLIM],ax
25508                                     ; 23/10/2022
25509                                     ;mov    ax, [cs:top_of_cdss]
25510                                     ;mov    [cs:ALLOCLIM], ax
25511
25512                                       ; 11/12/2022
25513                                       ; ds = cs
25514                                       ;push   cs
25515                                       ;pop    ds
25516
25517                                       ;mov    ax,[CONFBOT]
25518                                       ;mov    [ALLOCLIM],ax
25519
25520                                       ; 09/04/2024 - Retro DOS v5.0 (PCDOS 7.1 IBMBIO.COM)
25521                                       ;;;
25522                                       ;;mov   ax,[cs:ALLOCLIM]
25523                                       ;mov    ax,[ALLOCLIM]
25524                                       ;mov    [cs:prev_alloclim],ax
25525 00000E5A A3[6C03]                   mov     [prev_alloclim],ax
25526                                       ;mov    ax,[cs:memhi]
25527 00000E5D A1[6403]                   mov     ax,[memhi]
25528                                       ;mov    [cs:prev_memhi],ax
25529 00000E60 A3[6A03]                   mov     [prev_memhi],ax
25530                                   dosfts:
25531                                       ;;;
```

```
25532
25533 00000E63 E88C39                    call    round
25534
25535                                    ; 11/12/2022
25536                                    ; ds = cs
25537 00000E66 A0[9F02]                  mov     al,[FILES]
25538                                    ; 23/10/2022
25539                                    ;mov    al,[cs:FILES]
25540 00000E69 2C05                      sub     al,5
25541 00000E6B 764B                      jbe     short dofcbs
25542
25543 00000E6D 50                        push    ax
25544                                    ;mov    al,devmark_files ; 'F'
25545 00000E6E B046                      mov     al,'F'
25546 00000E70 E81808                    call    setdevmark              ; set devmark for sfts (files)
25547 00000E73 58                        pop     ax
25548 00000E74 30E4                      xor     ah,ah                   ; do not use cbw instruction!!!!!
25549                                                                    ;  it does sign extend.
25550                                    ; 11/12/2022
25551                                    ; ds = cs
25552 00000E76 8B1E[6203]                mov     bx,[memlo]
25553 00000E7A 8B16[6403]                mov     dx,[memhi]
25554 00000E7E C53E[6D02]                lds     di,[DOSINFO]            ;get pointer to dos data
25555                                    ; 23/10/2022
25556                                    ;mov    bx,[cs:memlo]
25557                                    ;mov    dx,[cs:memhi]
25558                                    ;lds    di,[cs:DOSINFO]
25559
25560                                    ;lds    di,[di+SYSI_SFT]        ;ds:bp points to sft
25561 00000E82 C57D04                    lds     di,[di+4]
25562
25563                                    ;mov    [di+SF.SFLink],bx
25564 00000E85 891D                      mov     [di],bx
25565 00000E87 895502                    mov     [di+SF.SFLink+2],dx     ;set pointer to new sft
25566
25567 00000E8A 0E                        push    cs
25568 00000E8B 1F                        pop     ds
25569
25570                                    ; 11/12/2022
25571                                    ; ds = cs
25572 00000E8C C43E[6203]                les     di,[memlo]              ;point to new sft
25573                                    ; 23/10/2022
25574                                    ;les    di,[cs:memlo]
25575
25576                                    ;mov    word [es:di+SF.SFLink],-1
25577 00000E90 26C705FFFF                mov     word [es:di],-1                 ; 0FFFFh
25578                                    ;mov    [es:di+SF.SFCount],ax
25579 00000E95 26894504                  mov     [es:di+4],ax
25580                                    ; 09/04/2024
25581 00000E99 B33B                      mov     bl,SF_ENTRY.size ; 59
25582                                    ;mov    bl,59
25583 00000E9B F6E3                      mul     bl                      ;ax = number of bytes to clear
25584 00000E9D 89C1                      mov     cx,ax
25585                                    ; 11/12/2022
25586                                    ; ds = cs
25587 00000E9F 0106[6203]                add     [memlo],ax              ;allocate memory
25588                                    ; 23/10/2022
25589                                    ;add    [cs:memlo],ax
25590 00000EA3 B80600                    mov     ax,6
25591                                    ; 11/12/2022
25592 00000EA6 0106[6203]                add     [memlo],ax              ;remember the header too
25593                                    ;add    [cs:memlo],ax
25594                                    ; 11/12/2022
25595 00000EAA 800E[6919]02              or      byte [setdevmarkflag],for_devmark ; 2
25596                                    ; 23/10/2022
25597                                    ;or     byte [cs:setdevmarkflag],2
25598 00000EAF E84039                    call    round                   ; check for mem error before the stosb
25599 00000EB2 01C7                      add     di,ax
25600 00000EB4 31C0                      xor     ax,ax
25601 00000EB6 F3AA                      rep     stosb                   ;clean out the stuff
25602
25603                                  ; allocate fcbs
25604                                  ; ---------------------------------------------------------------------------
25605
25606                                      ; 23/10/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 IO.SYS)
25607                                      ; (SYSINIT:0D48h)
25608                                  dofcbs:
25609                                      ; 11/12/2022
25610                                      ; ds = cs
25611                                      ;push   cs
25612                                      ;pop    ds
25613 00000EB8 E83739                    call    round
25614                                    ;mov    al,devmark_fcbs         ; 'X'   ;='x'
25615 00000EBB B058                      mov     al,'X'
25616 00000EBD E8CB07                    call    setdevmark
25617                                    ; 11/12/2022
25618                                    ; ds = cs
25619 00000EC0 A0[A002]                  mov     al,[FCBS]
25620                                    ;mov    al,[cs:FCBS]
25621 00000EC3 30E4                      xor     ah,ah                   ; do not use cbw instruction!!!!!
25622                                                                    ;  it does sign extend.
25623                                    ; 11/12/2022
25624 00000EC5 8B1E[6203]                mov     bx,[memlo]
25625 00000EC9 8B16[6403]                mov     dx,[memhi]
25626 00000ECD C53E[6D02]                lds     di,[DOSINFO]            ;get pointer to dos data
25627                                    ; 23/10/2022
25628                                    ;mov    bx,[cs:memlo]
25629                                    ;mov    dx,[cs:memhi]
25630                                    ;lds    di,[cs:DOSINFO]
25631
25632                                    ;mov    [di+SYSI_FCB],bx
25633                                    ;mov    [di+SYSI_FCB+2],dx ;set pointer to new table
25634                                    ; 23/10/2022
25635 00000ED1 895D1A                    mov     [di+1Ah],bx             ; [di+SYSI_FCB]
25636 00000ED4 89551C                    mov     [di+1Ch],dx             ; [di+SYSI_FCB+2]
25637
25638 00000ED7 2E8A1E[A102]               mov     bl,[cs:KEEP]
25639 00000EDC 30FF                      xor     bh,bh
25640                                    ;mov    [di+SYSI_KEEP],bx
25641 00000EDE 895D1E                    mov     [di+1Eh],bx             ; [di+SYSI_KEEP]
25642
25643 00000EE1 0E                        push    cs
25644 00000EE2 1F                        pop     ds
25645
25646 00000EE3 C43E[6203]                les     di,[memlo]              ;point to new table
25647                                    ;mov    word [es:di+SF.SFLink],-1
25648 00000EE7 26C705FFFF                mov     word [es:di],-1
25649                                    ;mov    [es:di+SF.SFCount],ax
25650                                    ; 02/11/2022
25651 00000EEC 26894504                  mov     [es:di+4],ax
25652 00000EF0 B33B                      mov     bl,SF_ENTRY.size ; 59
25653 00000EF2 89C1                      mov     cx,ax
25654 00000EF4 F6E3                      mul     bl                      ;ax = number of bytes to clear
25655 00000EF6 0106[6203]                add     [memlo],ax              ;allocate memory
```

```
25656                                   ;mov    ax,6
25657 00000EFA B80600                   mov     ax,SF.size-2 ; 6
25658 00000EFD 0106[6203]               add     [memlo],ax              ;remember the header too
25659                                   ;or     byte [setdevmarkflag],for_devmark ; 2
25660 00000F01 800E[6919]02             or      byte [setdevmarkflag],2
25661 00000F06 E8E938                   call    round                   ; check for mem error before the stosb
25662 00000F09 01C7                     add     di,ax                   ;skip over header
25663 00000F0B B041                     mov     al,'A'
25664                                 fillloop:
25665 00000F0D 51                       push    cx                      ; save count
25666 00000F0E B93B00                   mov     cx,SF_ENTRY.size ; 59 ; number of bytes to fill
25667 00000F11 FC                       cld
25668 00000F12 F3AA                     rep     stosb                   ; filled
25669
25670                                   ;mov    word [es:di-(SF_ENTRY.size)+SF_ENTRY.sf_ref_count],0  ; [es:di-59]
25671                                   ;mov    word [es:di-(SF_ENTRY.size)+SF_ENTRY.sf_position],0   ; [es:di-38]
25672                                   ;mov    word [es:di-(SF_ENTRY.size)+SF_ENTRY.sf_position+2],0 ; [es:di-36]
25673
25674                                   ; 18/12/2022
25675                                   ;cx = 0
25676 00000F14 26894DC5                mov     [es:di-(SF_ENTRY.size)+SF_ENTRY.sf_ref_count],cx ;0  ; [es:di-59]
25677 00000F18 26894DDA                mov     [es:di-(SF_ENTRY.size)+SF_ENTRY.sf_position],cx ;0   ; [es:di-38]
25678 00000F1C 26894DDC                mov     [es:di-(SF_ENTRY.size)+SF_ENTRY.sf_position+2],cx ;0 ; [es:di-36]
25679
25680                                   ; 23/10/2022
25681                                   ;mov     word [es:di-3Bh],0
25682                                   ;mov     word [es:di-26h],0
25683                                   ;mov     word [es:di-24h],0
25684
25685 00000F20 59                       pop     cx
25686 00000F21 E2EA                     loop    fillloop
25687
25688                                 ; allocate buffers
25689                                 ; ---------------------------------------------------------------------------
25690
25691                                 ; search through the list of media supported and allocate 3 buffers if the
25692                                 ; capacity of the drive is > 360kb
25693
25694                                   ; 18/12/2022
25695                                   ; cx = 0
25696 00000F23 833E[9902]FF             cmp     word [buffers],-1     ; has buffers been already set?
25697 00000F28 7403                     je      short dodefaultbuff
25698 00000F2A E98000                   jmp     dobuff                ; the user entered the buffers=.
25699
25700                                 dodefaultbuff:
25701                                   ; 18/12/2022
25702 00000F2D 890E[9B02]               mov     [h_buffers],cx ; 0
25703                                   ;inc    cx
25704                                   ;inc    cx
25705                                   ;mov    [buffers],cx ; 2
25706                                   ; 10/04/2024
25707 00000F31 C706[9902]0200           mov     word [buffers],2
25708
25709                                   ;mov    word [h_buffers],0    ; default is no heuristic buffers.
25710                                   ;mov    word [buffers],2      ; default to 2 buffers
25711
25712                                   ; 23/10/2022
25713                                   ; 04/09/2023
25714                                   ;push   ax
25715                                   ;push   ds ; 26/03/2019
25716
25717                                   ; 04/09/2023
25718                                   ; ds = cs
25719 00000F37 C42E[6D02]               les     bp,[DOSINFO]          ; search through the dpb's
25720                                   ;les    bp,[cs:DOSINFO]
25721                                   ;les    bp,[es:bp+SYSI_DPB]   ; get first dpb
25722                                   ; 11/12/2022
25723 00000F3B 26C46E00                 les     bp,[es:bp]
25724                                   ; 23/10/2022
25725                                   ;les    bp,[es:bp+0]   ; ! (MSDOS 5.0 IO.SYS address compability) !
25726
25727                                   ; 04/09/2023
25728                                   ; ds = cs
25729                                   ;push   cs
25730                                   ;pop    ds
25731                                 ;SYSINIT:0DE2h:
25732                                 nextdpb:                          ; test if the drive supports removeable media
25733                                   ;mov    bl,[es:bp+DPB.drive]
25734                                   ; 11/12/2022
25735 00000F3F 268A5E00                 mov     bl,[es:bp]
25736                                   ; 23/10/2022
25737                                   ;mov    bl,[es:bp+0]   ; ! (MSDOS 5.0 IO.SYS address compability) !
25738
25739                                   ;inc    bl
25740                                   ; 18/12/2022
25741 00000F43 43                       inc     bx
25742
25743                                   ;mov    ax,(IOCTL<<8)|8
25744 00000F44 B80844                   mov     ax,4408h
25745 00000F47 CD21                     int     21h             ; DOS - 2+ - IOCTL -
25746
25747                                 ; ignore fixed disks
25748
25749 00000F49 09C0                     or      ax,ax                   ; ax is nonzero if disk is nonremoveable
25750 00000F4B 7534                     jnz     short nosetbuf
25751
25752                                 ; get parameters of drive
25753
25754 00000F4D 31DB                     xor     bx,bx
25755                                   ;;mov   bl,[es:bp+DPB.drive]
25756                                   ; 11/12/2022
25757 00000F4F 268A5E00                 mov     bl,[es:bp]
25758                                   ; 23/10/2022
25759                                   ;mov    bl,[es:bp+0]   ; ! (MSDOS 5.0 IO.SYS address compability) !
25760
25761                                   ;inc    bl
25762                                   ; 18/12/2022
25763 00000F53 43                       inc     bx
25764
25765 00000F54 BA[BE4D]                 mov     dx,deviceparameters
25766                                   ;mov    ax,(IOCTL<<8)|GENERIC_IOCTL
25767 00000F57 B80D44                   mov     ax,440Dh
25768                                   ;mov    cx,(RAWIO<<8)|GET_DEVICE_PARAMETERS
25769 00000F5A B96008                   mov     cx,860h
25770 00000F5D CD21                     int     21h             ; DOS - 2+ - IOCTL -
25771 00000F5F 7220                     jc      short nosetbuf          ; get next dpb if driver doesn't support
25772                                                                   ; generic ioctl
25773                                 ; determine capacity of drive
25774                                 ; media capacity = #sectors * bytes/sector
25775
25776                                   ;mov    bx,[deviceparameters+A_DEVICEPARAMETERS.DP_BPB+A_BPB.TOTALSECTORS]
25777                                   ; 23/10/2022
25778 00000F61 8B1E[CD4D]               mov     bx,[deviceparameters+15] ; total sectors (16 bit)
25779
```

```
                                ; to keep the magnitude of the media capacity within a word,
                                ; 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
```

```
                                    push    ds                      ; save buffer info. ptr.
                                    push    bx
                                    call    set_buffer
                                    pop     bx
                                    pop     ds
                                ;now set the secondary buffer if specified.
                                    cmp     word [cs:h_buffers],0
                                    je      short xif16
                                    call    round
                                    ; 23/10/2022
                                    mov     cx,[cs:memlo]
                                    ;mov    [bx+BUFFINF.Cache_ptr],cx  ; [bx+6]
                                    mov     [bx+6],cx
                                    mov     cx,[cs:memhi]
                                    ;mov    [bx+BUFFINF.Cache_ptr+2],cx ; [bx+8]
                                    mov     [bx+8],cx
                                    mov     cx,[cs:h_buffers]
                                    ;mov    [bx+BUFFINF.Cache_count],cx ; [bx+10]
                                    mov     [bx+10],cx
                                    mov     ax,512                  ; 512 byte
                                    mul     cx
                                    mov     [cs:memlo],ax
                                    ;or     byte [cs:setdevmarkflag],for_devmark ; 2
                                    or      byte [cs:setdevmarkflag],2
                                    call    round
                                xif16:
                                ; ------------------------------------------------------------------------
                                ; allocate cdss
                                ; ------------------------------------------------------------------------
                                buf1:
                                    call    round
                                    push    ax
                                    ; 23/10/2022
                                    ;mov    ax,devmark_cds          ;='L'
                                    mov     ax, 'L'
                                    call    setdevmark
                                    pop     ax
                                    les     di,[cs:DOSINFO]
                                    ;mov    cl,[es:di+SYSI_NUMIO]
                                    mov     cl,[es:di+20h]
                                    cmp     cl,[cs:NUM_CDS]
                                    jae     short gotncds           ; user setting must be at least numio
                                    mov     cl,[cs:NUM_CDS]
                                gotncds:
                                    xor     ch,ch
                                    ;mov    [es:di+SYSI_NCDS],cl  ; [es:di+33]
                                    mov     [es:di+21h],cl
                                    mov     ax,[cs:memhi]
                                    ;mov    [es:di+SYSI_CDS+2],ax
                                    mov     [es:di+18h],ax
                                    mov     ax,[cs:memlo]
                                    ;mov    [es:di+SYSI_CDS],ax
                                    mov     [es:di+16h],ax
                                    mov     al,cl
                                    ;mov    ah,curdirlen ; curdir_list.size
                                    mov     ah,88
                                    mul     ah
                                    call    ParaRound
                                    add     [cs:memhi],ax

                                    ;or     byte [cs:setdevmarkflag],for_devmark ; 2
                                    or      byte [cs:setdevmarkflag],2
                                    call    round                   ; check for mem error before initializing
                                    ;lds    si,[es:di+SYSI_DPB] ; [es:di+0]
                                    lds     si,[es:di]
                                    ;les    di,[es:di+SYSI_CDS] ; [es:di+22]
                                    les     di,[es:di+16h]
                                    call    fooset
                                ; ------------------------------------------------------------------------
                                ; allocate space for internal stack
                                ; ------------------------------------------------------------------------
                                    push    cs
                                    pop     ds
                                ;   if the user did not entered stacks= command, as a default, do not install
                                ;   sytem stacks for pc1,pc xt,pc portable cases.
                                ;   otherwise,install it to the user specified value or to the default
                                ;   value of 9,128 for other systems.
                                    cmp     word [stack_addr],-1 ; has the user entered "stacks=" command?
                                    je      short doinstallstack  ; then install as specified by the user
                                    cmp     byte [sys_scnd_model_byte],0 ; pc1,xt has the secondary model byte = 0
                                    jne     short doinstallstack  ; other model should have default stack of 9,128
                                    cmp     byte [sys_model_byte],0FEh ; pc1, pc/xt or pc portable ?
                                    jae     short skipstack
                                doinstallstack:
                                    mov     ax,[stack_count]      ; stack_count = 0?
                                    or      ax,ax                 ; then, stack size must be 0 too.
                                    jz      short skipstack               ; don't install stack.

                                ;   dynamic relocation of stack code.

                                    call    round                   ;[memhi] = seg. for stack code
                                                                    ;[memlo] = 0

                                ; set devmark block into memory for mem command
                                ; devmark_id = 's' for stack

                                    ;mov    al,devmark_stk ;='S'
                                    ; 23/10/2022
                                    mov     al,'S'
                                    call    setdevmark

                                    mov     ax,[memhi]
                                    mov     es,ax           ;es -> seg. the stack code is going to move.
                                    ; 02/11/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
                                    ; 11/12/2022
                                    ; ds = cs
                                    ;push   cs
                                    ;pop    ds
                                    xor     si,si           ;!!we know that stack code is at the beginning of sysinit.
                                    xor     di,di
                                    mov     cx,endstackcode
```

```
26028 0000108E 890E[6203]                mov     [memlo],cx
26029 00001092 E85D37                    call    round           ;have enough space for relocation?
26030 00001095 F3A4                      rep     movsb
26031
26032 00001097 1E                        push    ds              ; stick the location of the NextStack entry
26033                                    ;;mov   ax,Bios_Data    ; into the Win386 Instance Data tables
26034                                    ;mov    ax,KERNEL_SEGMENT ; 70h
26035                                    ; 21/10/2022
26036 00001098 B87000                    mov     ax,DOSBIODATASEG ; 0070h
26037 0000109B 8ED8                      mov     ds,ax
26038 0000109D C706[0208][1000]          mov     word [NextStack],nextentry ; (8C0h for MSDOS 6.21 IO.SYS)
26039 000010A3 8C06[0408]                mov     [NextStack+2],es          ; (8C2h for MSDOS 6.21 IO.SYS)
26040
26041 000010A7 2EA1[6203]                mov     ax,[cs:memlo]
26042 000010AB 2EA3[9002]                mov     [cs:stack_addr],ax ;set for stack area initialization
26043 000010AF A3[0808]                  mov     [IT_StackLoc],ax ; pass it as Instance Data, too
26044 000010B2 2EA1[6403]                mov     ax,[cs:memhi]  ;this will be used by stack_init routine.
26045 000010B6 2EA3[9202]                mov     [cs:stack_addr+2],ax
26046 000010BA A3[0A08]                  mov     [IT_StackLoc+2],ax
26047
26048                                ;   space for internal stack area = stack_count(entrysize + stack_size)
26049
26050                                    ;mov    ax,entrysize ; mov ax,8
26051                                    ; 23/10/2022
26052 000010BD B80800                    mov     ax,8
26053 000010C0 2E0306[8E02]              add     ax,[cs:stack_size]
26054 000010C5 2EF726[8C02]              mul     word [cs:stack_count]
26055
26056 000010CA A3[0C08]                  mov     [IT_StackSize],ax ; pass through to Instance Tables
26057
26058 000010CD 1F                        pop     ds              ; no more need to access Instance Table
26059
26060 000010CE E84402                    call    ParaRound       ; convert size to paragraphs
26061
26062                                    ; 11/12/2022
26063                                    ; ds = cs
26064                                    ;add    [cs:memhi],ax
26065 000010D1 0106[6403]                add     [memhi],ax
26066                                    ;or     byte [cs:setdevmarkflag],for_devmark ; 2
26067                                    ;or     byte [cs:setdevmarkflag],2
26068 000010D5 800E[6919]02              or      byte [setdevmarkflag],2
26069                                    ;or     byte [setdevmarkflag],for_devmark ; 2
26070                                                            ;to set the devmark_size for stack by round routine.
26071 000010DA E81537                    call    round           ; check for memory error before
26072                                                            ; continuing
26073 000010DD E87D03                    call    stackinit       ; initialize hardware stack.
26074                                                            ; cs=ds=sysinitseg,es=relocated stack code & data
26075                                skipstack:
26076
26077                                    ; 10/04/2024 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMBIO.COM)
26078                                    ; (PCDOS 7.1 IBMBIO.COM - SYSINIT:11F0h)
26079                                    ;;;
26080                                    ;push   cs
26081                                    ;pop    ds
26082                                    ; ds = cs
26083 000010E0 803E[6E03]01              cmp     byte [dosdata_umb],1  ; PCDOS 7 feature - DOSDATA=UMB/NOUMB configuration
26084                                                                  ; 1 = DOSDATA=UMB, 2 = (UMB) done, 0 = NOUMB
26085 000010E5 7773                      ja      short dosdata_umb_done; 2 - done
26086 000010E7 727D                      jb      short dosdata_noumb   ; 0 - DOSDATA=NOUMB
26087
26088 000010E9 803E[8B16]EA              cmp     byte [setdevmark],0EAh
26089 000010EE 7476                      je      short dosdata_noumb
26090
26091 000010F0 B80258                    mov     ax,5802h
26092 000010F3 CD21                      int     21h             ; DOS - 3+ - GET/SET MEMORY ALLOCATION STRATEGY
26093                                                            ; AL = function code: (DOS 5beta) get UMB link state
26094 000010F5 98                        cbw
26095 000010F6 89C7                      mov     di,ax                   ; al = 01h -> UMBs in DOS memory chain
26096                                                                    ; save current (previous) UMB link state
26097 000010F8 BB0100                    mov     bx,1                    ; bx = 01h -> add UMBs to DOS memory chain
26098
26099 000010FB B80358                    mov     ax,5803h
26100 000010FE CD21                      int     21h
26101 00001100 7264                      jc      short dosdata_noumb
26102
26103 00001102 B80058                    mov     ax,5800h
26104 00001105 CD21                      int     21h             ; DOS - 3+ - GET/SET MEMORY ALLOCATION STRATEGY
26105                                                            ; AL = function code: get allocation strategy
26106
26107 00001107 89C6                      mov     si,ax                   ; ax = current strategy
26108                                                                    ; save current (previous) allocation strategy
26109 00001109 BB4000                    mov     bx,40h                  ; bl = new strategy = 40h - high memory first fit
26110
26111 0000110C B80158                    mov     ax,5801h
26112 0000110F CD21                      int     21h
26113
26114 00001111 8B1E[6403]                mov     bx,[memhi]
26115 00001115 2B1E[6A03]                sub     bx,[prev_memhi]
26116
26117 00001119 B448                      mov     ah,48h
26118 0000111B CD21                      int     21h             ; DOS - 2+ - ALLOCATE MEMORY
26119                                                            ; BX = number of 16-byte paragraphs desired
26120 0000111D 89C1                      mov     cx,ax                   ; ax = segment of allocated block
26121 0000111F 89FB                      mov     bx,di                   ; restore previous UMB link state
26122
26123 00001121 B80358                    mov     ax,5803h
26124 00001124 CD21                      int     21h             ; DOS - 3+ - GET/SET MEMORY ALLOCATION STRATEGY
26125                                                            ; AL = function code: (DOS 5beta) set UMB link state
26126 00001126 89F3                      mov     bx,si                   ; restore previous allocation strategy
26127
26128 00001128 B80158                    mov     ax,5801h
26129 0000112B CD21                      int     21h             ; DOS - 3+ - GET/SET MEMORY ALLOCATION STRATEGY
26130                                                            ; AL = function code: set allocation strategy
26131 0000112D 81F900A0                  cmp     cx,0A000h               ; Is the allocated memory block (segment) a UMB?
26132 00001131 7233                      jb      short dosdata_noumb   ; no
26133
26134                                    ;mov    word [ALLOCLIM],0FFFFh
26135                                    ;mov    word [memlo],0
26136 00001133 890E[6403]                mov     [memhi],cx
26137 00001137 49                        dec     cx
26138 00001138 8EC1                      mov     es,cx                   ; point to arena/mcb
26139                                    ; 10/04/2024
26140 0000113A 31C9                      xor     cx,cx ; 0
26141 0000113C 890E[6203]                mov     [memlo],cx ; 0
26142 00001140 49                        dec     cx
26143 00001141 890E[A502]                mov     [ALLOCLIM],cx ; 0FFFFh
26144
26145 00001145 26C70601000800            mov     word [es:1],8           ; [es:arena_owner], 8 ; set impossible owner
26146 0000114C 26C70608005344            mov     word [es:8],4453h       ; [es:arena_name],'SD' ; System Data
26147 00001153 FE06[6E03]                inc     byte [dosdata_umb]      ; 1 -> 2 ; DOSDATA=UMB done.
26148 00001157 E909FD                    jmp     dosfts
26149
26150                              dosdata_umb_done:
26151 0000115A A1[6A03]                  mov     ax,[prev_memhi]               ; (recent memory block/segment before UMBs)
```

```
26152 0000115D A3[6403]                 mov     [memhi],ax
26153 00001160 A1[6C03]                 mov     ax,[prev_alloclim]
26154 00001163 A3[A502]                 mov     [ALLOCLIM],ax
26155                                   dosdata_noumb:
26156                                   ;;;
26157
26158                                   ;skipstack:
26159                                   ; 24/10/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 IO.SYS)
26160                                   ; (SYSINIT:0F99h)
26161
26162                                   ; 11/12/2022
26163                                   ; ds = cs
26164                                   ;push  cs
26165                                   ;pop   ds
26166
26167 00001166 A0[9F02]                 mov     al,[FILES]
26168 00001169 30E4                     xor     ah,ah           ; do not use cbw instruction!!!!!
26169                                                           ;  it does sign extend.
26170 0000116B 89C1                     mov     cx,ax
26171 0000116D 31DB                     xor     bx,bx           ;close standard input
26172 0000116F B43E                     mov     ah,3Eh ; CLOSE
26173 00001171 CD21                     int     21h
26174 00001173 BB0200                   mov     bx,2
26175                                   rcclloop:               ;close everybody but standard output
26176 00001176 B43E                     mov     ah,3Eh ; CLOSE ; need output so we can print message
26177 00001178 CD21                     int     21h             ; in case we can't get new one open.
26178 0000117A 43                       inc     bx
26179 0000117B E2F9                     loop    rcclloop
26180
26181 0000117D BA[C54A]                 mov     dx,condev
26182 00001180 B002                     mov     al,2
26183 00001182 B43D                     mov     ah,3Dh ; OPEN  ;open con for read/write
26184 00001184 F9                       stc                     ; set for possible int 24
26185 00001185 CD21                     int     21h
26186 00001187 7305                     jnc     short goaux
26187 00001189 E89C38                   call    badfil
26188 0000118C EB13                     jmp     short goaux2
26189                                   goaux:
26190 0000118E 50                           push    ax
26191 0000118F BB0100                       mov     bx,1            ;close standard output
26192 00001192 B43E                         mov     ah,3Eh ; CLOSE
26193 00001194 CD21                         int     21h
26194 00001196 58                           pop     ax
26195
26196 00001197 89C3                         mov     bx,ax           ;new device handle
26197 00001199 B445                         mov     ah,45h ; XDUP
26198 0000119B CD21                         int     21h             ;dup to 1,stdout
26199 0000119D B445                         mov     ah,45h ; XDUP
26200 0000119F CD21                         int     21h             ;dup to 2,stderr
26201                                   goaux2:
26202 000011A1 BA[C94A]                     mov     dx,auxdev
26203 000011A4 B002                         mov     al,2            ;read/write access
26204 000011A6 E8B038                       call    open_dev
26205
26206 000011A9 BA[CD4A]                     mov     dx,prndev
26207 000011AC B001                         mov     al,1            ;write only
26208 000011AE E8A838                       call    open_dev
26209
26210                                   ;global rearm command for shared interrupt devices attached in the system;
26211                                   ;shared interrupt attachment has some problem when it issues interrupt
26212                                   ;during a warm reboot. once the interrupt is presented by the attachment,
26213                                   ;no further interrupts on that level will be presented until a global rearm
26214                                   ;is issued. by the request of the system architecture group, msbio will
26215                                   ;issue a global rearm after every device driver is loaded.
26216                                   ;to issue a global rearm: ;for pc1,xt,palace
26217                                   ;
26218                                   ;                   out 02f2h,xx  ; interrupt level 2
26219                                   ;                   out 02f3h,xx  ; interrupt level 3
26220                                   ;                   out 02f4h,xx  ; interrupt level 4
26221                                   ;                   out 02f5h,xx  ; interrupt level 5
26222                                   ;                   out 02f6h,xx  ; interrupt level 6
26223                                   ;                   out 02f7h,xx  ; interrupt level 7
26224                                   ;
26225                                   ;   for pc at,in addition to the above commands,
26226                                   ;   need to handle the secondary interrupt handler
26227                                   ;
26228                                   ;                   out 06f2h,xx  ; interrupt level 10
26229                                   ;                   out 06f3h,xx  ; interrupt level 11
26230                                   ;                   out 06f4h,xx  ; interrupt level 12
26231                                   ;                   out 06f6h,xx  ; interrupt level 14
26232                                   ;                   out 06f7h,xx  ; interrupt level 15
26233                                   ;
26234                                   ;   for round-up machine
26235                                   ;
26236                                   ;                   none.
26237
26238                                   ; where xx stands for any value.
26239                                   ;
26240                                   ; for your information,after naples level machine,the system service bios
26241                                   ; call (int 15h),function ah=0c0h returns the system configuration parameters
26242
26243                                       ; 24/10/2022
26244
26245 000011B1 50                         push    ax
26246 000011B2 53                         push    bx
26247 000011B3 52                         push    dx
26248 000011B4 06                         push    es
26249
26250 000011B5 B0FF                       mov     al,0FFh             ;reset h/w by writing to port
26251 000011B7 BAF202                     mov     dx,2F2h             ;get starting address
26252 000011BA EE                         out     dx,al               ; out 02f2h,0ffh
26253 000011BB 42                         inc     dx
26254 000011BC EE                         out     dx,al               ; out 02f3h,0ffh
26255 000011BD 42                         inc     dx
26256 000011BE EE                         out     dx,al               ; out 02f4h,0ffh
26257 000011BF 42                         inc     dx
26258 000011C0 EE                         out     dx,al               ; out 02f5h,0ffh
26259 000011C1 42                         inc     dx
26260 000011C2 EE                         out     dx,al               ; out 02f6h,0ffh
26261 000011C3 42                         inc     dx
26262 000011C4 EE                         out     dx,al               ; out 02f7h,0ffh
26263
26264                                   ;sb secondary global rearm
26265
26266 000011C5 B800F0                     mov     ax,0F000h           ;get machine type
26267 000011C8 8EC0                       mov     es,ax
26268 000011CA 26803EFEFFFC               cmp     byte [es:0FFFEh],0FCh ;q:is it a at type machine
26269 000011D0 740D                       je      short startrearm    ; *if at no need to check
26270
26271 000011D2 B4C0                       mov     ah,0C0h             ;get system configuration
26272 000011D4 CD15                       int     15h                 ; *
26273 000011D6 7216                       jc      short finishrearm   ; *jmp if old rom
26274
26275                                   ; test feature byte for secondary interrupt controller
```

```
26276
26277 000011D8 26F6470540                test    byte [es:bx+5],40h
26278                                     ; 24/10/2022
26279                                     ;test   byte [es:bx+ROMBIOS_DESC.bios_sd_featurebyte1],ScndIntController
26280 000011DD 740F                       je      short finishrearm     ;jmp if it is there
26281
26282                                   startrearm:
26283 000011DF B0FF                       mov     al,0FFh               ;write any pattern to port
26284 000011E1 BAF206                     mov     dx,6F2h               ;get starting address
26285 000011E4 EE                         out     dx,al                 ;out 06f2h,0ffh
26286 000011E5 42                         inc     dx                    ;bump address
26287 000011E6 EE                         out     dx,al                 ;out 06f3h,0ffh
26288 000011E7 42                         inc     dx                    ;bump address
26289 000011E8 EE                         out     dx,al                 ;out 06f4h,0ffh
26290 000011E9 42                         inc     dx                    ;bump address
26291 000011EA 42                         inc     dx                    ;bump address
26292 000011EB EE                         out     dx,al                 ;out 06f6h,0ffh
26293 000011EC 42                         inc     dx                    ;bump address
26294 000011ED EE                         out     dx,al                 ;out 06f7h,0ffh
26295
26296                                   finishrearm:
26297 000011EE 07                         pop     es
26298 000011EF 5A                         pop     dx
26299 000011F0 5B                         pop     bx
26300 000011F1 58                         pop     ax
26301
26302                                   ;    global rearm end ********************
26303
26304                                   ; ---------------------------------------------------------------------
26305                                   ; allocate sysinit_base for install= command
26306                                   ; ---------------------------------------------------------------------
26307                                   ; sysinit_base allocation.
26308                                   ;   check if endfile has been called to handle install= command.
26309
26310                                   set_sysinit_base:
26311
26312                                   ; ---------------------------------------------------------------------
26313                                   ;sysinit_base will be established in the secure area of
26314                                   ;lower memory when it handles the first install= command.
26315                                   ;sysinit_base is the place where the actual exec function will be called and
26316                                   ;will check sysinit module in high memory if it is damaged by the application
26317                                   ;program.  if sysinit module has been broken,then "memory error..." message
26318                                   ;is displayed by sysinit_base.
26319                                   ; ---------------------------------------------------------------------
26320
26321                                       ; 24/10/2022 - Retro DOS v4.0 (MSDOS 5.0 IO.SYS, SYSINIT)
26322                                       ; (SYSINIT:1028h)
26323
26324                                       ; 11/12/2022
26325                                       ; ds = cs
26326 000011F2 50                         push    ax                    ; set devmark for mem command
26327 000011F3 A1[6403]                   mov     ax,[memhi]
26328 000011F6 2B06[6803]                 sub     ax,[area]
26329 000011FA A3[6003]                   mov     [impossible_owner_size],ax ;remember the size in case.
26330                                     ;mov    al,devmark_inst ; 'T'
26331 000011FD B054                       mov     al,'T'
26332 000011FF E88904                     call    setdevmark
26333 00001202 58                         pop     ax
26334
26335 00001203 8B3E[6403]                 mov     di,[memhi]
26336 00001207 8EC7                       mov     es,di
26337 00001209 893E[D402]                 mov     [sysinit_base_ptr+2],di ; save this entry for the next use.
26338 0000120D 31FF                       xor     di,di
26339 0000120F 893E[D202]                 mov     [sysinit_base_ptr],di ; es:di -> destination.
26340 00001213 BE[2113]                   mov     si,sysinit_base             ;ds:si -> source code to be relocated.
26341 00001216 B98100                     mov     cx,end_sysinit_base-sysinit_base ; 129
26342                                     ; 24/10/2022
26343                                     ;mov    cx,128  ; 11DCh-115Ch  ; (MSDOS 5.0 IO.SYS, SYSINIT)
26344 00001219 010E[6203]                 add     [memlo],cx
26345                                     ;or     byte cs:[setdevmarkflag],for_devmark ; 2
26346                                     ; 11/12/2022
26347                                     ; ds = cs
26348                                     ;or     byte [cs:setdevmarkflag],2
26349 0000121D 800E[6919]02               or      byte [setdevmarkflag],2
26350                                     ;or     byte [setdevmarkflag],for_devmark
26351 00001222 E8CD35                     call    round                 ; check mem error. also,readjust memhi for the next use.
26352 00001225 F3A4                       rep     movsb                 ; reallocate it.
26353
26354 00001227 C706[D602][0813]           mov     word [sysinit_ptr],sysinitptr ; returning address from
26355 0000122D 8C0E[D802]                 mov     [sysinit_ptr+2],cs      ; sysinit_base back to sysinit.
26356                                     ;or     word [install_flag],has_installed ; set the flag.
26357                                     ;or     byte [install_flag],has_installed ; 2
26358                                     ; 11/12/2022
26359 00001231 800E[CE02]02              or      byte [install_flag],2
26360                                     ; 24/10/2022
26361                                     ;or     word [install_flag],2
26362
26363                                   ; ---------------------------------------------------------------------
26364                                   ; free the rest of the memory from memhi to confbot. still from confbot to
26365                                   ; the top of the memory will be allocated for sysinit and config.sys if
26366                                   ; have_install_cmd.
26367                                   ; ---------------------------------------------------------------------
26368
26369 00001236 E8B935                      call    round
26370 00001239 8B1E[6403]                 mov     bx,[memhi]
26371 0000123D A1[6803]                   mov     ax,[area]
26372 00001240 A3[5E03]                   mov     [old_area],ax         ; save [area]
26373 00001243 8EC0                       mov     es,ax                 ;calc what we needed
26374 00001245 29C3                       sub     bx,ax
26375                                     ; 24/10/2022
26376 00001247 B44A                       mov     ah,4Ah ; SETBLOCK
26377 00001249 CD21                       int     21h                   ;give the rest back
26378                                             ; DOS - 2+ - ADJUST MEMORY BLOCK SIZE (SETBLOCK)
26379                                             ; ES = segment address of block to change
26380                                             ; BX = new size in paragraphs
26381 0000124B 06                         push    es
26382 0000124C 8CC0                       mov     ax,es
26383 0000124E 48                         dec     ax
26384 0000124F 8EC0                       mov     es,ax                 ;point to arena
26385                                     ;mov    word [es:ARENA.OWNER],8       ;set impossible owner
26386                                     ;;mov   word [es:ARENA.NAME],4453h    ; System Data
26387                                     ;mov    word [es:ARENA.NAME],'SD'     ; System Data
26388                                     ; 24/10/2022
26389 00001251 26C70601000800             mov     word [es:1],8         ;set impossible owner
26390 00001258 26C70608005344             mov     word [es:8],'SD'      ; System Data
26391 0000125F 07                         pop     es
26392
26393 00001260 BBFFFF                     mov     bx,0FFFFh
26394 00001263 B448                       mov     ah,48h ; ALLOC
26395 00001265 CD21                       int     21h
26396 00001267 B448                       mov     ah,48h ; ALLOC
26397 00001269 CD21                       int     21h                   ; allocate the rest of the memory
26398                                             ; DOS - 2+ - ALLOCATE MEMORY
26399                                             ; BX = number of 16-byte paragraphs desired
```

```
26400 0000126B A3[6403]                   mov     [memhi],ax              ; start of the allocated memory
26401 0000126E C706[6203]0000             mov     word [memlo],0          ;  to be used next.
26402
26403                                   ;;;; at this moment,memory from [memhi]:0 to top-of-the memory is
26404                                   ;;;; allocated.
26405                                   ;;;; to protect sysinit,confbot module (from confbot (or =alloclim at
26406                                   ;;;; this time) to the top-of-the memory),here we are going to
26407                                   ;;;; 1). "setblock" from memhi to confbot.
26408                                   ;;;; 2). "alloc" from confbot to the top of the memory.
26409                                   ;;;; 3). "free alloc memory" from memhi to confbot.
26410
26411                                   ;memory allocation for sysinit,confbot module.
26412
26413 00001274 8EC0                         mov     es,ax
26414                                       ; 31/12/2022 - Retro DOS v4.2 (Modified MSDOS 6.21 IO.SYS)
26415                                       ; (SYSINIT:11DFh)
26416 00001276 8B1E[A302]                 mov     bx,[CONFBOT]
26417                                       ; 24/10/2022
26418                                       ;mov   bx,[top_of_cdss] ; mov bx,[confbot]
26419 0000127A 29C3                       sub     bx,ax                   ; confbot - memhi
26420 0000127C 4B                         dec     bx                      ; make a room for the memory block id.
26421 0000127D 4B                         dec     bx                      ; make sure!!!.
26422 0000127E B44A                       mov     ah,4Ah ; SETBLOCK
26423 00001280 CD21                       int     21h                     ; this will free (confbot to top of memory)
26424                                             ; DOS - 2+ - ADJUST MEMORY BLOCK SIZE (SETBLOCK)
26425                                             ; ES = segment address of block to change
26426                                             ; BX = new size in paragraphs
26427 00001282 BBFFFF                     mov     bx,0FFFFh
26428 00001285 B448                       mov     ah,48h ; ALLOC
26429 00001287 CD21                       int     21h
26430 00001289 B448                       mov     ah,48h ; ALLOC
26431 0000128B CD21                       int     21h                     ; allocate (confbot to top of memory)
26432                                             ; DOS - 2+ - ALLOCATE MEMORY
26433                                             ; BX = number of 16-byte paragraphs desired
26434 0000128D A3[6803]                   mov     [area],ax               ; save allocated memory segment.
26435                                             ;                       ; need this to free this area for command.com.
26436 00001290 8E06[6403]                 mov     es,[memhi]
26437 00001294 B449                       mov     ah,49h                  ; free allocated memory.
26438 00001296 CD21                       int     21h                     ; free (memhi to confbot(=area))
26439                                             ; DOS - 2+ - FREE MEMORY
26440                                             ; ES = segment address of area to be freed
26441                                   endfile_ret:
26442 00001298 C3                         retn
26443
26444                                   ; End of "EndFile" DOS structure configuration.
26445
26446                                   ; ----------------------------------------------------------------------
26447                                   ; 26/03/2019 - Retro DOS v4.0
26448                                   ; 24/10/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 IO.SYS, SYSINIT)
26449                                   ; ----------------------------------------------------------------------
26450                                   ; Do_Install_Exec
26451                                   ;
26452                                   ; This procedure is used to EXEC a program being loaded via the
26453                                   ; "install=" mechanism in config.sys. It does this by setting up
26454                                   ; the parameters, and then jumping to sysinit_base, which has been
26455                                   ; setup in low memory. When complete, sysinit_base will jump back
26456                                   ; up to this procedure (if sysinit remains uncorrupted by the installed
26457                                   ; program).
26458
26459                                   ;SYSINIT:10CFh:
26460
26461                                   do_install_exec:                ; now,handles install= command.
26462
26463 00001299 56                         push    si                      ; save si for config.sys again.
26464
26465                                   ; we are going to call load/exec function.
26466                                   ; set es:bx to the parameter block here;;;;;;;
26467                                   ; set ds:dx to the asciiz string. remember that we already has 0
26468                                   ; after the filename. so parameter starts after that. if next
26469                                   ; character is a line feed (i.e. 10),then assume that the 0
26470                                   ; we already encountered used to be a carrage return. in this
26471                                   ; case,let's set the length to 0 which will be followed by
26472                                   ; carridge return.
26473
26474                                   ; es:si -> command line in config.sys. points to the first non blank
26475                                   ;character after =.
26476
26477 0000129A 06                           push    es
26478 0000129B 1E                         push    ds
26479 0000129C 07                         pop     es
26480 0000129D 1F                         pop     ds                      ; es->sysinitseg,ds->confbot seg
26481 0000129E 89F2                       mov     dx,si                   ; ds:dx->file name,0 in config.sys image.
26482
26483 000012A0 31C9                       xor     cx,cx
26484 000012A2 FC                         cld
26485 000012A3 2EC606[F102]20             mov     byte [cs:ldexec_start],' ' ; clear out the parm area
26486 000012A9 BF[F202]                   mov     di,ldexec_parm
26487                                   installfilename:                ; skip the file name
26488 000012AC AC                         lodsb                           ; al = ds:si; si++
26489                                     ; 05/09/2023
26490 000012AD 08C0                       or      al,al
26491                                     ;cmp    al,0
26492                                     ;je     short got_installparm
26493                                     ;jmp    short installfilename
26494                                     ; 10/04/2024
26495 000012AF 75FB                       jnz     short installfilename
26496                                   got_installparm:                ; copy the parameters to ldexec_parm
26497 000012B1 AC                         lodsb
26498 000012B2 268805                     mov     [es:di],al
26499 000012B5 3C0A                       cmp     al,lf   ; cmp al,0Ah   ; line feed?
26500 000012B7 7405                       je      short done_installparm
26501 000012B9 FEC1                       inc     cl                      ; # of char. in the parm.
26502 000012BB 47                         inc     di
26503 000012BC EBF3                       jmp     short got_installparm
26504                                   done_installparm:
26505 000012BE 2E880E[F002]               mov     byte [cs:ldexec_line],cl ; length of the parm.
26506                                     ; 05/09/2023
26507 000012C3 08C9                       or      cl,cl
26508                                     ;cmp    cl,0                    ; if no parm,then
26509 000012C5 7506                       jne     short install_seg_set ; let the parm area
26510 000012C7 2EC606[F102]0D             mov     byte [cs:ldexec_start],cr ; 0Dh
26511                                                                     ; starts with cr.
26512                                   install_seg_set:
26513                                     ; 05/09/2023 - Retro DOS v4.2 IO.SYS (Optimization)
26514 000012CD 31DB                       xor     bx,bx
26515                                     ;mov    word [cs:0],0           ; make a null environment segment
26516 000012CF 2E891F                     mov     [cs:bx],bx ; 05/09/2023
26517 000012D2 8CC8                       mov     ax,cs                   ; by overlap jmp instruction of sysinitseg.
26518
26519                                   ;-----------------------------------------------M067----------------
26520                                   ;
26521                                   ;   the environment pointer is made 0. so the current environment ptr.
26522                                   ;   will be the same as pdb_environ which after dosinit is 0.
26523                                   ;
```

```
                                ;   mov    cs:[instexe.exec0_environ],0 ; set the environment seg.
                                ;
                                ;   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 87F9                       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
00001406 837F02FF                       cmp     word [bx+2],0FFFFh ; is the buffer (already) in HMA ?
0000140A 7423                           je      short gba_2     ; yes
                                    gba_1:
                                        ;;;

0000140C 2EA1[9D02]                     mov     ax, [cs:singlebuffersize]
00001410 2EF726[9902]                   mul     word [cs:buffers]
                                        ;add    ax,0Fh
00001415 83C00F                         add     ax,15
                                        ; 02/11/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
                                        ;and    ax,~15  ; 0FFF0h        ; para round
                                        ; 12/12/2022
00001418 24F0                           and     al,~15  ; 0F0h
0000141A 89C3                           mov     bx,ax
0000141C B8024A                         mov     ax,4A02h
                                        ;mov    ax,((multMULT<<8)+multMULTALLOCHMA)
0000141F CD2F                           int     2Fh     ; DOS 5+ - ALLOCATE HMA SPACE
                                                        ;     AX = 4A02h
                                                        ;     BX = number of bytes
                                                        ; Return:
                                                        ;     ES:DI -> start of allocated HMA block or FFFFh:FFFFh
                                                        ;     BX = number of bytes actually allocated
                                                        ;          (rounded up to next paragraph)
                                                        ; Notes:
                                                        ;     this call is not valid unless DOS is loaded in the HMA
                                                        ;     (DOS=HIGH)

00001421 83FFFF                         cmp     di,0FFFFh
00001424 7506                           jne     short got_hma

                                        ;mov    di,0            ; dont xor di,di Z flag needed
                                        ; 05/09/2023
                                        ; zf=1
00001426 47                             inc     di ; 0FFFFh -> 0
                                        ; zf=1
                                        ;zf=1
                                        ;xor    di,di   ; 25/10/2022
                                        ;zf=1
00001427 2E8E06[6403]                   mov     es,[cs:memhi]
                                    got_hma:
0000142C 5A                             pop     dx
0000142D 5B                             pop     bx
0000142E C3                             retn

                                        ; 11/04/2024 - Retro DOS v5.0
                                        ; PCDOS 7.1 IBMBIO.COM
                                        ;;;
                                    gba_2:
0000142F C43F                           les     di,[bx]
00001431 09FF                           or      di,di
                                        ;pop    dx
                                        ;pop    bx
                                        ;retn
                                        ; 11/04/2024 - Retro DOS v5.0
00001433 EBF7                           jmp     short got_hma
                                        ;;;

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

                                    set_buffer_info:

                                    ;function: set buf_link,buf_id,buf_sector
                                    ;
                                    ;in: es:di -> buffer header to be set.
                                    ;    ax = di
                                    ;
                                    ;out:
                                    ;    above entries set.

                                        ; 25/10/2022
00001435 2EFF36[BD02]                   push    word [cs:buf_prev_off]
                                        ;pop    word [es:di+buffinfo.buf_prev]
0000143A 268F4502                       pop     word [es:di+2]
0000143E 2EA3[BD02]                     mov     [cs:buf_prev_off],ax
00001442 2E0306[9D02]                   add     ax,[cs:singlebuffersize]     ;adjust ax
                                        ;mov    [es:di+buffinfo.buf_next],ax
00001447 268905                         mov     [es:di],ax
                                        ;mov    word [es:di+buffinfo.buf_ID],00FFh  ; new buffer free
0000144A 26C74504FF00                   mov     word [es:di+4],00FFh
                                        ;mov    word [es:di+buffinfo.buf_sector],0   ; to compensate the masm 3 bug
00001450 26C745060000                   mov     word [es:di+6],0
                                        ;mov    word [es:di+buffinfo.buf_sector+2],0 ; to compensate the masm 3 bug
00001456 26C745080000                   mov     word [es:di+8],0
0000145C C3                             retn

                                    ; ============================================================================
                                    ; MSSTACK initialization routine - MSDOS 6.0 - SYSDINIT1.ASM - 1991
                                    ; ----------------------------------------------------------------------------
                                    ; 27/03/2019 - Retro DOS v4.0

                                    ; --------------------------------------------------------------------------
                                    ; ibmstack initialization routine.
                                    ;
                                    ;   to follow the standard interrupt sharing scheme, msstack.asm
                                    ;   has been modified. this initialization routine also has to
                                    ;   be modified because for the interrupt level 7 and 15, firstflag
                                    ;   should be set to signal that this interrupt handler is the
                                    ;   first handler hooked to this interrupt vector.
                                    ;   we determine this by looking at the instruction pointed by
                                    ;   this vector. if it is iret, then this handler should be the
                                    ;   first one. in our case, only the interrupt vector 77h is the
                                    ;   interrupt level 15. (we don't hook interrupt level 7.)
                                    ;
                                    ;   the followings are mainly due to m.r.t; ptm fix of p886 12/3/86
                                    ;   some design changes are needed to the above interrupt sharing
                                    ;   method. the above sharing scheme assumes that 1). interrupt
                                    ;   sharing is never done on levels that have bios support. 2). "phantom"
                                    ;   interrupts would only be generated on levels 7 and 15.
                                    ;   these assumptions are not true any more. we have to use the firstflag
                                    ;   for every level of interrupt. we will set the firstflag on the following
                                    ;   conditions:
                                    ;
                                    ;    a.     if the cs portion of the vector is 0000, then "first"
                                    ;    b. else if cs:ip points to valid shared header, then not "first"
                                    ;    c. else if cs:ip points to an iret, then "first"
                                    ;    d. else if cs:ip points to dummy, then "first"
                                    ;
                                    ;   where dummy is - the cs portion must be f000, and the ip portion must
                                    ;   be equal to the value at f000:ff01. this location is the initial value
                                    ;   from vector_table for interrupt 7, one of the preserved addresses in all
                                    ;   the bioses for all of the machines.
```

```
                                ;
                                ;   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
```

```
27266                                       lds     bx,[es:si]
27267                                       push    ds
27268                                       pop     dx
27269
27270                                       cmp     dx,0
27271                                       je      short int_0A_first
27272
27273                                       cmp     byte [bx],0CFh
27274                                       je      short int_0A_first
27275
27276                                       cmp     word [bx+6],424Bh
27277                                       je      short int_0A_not_first
27278
27279                                       cmp     dx,0F000h
27280                                       jne     short int_0A_not_first
27281
27282                                       push    es
27283                                       push    dx
27284                                       mov     dx,0F000h
27285                                       mov     es,dx
27286                                       cmp     bx,[es:0FF01h]
27287                                       pop     dx
27288                                       pop     es
27289                                       je      short int_0A_first
27290                                   %Endif
27291
27292                                       ; 14/12/2022
27293                                       ; 25/10/2022
27294 00001537 E8EB00                     call    int_xx_first_check ; 27/03/2019 - Retro DOS v4.0
27295 0000153A 730C                       jnc     short int_0A_first
27296
27297                                   int_0A_not_first:
27298                                       ; 14/12/2022
27299                                       ; 25/10/2022
27300                                       ;pop    ds
27301 0000153C BF[C205]                   mov     di,INT19OLD0A
27302 0000153F BB[5900]                   mov     bx,old0A
27303 00001542 BA[5700]                   mov     dx,int0A
27304 00001545 E80701                     call    new_init_loop
27305
27306                                       ; 14/12/2022
27307                                       ;jmp    short int_0A_end
27308                                   ;int_0A_first:
27309                                       ; 25/10/2022
27310                                       ;pop    ds
27311
27312                                       ; 14/12/2022
27313                                   int_0A_first:
27314                                   int_0A_end:
27315
27316                                   stkinit_0B:
27317 00001548 BE2C00                     mov     si,0Bh*4 ; 44
27318
27319                                       ; 14/12/2022
27320                                       ; 25/10/2022
27321 0000154B E8D700                     call    int_xx_first_check ; 27/03/2019 - Retro DOS v4.0
27322 0000154E 730C                       jnc     short int_0B_end ; int_0B_first
27323
27324                                   ; 14/12/2022
27325                                   %if 0
27326                                       ; 25/10/2022 (MSDOS 5.0 IO.SYS compatibility)
27327                                       push    ds
27328                                       lds     bx,[es:si]
27329                                       push    ds
27330                                       pop     dx
27331
27332                                       cmp     dx,0
27333                                       je      short int_0B_first
27334
27335                                       cmp     byte [bx],0CFh
27336                                       je      short int_0B_first
27337
27338                                       cmp     word [bx+6],424Bh
27339                                       je      short int_0B_not_first
27340
27341                                       cmp     dx,0F000h
27342                                       jne     short int_0B_not_first
27343
27344                                       push    es
27345                                       push    dx
27346                                       mov     dx,0F000h
27347                                       mov     es,dx
27348                                       cmp     bx,[es:0FF01h]
27349                                       pop     dx
27350                                       pop     es
27351                                       je      short int_0B_first
27352                                   %endif
27353
27354                                   int_0B_not_first:
27355                                       ; 14/12/2022
27356                                       ; 25/10/2022
27357                                       ;pop    ds
27358 00001550 BF[C705]                   mov     di,INT19OLD0B
27359 00001553 BB[7100]                   mov     bx,old0B
27360 00001556 BA[6F00]                   mov     dx,int0B
27361 00001559 E8F300                     call    new_init_loop
27362
27363                                       ; 14/12/2022
27364                                       ;jmp    short int_0B_end
27365                                   ;int_0B_first:
27366                                       ; 25/10/2022
27367                                       ;pop    ds
27368
27369                                   int_0B_end:
27370
27371                                   stkinit_0C:
27372 0000155C BE3000                     mov     si,0Ch*4 ; 48
27373
27374                                       ; 14/12/2022
27375                                       ; 25/10/2022
27376 0000155F E8C300                     call    int_xx_first_check
27377 00001562 730C                       jnc     short int_0C_end ; int_0C_first
27378
27379                                   ; 14/12/2022
27380                                   %if 0
27381                                       ; 25/10/2022 (MSDOS 5.0 IO.SYS compatibility)
27382                                       push    ds
27383                                       lds     bx,[es:si]
27384                                       push    ds
27385                                       pop     dx
27386
27387                                       cmp     dx,0
27388                                       je      short int_0C_first
27389
```

```
                                    cmp     byte [bx],0CFh
                                    je      short int_0C_first

                                    cmp     word [bx+6],424Bh
                                    je      short int_0C_not_first

                                    cmp     dx,0F000h
                                    jne     short int_0C_not_first

                                    push    es
                                    push    dx
                                    mov     dx,0F000h
                                    mov     es,dx
                                    cmp     bx,[es:0FF01h]
                                    pop     dx
                                    pop     es
                                    je      short int_0C_first
                                %endif

                                int_0C_not_first:
                                    ; 14/12/2022
                                    ; 25/10/2022
                                    ;pop    ds
00001564 BF[CC05]                   mov     di,INT19OLD0C
00001567 BB[8900]                   mov     bx,old0C
0000156A BA[8700]                   mov     dx,int0C
0000156D E8DF00                     call    new_init_loop

                                    ; 14/12/2022
                                    ;jmp    short int_0C_end
                                ;int_0C_first:
                                    ; 25/10/2022
                                    ;pop    ds

                                int_0C_end:

                                stkinit_0D:
00001570 BE3400                     mov     si,0Dh*4 ; 52

                                    ; 14/12/2022
                                    ; 25/10/2022
00001573 E8AF00                     call    int_xx_first_check
00001576 730C                       jnc     short int_0D_end ; int_0D_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_0D_first

                                    cmp     byte [bx],0CFh
                                    je      short int_0D_first

                                    cmp     word [bx+6],424Bh
                                    je      short int_0D_not_first

                                    cmp     dx,0F000h
                                    jne     short int_0D_not_first

                                    push    es
                                    push    dx
                                    mov     dx,0F000h
                                    mov     es,dx
                                    cmp     bx,[es:0FF01h]
                                    pop     dx
                                    pop     es
                                    je      short int_0D_first
                                %endif

                                int_0D_not_first:
                                    ; 14/12/2022
                                    ; 25/10/2022
                                    ;pop    ds
00001578 BF[D105]                   mov     di,INT19OLD0D
0000157B BB[A100]                   mov     bx,old0D
0000157E BA[9F00]                   mov     dx,int0D
00001581 E8CB00                     call    new_init_loop

                                    ; 14/12/2022
                                    ;jmp    short int_0D_end
                                    ; 02/11/2022
                                ;int_0D_first:
                                    ;pop    ds

                                int_0D_end:

                                stkinit_0E:
00001584 BE3800                     mov     si,0Eh*4 ; 56

                                    ; 14/12/2022
                                    ; 25/10/2022
00001587 E89B00                     call    int_xx_first_check
0000158A 730C                       jnc     short int_0E_end ; int_0E_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_0E_first

                                    cmp     byte [bx],0CFh
                                    je      short int_0E_first

                                    cmp     word [bx+6],424Bh
                                    je      short int_0E_not_first

                                    cmp     dx,0F000h
                                    jne     short int_0E_not_first

                                    push    es
                                    push    dx
                                    mov     dx,0F000h
                                    mov     es,dx
                                    cmp     bx,[es:0FF01h]
```

```
                                        pop     dx
                                        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
```

```
27762 000015F0 730C                     jnc     short int_77_end ; int_77_first
27763
27764                                   ; 14/12/2022
27765                                   %if 0
27766                                       ; 25/10/2022 (MSDOS 5.0 IO.SYS compatibility)
27767                                       push    ds
27768                                       lds     bx,[es:si]
27769                                       push    ds
27770                                       pop     dx
27771
27772                                       cmp     dx,0
27773                                       je      short int_77_first
27774
27775                                       cmp     byte [bx],0CFh
27776                                       je      short int_77_first
27777
27778                                       cmp     word [bx+6],424Bh
27779                                       je      short int_77_not_first
27780
27781                                       cmp     dx,0F000h
27782                                       jne     short int_77_not_first
27783
27784                                       push    es
27785                                       push    dx
27786                                       mov     dx,0F000h
27787                                       mov     es,dx
27788                                       cmp     bx,[es:0FF01h]
27789                                       pop     dx
27790                                       pop     es
27791                                       je      short int_77_first
27792                                   %endif
27793
27794                                   int_77_not_first:
27795                                       ; 14/12/2022
27796                                       ; 25/10/2022
27797                                       ;pop    ds
27798 000015F2 BF[F405]                   mov     di,INT19OLD77
27799 000015F5 BB[3101]                   mov     bx,old77
27800 000015F8 BA[2F01]                   mov     dx,int77
27801 000015FB E85100                     call    new_init_loop
27802
27803                                       ; 14/12/2022
27804                                       ;jmp    short int_77_end
27805                                   ;int_77_first:
27806                                       ; 25/10/2022
27807                                       ;pop    ds
27808
27809                                   int_77_end:
27810 000015FE 1E                         push    ds
27811 000015FF B800F0                     mov     ax,0F000h               ; look at the model byte
27812 00001602 8ED8                       mov     ds,ax
27813 00001604 803EFEFFF9                 cmp     byte [0FFFEh],0F9h ; mdl_convert ; pc convertible?
27814 00001609 1F                         pop     ds
27815 0000160A 7504                       jne     short skip_enablenmis
27816
27817 0000160C B027                       mov     al,27h                  ; enable convertible nmis
27818 0000160E E672                       out     72h,al
27819
27820                                   ; 25/10/2022
27821                                   ; (MSDOS 5.0 SYSINIT:15FBh)
27822
27823                                   skip_enablenmis:
27824 00001610 FB                         sti
27825                                     ;;mov   ax,Bios_Data ; 70h
27826                                     ;mov    ax,KERNEL_SEGMENT ; 70h
27827                                     ; 21/10/2022
27828 00001611 B87000                     mov     ax,DOSBIODATASEG ; 0070h
27829 00001614 8ED8                       mov     ds,ax
27830
27831                                     ;mov    [640h],1 ; SYSINIT:1736h for MSDOS 6.21 IO.SYS
27832
27833 00001616 C606[B105]01               mov     byte [INT19SEM],1       ; indicate that int 19
27834                                                                       ; initialization is complete
27835
27836 0000161B 5D                         pop     bp                      ; restore all
27837 0000161C 5E                         pop     si
27838 0000161D 5F                         pop     di
27839 0000161E 5A                         pop     dx
27840 0000161F 59                         pop     cx
27841 00001620 5B                         pop     bx
27842 00001621 07                         pop     es
27843 00001622 1F                         pop     ds
27844 00001623 58                         pop     ax
27845 00001624 C3                         retn
27846
27847                                   ; 14/12/2022
27848                                   ; ----------------------------------------------------------------------------
27849
27850                                       ; 14/12/2022
27851                                       ; 25/10/2022
27852                                   ;%if 0
27853                                       ; 27/03/2019 - Retro DOS v4.0
27854                                   int_xx_first_check:
27855 00001625 1E                         push    ds
27856 00001626 26C51C                     lds     bx,[es:si]
27857 00001629 1E                         push    ds
27858 0000162A 5A                         pop     dx
27859
27860                                     ;cmp    dx,0
27861                                     ;je     short int_xx_first
27862                                     ; 05/09/2023
27863 0000162B 21D2                       and     dx,dx
27864 0000162D 741E                       jz      short int_xx_first
27865
27866 0000162F 803FCF                     cmp     byte [bx],0CFh
27867 00001632 7419                       je      short int_xx_first
27868
27869 00001634 817F064B42                 cmp     word [bx+6],424Bh
27870 00001639 7411                       je      short int_xx_not_first
27871
27872 0000163B 81FA00F0                   cmp     dx,0F000h
27873 0000163F 750B                       jne     short int_xx_not_first
27874
27875 00001641 06                         push    es
27876                                     ;push   dx
27877                                     ;mov    dx,0F000h
27878 00001642 8EC2                       mov     es,dx
27879 00001644 263B1E01FF                 cmp     bx,[es:0FF01h]
27880                                               ;pop    dx
27881 00001649 07                         pop     es
27882 0000164A 7401                       je      short int_xx_first
27883
27884                                   int_xx_not_first:
27885 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
```

```
28132 0000174D C706[9003][9403]          mov     word [MagicBackdoor],NullBackdoor
28133                                pre_exit_err:
28134 00001753 B84000                    mov     ax,40h                  ; SYSPRE_BADFILE_ERROR
28135                                                                    ; (problem loading dblspace.bin)
28136 00001756 C3                        retn
28137
28138                                no_driver_version_fail:
28139 00001757 09C0                      or      ax,ax                   ; error code returned?
28140 00001759 75EE                      jnz     short fail_driver_load
28141
28142                                magic_is_resident:
28143                                    ; 13/04/2024
28144                                    ; ds = cs
28145                                    ;mov    ax,[cs:break_addr]
28146 0000175B A1[7D03]                  mov     ax,[break_addr]
28147 0000175E E8B4FB                    call    ParaRound               ; convert to paragraphs
28148                                    ;add    ax,[cs:break_addr+2]
28149                                    ;mov    [cs:DevBrkAddr+2],ax
28150                                    ;mov    word [cs:DevBrkAddr],0; store normalized end here
28151 00001761 0306[7F03]                add     ax,[break_addr+2]
28152 00001765 A3[3F24]                  mov     [DevBrkAddr+2],ax
28153 00001768 C706[3D24]0000            mov     word [DevBrkAddr],0
28154 0000176E BB0400                    mov     bx,4                    ; inquire how many paragraphs it wants
28155                                    ;call   far [cs:MagicBackdoor]
28156 00001771 FF1E[9003]                call    far [MagicBackdoor]
28157                                    ;mov    bx,[cs:ALLOCLIM]        ; get top of free memory
28158 00001775 8B1E[A502]                mov     bx,[ALLOCLIM]
28159 00001779 29C3                      sub     bx,ax                   ; see how much we'll lower it
28160                                    ;cmp    bx,[cs:DevBrkAddr+2]    ; is there that much room free?
28161 0000177B 3B1E[3F24]                cmp     bx,[DevBrkAddr+2]
28162 0000177F 7212                      jb      short cant_move_driver
28163                                    ;;sub   [cs:ALLOCLIM],ax        ; (mov [cs:ALLOCLIM],bx)
28164                                    ;mov    [cs:ALLOCLIM],bx ; Retro DOS v5.0 ; 08/04/2024
28165                                    ; 13/04/2024
28166 00001781 891E[A502]                mov     [ALLOCLIM],bx
28167                                    ;mov    es,[cs:ALLOCLIM]
28168 00001785 8E06[A502]                mov     es,[ALLOCLIM]
28169 00001789 BB0600                    mov     bx,6                    ; tell the driver to move itself
28170                                    ;call   far [cs:MagicBackdoor]
28171 0000178C FF1E[9003]                call    far [MagicBackdoor]
28172                                    ;mov    [cs:DevBrkAddr+2],ax    ; save end of low stub
28173 00001790 A3[3F24]                  mov     [DevBrkAddr+2],ax       ; save end of low stub
28174
28175                                cant_move_driver:
28176                                    ;mov    ax,[cs:DevBrkAddr+2]    ; get terminate segment
28177 00001793 A1[3F24]                  mov     ax,[DevBrkAddr+2]
28178                                    ;cmp    ax,[cs:DevLoadEnd]      ; terminate size TOO big?
28179 00001796 3B06[3724]                cmp     ax,[DevLoadEnd]
28180 0000179A 77B7                      ja      short pre_exit_err      ; error out if so
28181
28182                                ;------ deal with block device drivers
28183
28184                                _isblock:                           ; if no units found,erase the device
28185                                    ; 13/04/2024
28186                                    ; ds = cs
28187                                    ;mov    al,[cs:unitcount]
28188 0000179C A0[7C03]                  mov     al,[unitcount]
28189 0000179F 08C0                      or      al,al
28190 000017A1 74B0                      jz      short pre_exit_err
28191 000017A3 30E4                      xor     ah,ah
28192                                    ;lds    si,[cs:DevEntry]        ; set ds:si to header
28193 000017A5 C536[3924]                lds     si,[DevEntry]
28194 000017A9 88440A                    mov     [si+10],al              ; mov [si+SYSDEV.NAME],al
28195                                                                    ; number of units in name field
28196                                                                    ; device drivers are *supposed*
28197                                                                    ; to do this for themselves.
28198 000017AC 89C1                      mov     cx,ax
28199 000017AE 2EC43E[6D02]              les     di,[cs:DOSINFO]                 ; es:di point to dos info
28200 000017B3 268A6520                  mov     ah,[es:di+20h]          ; [es:di+SYSI_NUMIO]
28201                                                                    ; get number of devices
28202 000017B7 88E2                      mov     dl,ah
28203 000017B9 00C4                      add     ah,al                   ; check for too many devices
28204 000017BB 80FC1A                    cmp     ah,26                   ; 'A' - 'Z' is 26 devices
28205 000017BE 7793                      ja      short pre_exit_err
28206 000017C0 2E800E[6919]02            or      byte [cs:setdevmarkflag],2
28207 000017C6 E83D1F                    call    DevSetBreak
28208                                    ;jnc    short _ok_block
28209                                    ;jmp    pre_exit_err
28210                                    ; 13/04/2024
28211 000017C9 7288                      jc      short pre_exit_err  ; ds <> cs
28212
28213                                _ok_block:
28214 000017CB 26886520                  mov     [es:di+20h],ah          ; [es:di+SYSI_NUMIO] ; update the amount
28215
28216 000017CF 2EC51E[8103]               lds     bx,[cs:bpb_addr]        ; point to bpb array (*)
28217 000017D4 30F6                      xor     dh,dh
28218
28219                                _perunit:
28220 000017D6 2EC42E[6D02]              les     bp,[cs:DOSINFO]
28221 000017DB 26C46E00                  les     bp,[es:bp+0]            ; [es:bp.sysi_dpb]
28222                                                                    ; get first dpb
28223                                                                    ; [es:bp+SysInitvars.SYSI_DPB] ; [es:bp+0]
28224                                _scandpb:
28225 000017DF 26837E19FF                cmp     word [es:bp+19h],0FFFFh ; -1 ; [es:bp.dpb_next_dpb]
28226 000017E4 7406                      jz      short _foundpb
28227 000017E6 26C46E19                  les     bp,[es:bp+19h]          ; les bp,[es:bp.dpb_next_dpb]
28228                                                                    ; [es:bp+DPB.NEXT_DPB]
28229 000017EA EBF3                      jmp     short _scandpb
28230
28231                                ;   We've found the end of the DPB chain. Now extend it.
28232
28233                                _foundpb:
28234 000017EC 2EA1[3D24]                mov     ax,[cs:DevBrkAddr]
28235 000017F0 26894619                  mov     [es:bp+19h],ax          ; [es:bp.dpb_next_dpb] ; DPB.NEXT_DPB
28236 000017F4 2EA1[3F24]                mov     ax,[cs:DevBrkAddr+2]
28237 000017F8 2689461B                  mov     [es:bp+1Bh],ax          ; [es:bp.dpb_next_dpb+2] ; DPB.NEXT_DPB+2
28238 000017FC 2EC42E[3D24]              les     bp,[cs:DevBrkAddr]
28239 00001801 26C74619FFFF              mov     word [es:bp+19h],0FFFFh ; -1
28240 00001807 26C64618FF                mov     byte [es:bp+18h],0FFh ; [es:bp.dpb_first_access],-1
28241                                                                    ; DPB.FIRST_ACCESS
28242 0000180C 2E8306[3D24]3D            add     word [cs:DevBrkAddr],61 ; DPBSIZ ; 3Dh
28243 00001812 E8D01E                    call    RoundBreakAddr
28244 00001815 8B37                      mov     si,[bx]                 ; ds:si points to bpb (*)
28245                                                                    ; (mov si,[bx] ..and then.. add bx,2)
28246                                                                    ; Note: If unit count > 1,bx points to a BPB in the BPB array,
28247                                                                    ; the array address is in [bpb_addr] (*)
28248                                                                    ; Erdogan Tan - 07/07/2023
28249 00001817 26885600                  mov     [es:bp+0],dl                    ; mov word [es:bp.dpb_drive],dx
28250                                                                    ; [es:bp+DPB.DRIVE],dl
28251 0000181B 26887601                  mov     [es:bp+1],dh                    ; [es:bp+DPB.UNIT],dh
28252 0000181F 52                        push    dx
28253 00001820 51                        push    cx
28254 00001821 BA5241                    mov     dx,4152h                ; DX = signature 4152h ('AR') for FAT32 extended BPB/DPB
28255 00001824 31C9                      xor     cx,cx   ; 0
```

```
28256 00001826 26894E1D                 mov     [es:bp+1Dh],cx          ; DPB.NEXT_FREE ; last allocated cluster #
28257 0000182A 394C0B                   cmp     [si+0Bh],cx             ; BPB.fatsecs16 ; [si+A_BPB.BPB_SECTORSPERFAT]
28258 0000182D 7514                     jnz     short _setdpb           ; FAT DPB (33 bytes)
28259                                                                   ; FAT32 DPB (61 bytes)
28260 0000182F 26894E39                 mov     [es:bp+39h],cx          ; DPB.FAT32_NXTFREE = 0
28261 00001833 26894E3B                 mov     [es:bp+3Bh],cx          ; DPB.FAT32_NXTFREE+2 = 0
28262 00001837 49                       dec     cx                      ; 0FFFFh ; -1
28263 00001838 26894E1F                 mov     [es:bp+1Fh],cx          ; DPB.FREE_CNT (-1 = unknown)
28264 0000183C 26894E21                 mov     [es:bp+21h],cx          ; DPB.FREE_CNT+2 (-1 = unknown)
28265 00001840 B95845                   mov     cx,4558h                ; CX = signature 4558h ('EX') for FAT32 extended BPB/DPB
28266
28267                                 _setdpb:
28268 00001843 B453                     mov     ah,53h          ; SETDPB ; hidden system call
28269 00001845 CD21                     int     21h             ; DOS - 2+ internal - TRANSLATE BIOS PARAMETER BLOCK
28270                                                           ; DS:SI -> BPB (BIOS Parameter Block)
28271                                                           ; ES:BP -> buffer for DOS Drive Parameter Block
28272                                                           ; (if CX=4558h & DX=4152h,FAT32 Extended DPB will be set)
28273 00001847 59                       pop     cx
28274 00001848 5A                       pop     dx
28275 00001849 268B4602                 mov     ax,[es:bp+2]            ; [es:bp.dpb_sector_size]  ; [es:bp+DPB.SECTOR_SIZE]
28276 0000184D 06                       push    es
28277 0000184E 2EC43E[6D02]             les     di,[cs:DOSINFO]
28278 00001853 263B4510                 cmp     ax,[es:di+10h]          ; [es:di.sysi_maxsec] ; [es:di+SysInitvars.SYSI_MAXSEC]
28279 00001857 07                       pop     es
28280 00001858 7604                     jbe     short _iblk_1
28281                                   ; 13/04/2024
28282                                   ; ds <> cs
28283 0000185A B84000                   mov     ax,40h                  ; SYSPRE_BADFILE_ERROR ; (pre_exit_err)
28284                                                                   ; (problem loading dblspace.bin)
28285 0000185D C3                       retn
28286
28287                                 _iblk_1:
28288 0000185E 1E                       push    ds
28289 0000185F 2EC506[3924]             lds     ax,[cs:DevEntry]
28290 00001864 26894613                 mov     [es:bp+13h],ax          ; [es:bp+DPB.DRIVER_ADDR]
28291 00001868 268C5E15                 mov     [es:bp+15h],ds          ; [es:bp+DPB.DRIVER_ADDR+2]
28292 0000186C 1F                       pop     ds
28293 0000186D FEC2                     inc     dl                      ; increment drive number
28294 0000186F FEC6                     inc     dh                      ; increment unit number
28295 00001871 43                       inc     bx
28296 00001872 43                       inc     bx                      ; point to next BPB
28297                                                                   ; (in the BPB array) (*) -add bx,2-
28298 00001873 49                       dec     cx                      ; loop _foundpb
28299 00001874 7403                     jz      short _linkit
28300 00001876 E973FF                   jmp     _foundpb
28301
28302                                 _linkit:
28303 00001879 0E                        push    cs
28304 0000187A 1F                       pop     ds
28305 0000187B E825F5                   call    TempCDS                 ; set cds for new drives
28306                                   ; 13/04/2024
28307                                   ; (DS may not be same with CS here)
28308 0000187E 2EC43E[6D02]             les     di,[cs:DOSINFO]                 ; es:di = dos table (SysInitVars)
28309 00001883 268B4522                 mov     ax,[es:di+22h]          ; [es:di+SYSI_DEV] ; dx:cx = head of list
28310 00001887 268B5D24                 mov     bx,[es:di+24h]          ; [es:di+SYSI_DEV+2]
28311 0000188B 2EC536[3924]             lds     si,[cs:DevEntry]        ; ds:si = device location
28312 00001890 8904                     mov     [si],ax                 ; link in the driver
28313 00001892 895C02                   mov     [si+2],bx
28314 00001895 26897522                 mov     [es:di+22h],si          ; [es:di+SYSI_DEV] ; set head of list in dos
28315 00001899 268C5D24                 mov     [es:di+24h],ds          ; [es:di+SYSI_DEV+2]
28316 0000189D E8881E                   call    DevBreak                ; mark successful install
28317                                   ; 13/04/2024
28318                                   ; ds = cs
28319                                   ;mov    cx,[cs:DevBrkAddr+2]    ; pass it a work buffer
28320                                   ;mov    dx,[cs:ALLOCLIM]        ; address in cx (segment)
28321 000018A0 8B0E[3F24]               mov     cx,[DevBrkAddr+2]
28322 000018A4 8B16[A502]               mov     dx,[ALLOCLIM]
28323 000018A8 29CA                     sub     dx,cx                   ; for len dx (paragraphs)
28324 000018AA B80055                   mov     ax,5500h                ; we're shuffle aware,but don't move
28325                                                                   ; any drives at this point.
28326 000018AD BB0200                   mov     bx,2                    ; switch what we can now
28327                                   ;call   far [cs:MagicBackdoor]
28328 000018B0 FF1E[9003]               call    far [MagicBackdoor]
28329                                 pre_exit:                         ; no errors!
28330 000018B4 31C0                     xor     ax,ax                   ; zf=1
28331                                 no_magic:   ; 13/04/2024
28332 000018B6 C3                       retn
28333
28334                                 ; ============== S U B R O U T I N E =======================================
28335
28336                                 ; 08/04/2024 - Retro DOS v5.0
28337                                 ; (PCDOS 7.1 IBMBIO.COM - SYSINIT:1B9Fh)
28338
28339                                 ;***MagicPostload -- called to clean up and make sure Magic is final placed
28340
28341                                 MagicPostload:
28342                                     ; 13/04/3024
28343                                     ; ds = cs
28344 000018B7 E89C00                   call    get_dblspace_version  ; is it there?
28345 000018BA 75FA                     jnz     short no_magic        ; done if not
28346 000018BC F7C20080                 test    dx,8000h              ; is it already permanent?
28347 000018C0 74F4                     jz      short no_magic        ; no,done if so (not in final position)
28348 000018C2 BBFFFF                   mov     bx,0FFFFh ; -1        ; how much space does it want?
28349 000018C5 B8114A                   mov     ax,4A11h              ; multMagicdrv
28350                                                                 ; DBLSPACE.BIN - GET RELOCATION SIZE
28351 000018C8 CD2F                     int     2Fh                   ; get paragraphs into ax
28352 000018CA 40                       inc     ax                    ; extra 2 paragraphs for the stub
28353 000018CB 40                       inc     ax                    ; ((tiny_stub_end-tiny_stub_start)+15)/16
28354                                                                 ; (18+15)/16 = 2
28355                                   ;mov    [cs:DevSize],ax             ; store that (**)
28356                                   ;mov    byte [cs:DeviceHi],0  ; not to be loaded in UMB
28357                                   ;mov    [cs:bpb_addr+2],cs    ; pass name so that
28358                                   ;                             ; arena header can be set
28359                                   ;mov    word [cs:bpb_addr],MagicDDName ; "\DBLSPACE.BIN"
28360                                   ; 13/04/2024
28361                                   ; ds = cs
28362 000018CC A3[3324]                 mov     [DevSize],ax
28363 000018CF C606[5124]00             mov     byte [DeviceHi],0
28364 000018D4 8C0E[8303]               mov     [bpb_addr+2],cs
28365 000018D8 C706[8103][AB16]         mov     word [bpb_addr],MagicDDName
28366
28367 000018DE E8112F                   call    round                 ; normalize memhi:memlo
28368 000018E1 E8811C                   call    InitDevLoad           ; set up sub-arena,DevLoadAddr,
28369                                                                 ; DevLoadEnd,and DevEntry
28370                                                                 ; gets arena name from bpb_addr
28371                                   ; 13/04/2024
28372                                   ; ds = cs
28373                                   ;mov    es,[cs:DevLoadAddr]   ; (**) (InitDevload sets this)
28374 000018E4 8E06[3524]               mov     es,[DevLoadAddr]
28375
28376                                 ;   First, move a little header in place so that this looks
28377                                 ;   to the mem command like a legitimate driver load. Otherwise,
28378                                 ;   it will display garbage for the device name
28379
```

```
28380 000018E8 31FF                     xor     di,di                   ; move a little header in place
28381                                                                   ; so that this looks to the mem command
28382                                                                   ; like a legitimate driver load
28383 000018EA BE[C816]                 mov     si,tiny_stub_start
28384                                   ;mov    cx,18                   ; (tiny_stub_end-tiny_stub_start)
28385 000018ED B91200                   mov     cx,tiny_stub_end-tiny_stub_start
28386 000018F0 F3A4                     rep movsb                       ; move it!
28387 000018F2 8CC0                     mov     ax,es                   ; advance es appropriately
28388 000018F4 40                       inc     ax                      ; add ax,((tiny_stub_end-tiny_stub_start)+15)/16
28389 000018F5 40                       inc     ax
28390 000018F6 8EC0                     mov     es,ax
28391 000018F8 BBFEFF                   mov     bx,0FFFEh ; -2          ; final placement!
28392 000018FB B8114A                   mov     ax,4A11h                ; multMagicdrv
28393 000018FE CD2F                     int     2Fh                     ; DBLSPACE.BIN - RELOCATE
28394                                                                   ; es = segment to which to relocate DBLSPACE.BIN
28395                                   ;mov    ax,[cs:DevLoadAddr] ; (**)
28396                                   ;add    ax,[cs:DevSize]                 ; calculate seg after DD load
28397                                   ;mov    [cs:DevBrkAddr+2],ax  ; save as ending address!
28398                                   ;mov    word [cs:DevBrkAddr],0
28399                                   ; 13/04/2024
28400                                   ; ds = cs
28401 00001900 A1[3524]                 mov     ax,[DevLoadAddr]
28402 00001903 0306[3324]               add     ax,[DevSize]
28403 00001907 A3[3F24]                 mov     [DevBrkAddr+2],ax
28404 0000190A C706[3D24]0000           mov     word [DevBrkAddr],0
28405
28406 00001910 E8F31D                   call    DevSetBreak             ; go ahead and alloc mem for device
28407                                   ;call   DevBreak
28408                                 ;no_magic:
28409                                   ;retn
28410                                   ; 13/04/2024 - Retro DOS v5.0
28411 00001913 E9121E                   jmp     DevBreak
28412
28413                                 ; =============== S U B R O U T I N E ===================================
28414
28415                                 ; 08/04/2024 - Retro DOS v5.0
28416                                 ; (PCDOS 7.1 IBMBIO.COM - SYSINIT:1C08h)
28417
28418                                 ;***MagicSetCdss -- disable CDSs for still unmounted DblSpace drives
28419                                 ;
28420                                 ;   entry:
28421                                 ;      CDSs are now persistent and in their final place
28422
28423                                 MagicSetCdss:
28424                                     ; 13/04/2024 - Retro DOS v5.0
28425                                     ; ds = cs
28426 00001916 E83D00                   call    get_dblspace_version  ; is it there?
28427 00001919 753A                     jnz     short magic_set_exit  ; done if not
28428                                                                   ; cl = first DblSpace drive in ASCII
28429                                                                   ; ch = number of DblSpace drive letters
28430 0000191B 2EC536[6D02]             lds     si,[cs:DOSINFO]               ; point to DOS data area (SysInitVars)
28431 00001920 C57416                   lds     si,[si+16h]           ; lds si,[si+SYSI_CDS] ; fetch CDSs
28432 00001923 B458                     mov     ah,88                 ; curdirLen
28433 00001925 80E941                   sub     cl,'A'                ; make it zero based.
28434 00001928 88C8                     mov     al,cl                 ; get first DblSpace drive letter
28435 0000192A F6E4                     mul     ah                    ; find first DblSpace CDS
28436 0000192C 01C6                     add     si,ax                 ; cds pointer
28437 0000192E 88CA                     mov     dl,cl                 ; save for drive testing loop
28438 00001930 88E9                     mov     cl,ch                 ; get DblSpace drive count into cx
28439 00001932 30ED                     xor     ch,ch
28440
28441                                 ;   We know cx > 0, or else the driver wouldn't have stayed resident
28442
28443                                 magic_set_cdss_1:
28444 00001934 51                        push    cx
28445 00001935 52                       push    dx
28446 00001936 1E                       push    ds
28447 00001937 56                       push    si
28448 00001938 B8114A                   mov     ax,4A11h              ; multMagicdrv
28449 0000193B BB0100                   mov     bx,1                  ; MD_DRIVE_MAP ; inquire drive map
28450 0000193E CD2F                     int     2Fh                   ; DBLSPACE.BIN - "GetDriveMapping"
28451                                                                 ; see if this is an unused DblSpace drive
28452 00001940 5E                       pop     si
28453 00001941 1F                       pop     ds
28454 00001942 5A                       pop     dx
28455 00001943 59                       pop     cx
28456 00001944 38DA                     cmp     dl,bl                 ; if mapped to itself,it is vacant
28457 00001946 7504                     jnz     short magic_set_cdss_2; skip if used
28458 00001948 806444BF                 and     byte [si+44h],0BFh    ; Retro DOS v5.0 ; 08/04/2024
28459                                   ;and    word [si+43h],0BFFFh
28460                                                                 ; reset the bit in flags (curdir_inuse bit)
28461                                                                 ; [si+curdir_list.cdir_flags],~curdir_inuse ; word
28462                                                                 ; (.. [si+1+curdir_list.cdir_flags],0BFh ; byte)
28463                                 magic_set_cdss_2:
28464 0000194C 83C658                   add     si,88                 ; curdirLen
28465 0000194F FEC2                     inc     dl                    ; next drive
28466 00001951 E2E1                     loop    magic_set_cdss_1
28467                                   ; 13/04/2024
28468 00001953 0E                       push    cs
28469 00001954 1F                       pop     ds
28470                                 magic_set_exit:
28471 00001955 C3                       retn
28472
28473                                 ; =============== S U B R O U T I N E ===================================
28474
28475                                 ; 08/04/2024 - Retro DOS v5.0
28476                                 ; (PCDOS 7.1 IBMBIO.COM - SYSINIT:1C47h)
28477
28478                                 get_dblspace_version:
28479 00001956 B8114A                   mov     ax,4A11h        ; multMagicdrv
28480                                                           ; DBLSPACE.BIN - "GetVersion" - INSTALLATION CHECK
28481                                                           ; (BX = 0)
28482 00001959 31DB                     xor     bx,bx           ; MD_VERSION = 0
28483 0000195B CD2F                     int     2Fh             ; Return:
28484                                                           ;  AX = 0000h (successful)
28485                                                           ;  BX = 444Dh ("DM")
28486                                                           ;  CL = first drive letter used by DBLSPACE (41h = A:)
28487                                                           ;  CH = number of drive letters used by DBLSPACE
28488                                                           ;  DX = internal DBLSPACE.BIN version number (bits 14-0)
28489                                                           ;  bit 15 set if DBLSPACE.BIN has not yet been relocated
28490                                                           ;  to final position in memory (i.e. DBLSPACE.SYS /MOVE)
28491 0000195D 09C0                     or      ax,ax           ; ax = 0 (successful,zf=1)
28492 0000195F C3                       retn
28493
28494                                 ; ---------------------------------------------------------------------------
28495                                 ; SYSCONF.ASM - MSDOS 6.0 - 1991
28496                                 ; ---------------------------------------------------------------------------
28497                                 ; 27/03/2019 - Retro DOS v4.0
28498
28499                                 ;MULTI_CONFIG      equ 1
28500
28501                                 HIGH_FIRST  equ 080h            ; from ARENA.INC - modifier for
28502                                                                 ;       allocation strategy call
28503
```

```
                        ;have_install_cmd equ 00000001b   ; config.sys has install= commands
                        ;has_installed       equ 00000010b       ; sysinit_base installed.

                        default_filenum equ 8

                        ;stacksw    equ true              ; include switchable hardware stacks

                        ; external variable defined in ibmbio module for multi-track

                        ;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.

                        ; if stacksw

                        ; internal stack parameters

                        ;entrysize  equ 8

                        ;mincount   equ 8
                        ;defaultcount      equ 9
                        ;maxcount   equ 64

                        ;minsize    equ 32
                        ;defaultsize       equ 128
                        ;maxsize    equ 512

                        DOS_FLAG_OFFSET    equ 86h

                        ;ifdef MULTI_CONFIG
                        ;
                        ;   config_envlen must immediately precede config_wrkseg, because they
                        ;   may be loaded as a dword ptr

                        ; 30/12/2022 - Retro DOS v4.2 (Modified MSDOS 6.21 IO.SYS)
                        ; 25/10/2022
00001960 0000           config_envlen:     dw  0                  ; when config_wrkseg is being used as
                                                                   ;  a scratch env, this is its length
00001962 0000           config_wrkseg:     dw  0                  ; config work area (above confbot)
                                                                   ;  segment of work area

00001964 00             config_cmd: db  0                 ; current config cmd
                                                          ;  (with CONFIG_OPTION_QUERY bit intact)
00001965 00             config_multi:      db  0                  ; non-zero if multi-config config.sys

                        ;endif ; MULTI_CONFIG

00001966 00             multdeviceflag:    db  0

00001967 0000           devmark_addr:      dw  0                  ;segment address for devmark.

00001969 00             setdevmarkflag: db  0                     ;flag used for devmark

                        ; 30/12/2022
                        ; 12/12/2022
0000196A 00             driver_units:      db  0                  ;total unitcount for driver

                        ; 12/12/2022
                        ;ems_stub_installed:
                        ;           db  0

                        ; 12/12/2022
                        ;align 2

                        badparm_ptr:       ; label dword
0000196B 0000           badparm_off:       dw  0
0000196D 0000           badparm_seg:       dw  0

                        ;****************************************************************************
                        ;take care of config.sys file.
                        ;system parser data and code.
                        ;****************************************************************************

                        ;*********************************************************************
                        ; parser options set for msbio sysconf module
                        ;*********************************************************************
                        ;
                        ;**** default assemble swiches definition **************************

                        ;farsw      equ 0          ; near call expected
                        ;datesw     equ 0          ; check date format
                        ;timesw     equ 0          ; check time format
                        ;filesw     equ 1          ; check file specification
                        ;capsw      equ 0          ; perform caps if specified
                        ;cmpxsw     equ 0          ; check complex list
                        ;numsw      equ 1          ; check numeric value
                        ;keysw      equ 0          ; support keywords
                        ;swsw       equ 1          ; support switches
                        ;val1sw     equ 1          ; support value definition 1
                        ;val2sw     equ 0          ; support value definition 2
                        ;val3sw     equ 1          ; support value definition 3
                        ;drvsw      equ 1          ; support drive only format
                        ;qussw      equ 0          ; support quoted string format

                        ; psdata_seg equ cs

                            ;.xlist
                            ;include parse.asm            ;together with psdata.inc
                            ;.list

                        ; PSDATA.INC - MSDOS 6.0 - 1991
                        ; ============================================================================
                        ; 27/03/2019 - Retro DOS v4.0

                        ; 30/03/2019
                        ; VERSION.INC (MSDOS 6.0)
                        ; Set DBCS Blank constant

                        ; ifndef DBCS
                        DB_SPACE EQU 2020h
                        DB_SP_HI EQU 20h
                        DB_SP_LO EQU 20h
                        ; else

                        ;*********************************************************************
                        ;
                        ; Parser include file
                        ;*********************************************************************

                        ;**** Equation field
                        ;-------- Character code definition

                        _$P_DBSP1      equ  DB_SP_HI       ;AN000; 1st byte of DBCS blank
```

```
                                _$P_DBSP2       equ DB_SP_LO        ;AN000; 2nd byte of DBCS blank
                                _$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
                        ;                               ; 0002H=REPEATS ALLOWED
                        ;                               ; 0001H=OPTIONAL
                        ;       DW      FUNCTION_FLAGS
                        ;                               ; 0001H=CAP RESULT BY FILE TABLE
                        ;                               ; 0002H=CAP RESULT BY CHAR TABLE
                        ;                               ; 0010H=REMOVE ":" AT END
                        ; (tm10)                        ; 0020H=colon is not necessary for switch
                        ;       DW      RESULT          ; RESULT BUFFER
                        ;       DW      VALUES          ; VALUE LISTS
                        ;       DB      nid             ; NUMBER OF KEYWORD/SWITCH SYNONYMS IN FOLLOWING LIST
                        ;       DB      "...",0          ; IF n >0, KEYWORD 1
                        ;       :
                        ;
                        ;Note:
                        ;    - The MATCH_FLAG is bit significant. You can set, for example, TIME bit and
                        ;      DATE bit simalteniously.
                        ;
                        ;      The parser examins each bit along with the following priority.
                        ;
                        ;      COMPLEX -> DATE -> TIME -> NUMERIC VAL -> SIGNED NUMERIC VAL -> DRIVE ->
                        ;      FILE SPEC -> SIMPLE STRING.
                        ;
                        ;    - When the FUNCTION_FLAG is 0001 or 0002, the STRING pointed to by a pointer
                        ;      in the result buffer is capitalized.
                        ;
                        ;    - Match_Flags 0001H and 0002H have meaning only for the positional.
                        ;
                        ;    - The "...",0 (bottom most line) does require '=' or '/'. When you need a
                        ;      switch, for example, '/A', then STRING points to;
                        ;
                        ;                   DB    1         ; number of following synonyms
                        ;                   DB   '/A',0
                        ;
                        ;      When you need a keyword, for example, 'CODEPAGE=', then "...",0 will be;
                        ;
                        ;                   DB    1         ; number of following synonyms
                        ;                   DB   'CODEPAGE=',0
                        ;
                        ;    - "..." must consist of upper case characters only because the parser
                        ;      performs pattern matching after converting input to upper case (by
                        ;      using the current country upper case table)
                        ;
                        ;    - One "..." can contain only one switch or keyword. If you need, for
                        ;      example /A and /B, the format will be;
                        ;
                        ;                   DB    2         ; number of following synonyms
                        ;                   DB    '/A',0
                        ;                   DB    '/B',0
                        ; ---------------------------------------------------------------------------

                        ;**** Match_Flags

                        _$P_Num_Val    equ  8000h          ;AN000; Numeric Value
                        _$P_SNum_Val          equ  4000h          ;AN000; Signed numeric value
                        _$P_Simple_S          equ  2000h          ;AN000; Simple string
                        _$P_Date_S     equ  1000h          ;AN000; Date string
                        _$P_Time_S     equ  0800h          ;AN000; Time string
                        _$P_Cmpx_S     equ  0400h          ;AN000; Complex string
                        _$P_File_Spc          equ  0200h          ;AN000; File Spec
                        _$P_Drv_Only          equ  0100h          ;AN000; Drive Only
                        _$P_Qu_String         equ  0080h          ;AN000; Quoted string
                        _$P_Ig_Colon          equ  0010h          ;AN000; Ignore colon at end in match
                        _$P_Repeat     equ  0002h          ;AN000; Repeat allowed
                        _$P_Optional          equ  0001h          ;AN000; Optional

                        ;**** Function flags

                        _$P_CAP_File          equ  0001h          ;AN000; CAP result by file table
                        _$P_CAP_Char          equ  0002h          ;AN000; CAP result by character table
                        _$P_Rm_Colon          equ  0010h          ;AN000; Remove ":" at the end
                        _$P_colon_is_not_necessary equ 0020h     ;AN000;(tm10) /+10 and /+:10

                        ;-------------------------------- Control block structure
                        struc _$P_Control_Blk
00000000 ????            .Match_Flag:       resw 1        ;AN000; Controls type matched
00000002 ????            .Function_Flag: resw 1           ;AN000; Function should be taken
00000004 ????            .Result_Buf:       resw 1        ; Result buffer address
00000006 ????            .Value_List:       resw 1        ;AN000; Value list address
00000008 ??              .nid:              resb 1        ;AN000; # of keyword/SW synonyms
00000009 ??              .KEYorSW:   resb 1       ;AN000; keyword or sw
                        endstruc

                        ; ---------------------------------------------------------------------------
                        ; << Value List Definition >>
                        ;
                        ;VALUES     LABEL   BYTE
                        ;           DB      nval                ; NUMBER OF VALUE DEFINITIONS (0 - 3)
                        ;         +-
                        ;         | DB      nrng                ; NUMBER OF RANGES
                        ;         | +DB     ITEM_TAG            ; RETURN VALUE IF RANGE MATCHED
                        ;         | +DD     X,Y                 ; RANGE OF VALUES
                        ;         | :
                        ;         | DB      nnval               ; NUMBER OF CHOICES
                        ;         | +DB     ITEM_TAG            ; RETURN VALUE IF NUMBER CHOICE MATCHED
                        ;         | +DD     VALUE               ; SPECIFIC CHOICE IF NUMBER
                        ;         | :
                        ;         | DB      nstrval             ; NUMBER OF CHOICES
                        ;         | +DB     ITEM_TAG            ; RETURN VALUE IF STRING CHOICE MATCHED
                        ;         | +DW     STRING              ; SPECIFIC CHOICE IF STING
                        ;         +- :
                        ;
                        ;STRING     DB      "...",0               ; ASCIIZ STRING IMAGE
                        ;
                        ;Note:
                        ;    - ITEM_TAG must not be 0FFH, which will be used in the result buffer
                        ;      when no choice lists are provided.
                        ;
                        ;    - STRING must consist of upper case characters only because the parser
                        ;      performs pattern matching after converting input to upper case (by
                        ;      using the current country upper case table)
                        ; ---------------------------------------------------------------------------

                        _$P_nval_None         equ  0            ;AN000; no value list ID
                        _$P_nval_Range        equ  1            ;AN000; range list ID
                        _$P_nval_Value        equ  2            ;AN000; value list ID
                        _$P_nval_String       equ  3            ;AN000; string list ID
                        _$P_Len_Range         equ  9            ;AN000; Length of a range choice(two DD plus one DB)
                        _$P_Len_Value         equ  5            ;AN000; Length of a value choice(one DD plus one DB)
                        _$P_Len_String        equ  3            ;AN000; Length of a string choice(one DW plus one DB)
                        _$P_No_nrng    equ  0            ;AN000; (tm07) no nrng. nnval must not be 0.

                        struc _$P_Val_List
```

```
28876 00000000 ??                       .NumofList: resb 1                      ;AN000; number of following choice
28877 00000001 ????                     .Val_XL:    resw 1                      ;AN000; lower word of value
28878 00000003 ????                     .Val_XH:    resw 1                      ;AN000; higher word of value
28879 00000005 ????                     .Val_YL:    resw 1                      ;AN000; lower word of another value
28880 00000007 ????                     .Val_YH:    resw 1                      ;AN000; higher word of another value
28881                                   endstruc
28882
28883                                   ; ---------------------------------------------------------------------------
28884                                   ; << Result Buffer Definition  >>
28885                                   ;
28886                                   ;RESULT     LABEL   BYTE                    ; BELOW FILLED IN FOR DEFAULTS
28887                                   ;           DB      type                    ; TYPE RETURNED: 0=RESERVED,
28888                                   ;                                           ;       1=NUMBER, 2=LIST INDEX,
28889                                   ;                                           ;       3=STRING, 4=COMPLEX,
28890                                   ;                                           ;       5=FILESPEC, 6=DRIVE
28891                                   ;                                           ;       7=DATE, 8=TIME
28892                                   ;                                           ;       9=QUOTED STRING
28893                                   ;           DB      ITEM_TAG                ; MATCHED ITEM TAG
28894                                   ;
28895                                   ;           dw      synonym@                ; es:@ points to found SYNONYM if provided.
28896                                   ;
28897                                   ;             +-
28898                                   ;         | DD      n                       ; VALUE IF NUMBER
28899                                   ;         | or
28900                                   ;         |  DW     i                       ; INDEX (OFFSET) INTO VALUE LIST
28901                                   ;         |                                 ; (ES presents Segment address)
28902                                   ;         | or
28903                                   ;         |  DD     STRING                  ; OFFSET OF STRING VALUE
28904                                   ;         | or
28905                                   ;         |  DB     drv                     ; DRIVE NUMBER (1-A, 2-B,..., 26-Z)
28906                                   ;         | or
28907                                   ;         |  DW     YEAR      ;(1980-2099)  IN CASE OF DATE
28908                                   ;         |  DB     MONTH     ;(1-12)      Note: Range check is not performed.
28909                                   ;         |  DB     DATE      ;(1-31)            0 is filled when the corresponding field was not specified.
28910                                   ;         | or
28911                                   ;         |  DB     HOUR      ;(0-23)       IN CASE OF TIME
28912                                   ;         |  DB     MINUTES    ;(0-59)      Note: Range check is not performed .
28913                                   ;         |  DB     SECONDS    ;(0-59)           0 is filled when the corresponding field was not specified .
28914                                   ;         |  DB     HUNDREDTHS ;(0-99)
28915                                   ;          +-
28916                                   ;
28917                                   ;
28918                                   ;Note: ITEM_TAG is 0FFH when the caller does not specify the choice
28919                                   ;      list.
28920                                   ;
28921                                   ;      YEAR: If the input value for the year is less than 100, parser
28922                                   ;        adds 1900 to it.  For example, when 87 is input to parser for
28923                                   ;        the year value, he returns 1987.
28924                                   ; ---------------------------------------------------------------------------
28925
28926                                   ;-------------------------------- Result block structure
28927                                   struc _$P_Result_Blk
28928 00000000 ??                       .Type:        resb 1            ;AN000; Type returned
28929 00000001 ??                       .Item_Tag:    resb 1            ;AN000; Matched item tag
28930 00000002 ????                     .SYNONYM_Ptr: resw 1            ;AN000; pointer to Synonym list returned
28931 00000004 ????????                 .Picked_Val:  resb 4            ;AN000; value
28932                                   endstruc
28933
28934                                   ;--------------------------------
28935                                   ;**** values for the type field in the result block
28936
28937                                   _$P_EOL               equ 0             ;AN000; End of line
28938                                   _$P_Number     equ 1                ;AN000; Number
28939                                   _$P_List_Idx          equ 2             ;AN000; List Index
28940                                   _$P_String     equ 3                ;AN000; String
28941                                   _$P_Complex    equ 4                ;AN000; Complex
28942                                   _$P_File_Spec         equ 5             ;AN000; File Spec
28943                                   _$P_Drive      equ 6                ;AN000; Drive
28944                                   _$P_Date_F     equ 7                ;AN000; Date
28945                                   _$P_Time_F     equ 8                ;AN000; Time
28946                                   _$P_Quoted_String  equ    9             ;AN000; Quoted String
28947
28948                                   _$P_No_Tag     equ 0FFh             ;AN000; No ITEM_TAG found
28949
28950                                   ;**** Return code
28951                                   ;
28952                                   ; following return code will be returned in the AX register.
28953
28954                                   _$P_No_Error          equ 0             ;AN000; No error
28955                                   _$P_Too_Many          equ 1             ;AN000; Too many operands
28956                                   _$P_Op_Missing        equ 2             ;AN000; Required operand missing
28957                                   _$P_Not_In_SW         equ 3             ;AN000; Not in switch list provided
28958                                   _$P_Not_In_Key        equ 4             ;AN000; Not in keyword list provided
28959                                   _$P_Out_Of_Range   equ    6             ;AN000; Out of range specified
28960                                   _$P_Not_In_Val        equ 7             ;AN000; Not in value list provided
28961                                   _$P_Not_In_Str        equ 8             ;AN000; Not in string list provided
28962                                   _$P_Syntax     equ 9                ;AN000; Syntax error
28963                                   _$P_RC_EOL     equ -1               ;AN000; End of command line
28964
28965                                   ; DATA - Retro DOS v4.0 - 27/03/2019
28966
28967                                   ; MSDOS 6.2 IO.SYS SYSINIT:179Ch
28968
28969                                   ; 14/04/2024 - Retro DOS v5.0
28970                                   ; PCDOS 7.1 IBMBIO.COM - SYSINIT:1C62h
28971
28972                                   ;********************** Local Data ***********************************
28973 0000196F 0000                     _$P_ORDINAL:          dw  0             ;AN000; Operand ordinal save area
28974 00001971 0000                     _$P_RC:               dw  0             ;AN000; Return code from parser
28975 00001973 0000                     _$P_SI_Save:          dw  0             ;AN000; Pointer of command buffer
28976 00001975 0000                     _$P_DX:               dw  0             ;AN000; Return result buffer address
28977 00001977 00                       _$P_Terminator:       db  0             ;AN000; Terminator code (ASCII)
28978 00001978 0000                     _$P_DBCSEV_OFF:       dw  0             ;AN000; Offset of DBCS EV
28979 0000197A 0000                     _$P_DBCSEV_SEG:       dw  0             ;AN000; Segment of DBCS EV
28980 0000197C 0000                     _$P_Flags:    dw  0             ;AN000; Parser internal flags
28981                                   %define _$P_Flags1 _$P_Flags            ;AN038; to reference first byte flags
28982                                   %define _$P_Flags2 _$P_Flags+1          ;AN038; to reference second byte flags only
28983
28984                                   ;in second byte of _$P_Flags, referenced as _$P_Flags2:
28985                                   _$P_equ               equ 01h         ;AN000; "=" packed in string buffet
28986                                   _$P_Neg               equ 02h         ;AN000; Negative value
28987                                   _$P_Time12     equ 04h          ;AN000; set when PM is specified
28988                                   _$P_Key_Cmp    equ 08h          ;AN000; set when keyword compare
28989                                   _$P_SW_Cmp     equ 10h          ;AN000; set when switch compare
28990                                   _$P_Extra      equ 20h          ;AN000; set when extra delimiter found
28991                                   _$P_SW                equ 40h         ;AN000; set when switch found (tm08)
28992                                   _$P_Signed     equ 80h          ;AN000; signed numeric specified
28993
28994                                   ;in first byte of _$P_Flags, referenced as _$P_Flags1:
28995                                   _$P_time12am          equ 01h         ;AN038; set when AM is specified on time
28996                                   _$P_TIME_AGAIN        equ 02h         ;AN039; SET WHEN READY TO RE-PARSE TIME
28997
28998 0000197E 0000                     _$P_SaveSI_Cmpx:   dw     0            ;AN000; save si for later use by complex
28999 00001980 0000                     _$P_KEYorSW_Ptr:   dw     0            ;AN000; points next to "=" or ":" code
```

```
00001982 0000                   _$P_Save_EOB:         dw  0             ;AN000; save pointer to EOB
00001984 0000                   _$P_Found_SYNONYM: dw     0             ;AN000; es:@ points to found synonym

00001986 00<rep 80h>            _$P_STRING_BUF:       times 128 db 0    ;AN000; Pick a operand from command line
                                _$P_STRING_BUF_END equ    $             ;AN000;

                                ; 25/10/2022
                                ; (MSDOS 5.0 IO.SYS, SYSINIT:16F8h)

00001A06 FF                     _$P_Char_CAP_Ptr:  db     0FFh          ;AN000; info id
00001A07 0000                                      dw   0              ;AN000; offset of char case map table
00001A09 0000                                      dw   0              ;AN000; segment of char case map table
                                ; 25/10/2022
                                ;IF CAPSW
                                ;_$P_File_CAP_Ptr: db     0FFh          ;AN000; info id
                                ;                  dw   0              ;AN000; offset of file case map table
                                ;                  dw   0              ;AN000; segment of file case map table
                                ;ENDIF

                                ; (tm06) IF FileSW               ;AN000;(Check if file spec is supported)
                                ;

                                ;M029
                                ;!!!WARNING!!!
                                ; In routine SYSPARSE (parse.asm), _$P_FileSp_Char is reinitialized using
                                ;hardcoded strings. If the chars in the string are changed here, corresponding
                                ;changes need to be made in SYSPARSE

                                ;IF FileSW+DrvSW                 ;AN000;(Check if file spec is supported)

                                ; 25/10/2022
                                ; (MSDOS 5.0 IO.SYS, SYSINIT:16FDh)

00001A0B 5B5D7C3C3E2B3D3B22     _$P_FileSp_Char       db  '[]|<>+=;"'     ;AN000; delimitter of file spec
                                _$P_FileSp_Len        equ $-_$P_FileSp_Char ;AN000;

                                ;ENDIF                                  ;AN000;(of FileSW)

                                ; delimiter parsing
                                _$P_colon_period   equ    01h           ;AN032; check for colon & period
                                _$P_period_only       equ 02h           ;AN032; check only for period

                                ;filespec error flag
00001A14 00                     _$P_err_flag:         db  0             ;AN033; flag set if filespec parsing error
                                                                        ;AN033;  was detected.
                                _$P_error_filespec equ    01h           ;AN033; mask to set flag


                                ; PARSE.ASM - MSDOS 6.0 - 1991
                                ; ============================================================================
                                ; 27/03/2019 - Retro DOS v4.0

                                ;****************************************************************************
                                ; SysParse;
                                ;
                                ;  Function : Parser Entry
                                ;
                                ;  Input: DS:SI -> command line
                                ;     ES:DI -> parameter block
                                ;     cs -> psdata.inc
                                ;     CX = operand ordinal
                                ;
                                ;     Note:  ES is the segment containing all the control blocks defined
                                ;            by the caller, except for the DOS COMMAND line parms, which
                                ;            is in DS.
                                ;
                                ;  Output: CY = 1   error of caller, means invalid parameter block or
                                ;               invalid value list. But this parser does NOT implement
                                ;               this feature. Therefore CY always zero.
                                ;
                                ;      CY = 0   AX = return code
                                ;               BL = terminated delimiter code
                                ;               CX = new operand ordinal
                                ;               SI = set past scaned operand
                                ;               DX = selected result buffer
                                ;
                                ; Use:      _$P_Skip_Delim, _$P_Chk_EOL, _$P_Chk_Delim, _$P_Chk_DBCS
                                ;   _$P_Chk_Swtch, _$P_Chk_Pos_Control, _$P_Chk_Key_Control
                                ;   _$P_Chk_Sw_Control, _$P_Fill_Result
                                ;
                                ; Vars: _$P_Ordinal(RW), _$P_RC(RW), _$P_SI_Save(RW), _$P_DX(R), _$P_Terminator(R)
                                ;   _$P_SaveSI_Cmpx(W), _$P_Flags(RW), _$P_Found_SYNONYM(R), _$P_Save_EOB(W)
                                ;
                                ;-------- Modification History ----------------------------------------
                                ;
                                ;  4/04/87 : Created by K. K,
                                ;  4/28/87 : _$P_Val_YH assemble error (tm01)
                                ;       : JMP SHORT assemble error (tm02)
                                ;  5/14/87 : Someone doesn't want to include psdata (tm03)
                                ;  6/12/87 : _$P_Bridge is missing when TimeSw equ 0 and (CmpxSw equ 1 or
                                ;        DateSW equ 1)          (tm04)
                                ;  6/12/87 : _$P_SorD_Quote is missing when QusSw equ 0 and CmpxSW equ 1
                                ;                               (tm05) in PSDATA.INC
                                ;  6/12/87 : _$P_FileSp_Char and _$P_FileSP_Len are missing
                                ;        when FileSW equ 0 and DrvSW equ 1 (tm06) in PSDATA.INC
                                ;  6/18/87 : $VAL1 and $VAL3, $VAL2 and $VAL3 can be used in the same
                                ;        value-list block       (tm07)
                                ;  6/20/87 : Add _$P_SW to check if there's an omiting parameter after
                                ;        switch (keyword) or not. If there is, backup si for next call
                                ;        (tm08)
                                ;  6/24/87 : Complex Item checking does not work correctly when CmpSW equ 1
                                ;        and DateSW equ 0 and TimeSW equ 0 (tm09)
                                ;  6/24/87 : New function flag _$P_colon_is_not_necessary for switch
                                ;        /+15 and /+:15 are allowed for user (tm10)
                                ;  6/29/87 : ECS call changes DS register but it causes the address problem
                                ;        in user's routines. _$P_Chk_DBCS (tm11)
                                ;  7/10/87 : Switch with no_match flag (0x0000H) does not work correctly
                                ;                               (tm12)
                                ;  7/10/87 : Invalid switch/keyword does not work correctly
                                ;                               (tm13)
                                ;  7/10/87 : Drive_only breaks 3 bytes after the result buffer
                                ;                               (tm14)
                                ;  7/12/87 : Too_Many_Operands sets DX=0 as the PARSE result
                                ;                               (tm15)
                                ;  7/24/87 : Negative lower bound on numeric ranges cause trouble
                                ;
                                ;  7/24/87 : Quoted strings being returned with quotes.
                                ;
                                ;  7/28/87 : Kerry S (;AN018;)
                                ;        Non optional value on switch (match flags<>0 and <>1) not flagged
                                ;        as an error when missing.       Solution: return error 2.  Modules
                                ;        affected: _$P_Chk_SW_Control.
                                ;
                                ;  7/29/87 : Kerry S (;AN019;)
```

```
29124                           ;       Now allow the optional bit in match flags for switches.  This
29125                           ;       allows the switch to be encountered with a value or without a
29126                           ;       value and no error is returned.
29127                           ;
29128
29129                           ;  8/28/87 : Ed K, Kerry S (;AN020;)
29130                           ;  9/14/87   In PROC _$P_Get_DecNum, when checking for field separators
29131                           ;       within a date response, instead of checking just for the one
29132                           ;       character defined by the COUNTRY DEPENDENT INFO, check for
29133                           ;       all three chars, "-", "/", and ".". Change _$P_Chk_Switch to allow
29134                           ;       slashes in date strings when DateSw (assembler switch) is set.
29135
29136                           ;  9/1/87  : Kerry S (;AN021)
29137                           ;       In PROC _$P_String_Comp, when comparing the switch or keyword on
29138                           ;       the command line with the string in the control block the
29139                           ;       comparing was stopping at a colon (switch) or equal (keyword)
29140                           ;       on the command line and assuming a match.      This allowed a shorter
29141                           ;       string on the command line than in the synonym list in the control
29142                           ;       block.  I put in a test for a null in the control block so the
29143                           ;       string in the control block must be the same length as the string
29144                           ;       preceeding the colon or equal on the command line.
29145
29146                           ;  8/28/87 : Kerry S (;AN022;)
29147                           ;       All references to data in PSDATA.INC had CS overrides.  This caused
29148                           ;       problems for people who included it themselves in a segment other
29149                           ;       than CS.  Added switch to allow including PSDATA.INC in any
29150                           ;       segment.
29151
29152                           ;  9/16/87 : Ed K (;AN023;) PTM1040
29153                           ;       in _$P_set_cdi PROC, it assumes CS points to psdata. Change Push CS
29154                           ;       into PUSH cs.  In _$P_Get_DecNum PROC, fix AN020
29155                           ;       forced both TIME and DATE to use the delims, "-","/",".".
29156                           ;       Created FLag, in _$P_time_Format PROC, to request the delim in
29157                           ;       BL be used if TIME is being parsed.
29158
29159                           ;  9/24/87 : Ed K
29160                           ;       Removed the include to STRUC.INC.      Replaced the STRUC macro
29161                           ;       invocations with their normally expanded code; made comments
29162                           ;       out of the STRUC macro invocation statements to maintain readability.
29163
29164                           ;  9/24/87 : Ed K (;AN024;) PTM1222
29165                           ;       When no CONTROL for a keyword found, tried to fill in RESULT
29166                           ;       pointed to by non-existant CONTROL.
29167
29168                           ; 10/15/87 : Ed K (;AN025;) PTM1672
29169                           ;       A quoted text string can be framed only by double quote.  Remove
29170                           ;       support to frame quoted text string with single quote.
29171                           ;       (apostrophe) _$P_SorD_Quote is removed from PSDATA.INC.
29172                           ;       _$P_SQuote EQU also removed from PSDATA.INC.  Any references to
29173                           ;       single quote in PROC prologues are left as is for history reasons.
29174
29175                           ;       This fixes another bug, not mentioned in p1672, in that two
29176                           ;       quote chars within a quoted string is supposed to be reported as
29177                           ;       one quote character, but is reported as two quotes.  This changed
29178                           ;       two instructions in PROC _$P_Quoted_Str.
29179
29180                           ;       Also fixed are several JMP that caused a NOP, these changed to
29181                           ;       have the SHORT operator to avoid the unneeded NOP.
29182
29183                           ;       The code and PSDATA.INC have been aligned for ease of reading.
29184
29185                           ; 10/26/87 : Ed K (;AN026;) PTM2041, DATE within SWITCH, BX reference to
29186                           ;       psdata buffer should have cs.
29187
29188                           ; 10/27/87 : Ed K (;AN027;) PTM2042 comma between keywords implies
29189                           ;       positional missing.
29190
29191                           ; 11/06/87 : Ed K (;AN028;) PTM 2315 Parser should not use line feed
29192                           ;       as a line delimiter, should use carriage return.
29193                           ;       Define switch: LFEOLSW, if on, accept LF as end of line char.
29194
29195                           ; 11/11/87 : Ed K (;AN029;) PTM 1651 GET RID OF WHITESPACE AROUND "=".
29196
29197                           ; 11/18/87 : Ed K (;AN030;) PTM 2551 If filename is just "", then
29198                           ;       endless loop since SI is returned still pointing to start
29199                           ;       of that parm.
29200
29201                           ; 11/19/87 : Ed K (;AN031;) PTM 2585 date & time getting bad values.
29202                           ;       Vector to returned string has CS instead of cs, but
29203                           ;       when tried to fix it on previous version, changed similar
29204                           ;       but wrong place.
29205
29206                           ; 12/09/87 : Bill L (;AN032;) PTM 2772 colon and period are now valid
29207                           ;       delimiters between hours, minutes, seconds for time. And period
29208                           ;       and comma are valid delimiters between seconds and 100th second.
29209
29210                           ; 12/14/87 : Bill L (;AN033;) PTM 2722 if illegal delimiter characters
29211                           ;       in a filespec, then flag an error.
29212
29213                           ; 12/22/87 : Bill L (;AN034;)         All local data to parser is now
29214                           ;       indexed off of the cs equate instead of the DS register.
29215                           ;       Using this method, DS can point to the segment of PSP or to psdata
29216                           ;  -->          local parser data. Why were some references to local data changed
29217                           ;       to do this before, but not all ?????
29218
29219                           ; 02/02/88 : Ed K (;AC035;) INSPECT utility, suggests optimizations.
29220
29221                           ; 02/05/88 : Ed K (;AN036;) P3372-UPPERCASE TRANSLATION, cs HOSED.
29222                           ;
29223                           ; 02/08/88 : Ed K (;AN037;) P3410-AVOID POP OF CS, CHECK BASESW FIRST.
29224
29225                           ; 02/19/88 : Ed K (;AN038;) p3524 above noon and "am" should be error
29226
29227                           ; 02/23/88 : Ed K (;AN039;) p3518 accept "comma" and "period" as decimal
29228                           ;       separator in TIME before hundredths field.
29229                           ;
29230                           ; 08/09/90 : SA    M005    Prevented parser from recognizing '=' signs within
29231                           ;                   strings as keywords.
29232                           ;
29233                           ;*****************************************************************************
29234
29235                           ;IF FarSW                          ;AN000;(Check if need far return)
29236                           ;SysParse proc far                 ;AN000;
29237                           ;ELSE                                      ;AN000;
29238                           ;SysParse proc near                        ;AN000;
29239                           ;ENDIF                                     ;AN000;(of FarSW)
29240
29241                           ; 27/03/2019 - Retro DOS v4.0
29242                           ; (MSDOS 6.21 IO.SYS - SYSINIT:1842h)
29243
29244                           ; 25/10/2022 - Retro DOS v4.0
29245                           ; (MSDOS 5.0 IO.SYS - SYSINIT:1707h)
29246
29247                           ; 06/09/2023 - Retro DOS v4.2 IO.SYS Optimization (& Retro DOS v5.0 pre-work)
```

```
                                ; (PCDOS 7.1 IBMBIO.COM - SYSINIT:1D08h)

                                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
```

```
29371 00001AB1 EBBE                     jmp     short _$P_Pack_Loop   ;AN000; process to next byte
29372
29373                                 _$P_Pack_End_BY_EOL:                    ;AN000;
29374 00001AB3 4E                       dec     si                    ;AN000; backup si pointer
29375                                 _$P_Pack_End:                           ;AN000;
29376 00001AB4 2E8936[7319]             mov     [cs:_$P_SI_Save],si     ;AC034; save next pointer, SI
29377                                   ; 07/09/2023
29378                                   ;mov    byte [cs:bx],_$P_NULL ;AN000; put NULL at the end
29379 00001AB9 30E4                     xor     ah,ah ; 0 ; *
29380 00001ABB 2E8827                   mov     [cs:bx],ah ; _$P_NULL ;AN000; put NULL at the end
29381                                   ;
29382 00001ABE 2E891E[8219]             mov     [cs:_$P_Save_EOB],bx    ;AC034; 3/17/87 keep the address for later use of complex
29383                                   ;mov    bx,[es:di+_$P_PARMS_Blk.PARMSX_Address] ;AN000; get PARMSX address
29384 00001AC3 268B1D                   mov     bx,[es:di]
29385                                   ;;lea   si,[cs:_$P_STRING_BUF];AC034;
29386                                   ; 02/11/2022
29387                                   ;lea    si,[_$P_STRING_BUF]
29388                                   ; 07/09/2023
29389 00001AC6 BE[8619]                 mov     si,_$P_STRING_BUF
29390 00001AC9 2E803C2F                 cmp     byte [cs:si],_$P_Switch ;AN000; the operand begins w/ switch char ?
29391 00001ACD 7440                     je      short _$P_SW_Manager  ;AN000; if yes, process as switch
29392
29393 00001ACF 2E803C22                 cmp     byte [cs:si],_$P_DQuote       ;M005;is it a string?
29394 00001AD3 7408                     je      short _$P_Positional_Manager ;M005;if so, process as one!
29395
29396 00001AD5 2EF606[7D19]01           test    byte [cs:_$P_Flags2],_$P_equ ;AC034; the operand includes equal char ?
29397 00001ADB 7552                     jnz     short _$P_Key_Manager ;AN000; if yes, process as keyword
29398
29399                                 _$P_Positional_Manager:                 ;AN000; else process as positional
29400 00001ADD 268A4701                 mov     al,[es:bx+_$P_PARMSX_Blk.MaxP] ;AN000; get maxp
29401                                   ; 07/09/2023
29402                                   ;xor    ah,ah                 ;AN000; ax = maxp
29403 00001AE1 2E3906[6F19]             cmp     [cs:_$P_ORDINAL],ax   ;AC034; too many positional ?
29404 00001AE6 7312                     jae     short _$P_Too_Many_Error ;AN000; if yes, set exit code to too many
29405
29406 00001AE8 2EA1[6F19]               mov     ax,[cs:_$P_ORDINAL]   ;AC034; see what the current ordinal
29407 00001AEC D1E0                     shl     ax,1                  ;AN000; ax = ax*2
29408 00001AEE 43                       inc     bx                    ;AC035; add '2' to
29409 00001AEF 43                       inc     bx                    ;AC035;  BX reg
29410                                                                 ;AN000; now bx points to 1st CONTROL
29411 00001AF0 01C3                     add     bx,ax                 ;AN000; now bx points to specified CONTROL address
29412 00001AF2 268B1F                   mov     bx,[es:bx]            ;AN000; now bx points to specified CONTROL itself
29413 00001AF5 E87200                   call    _$P_Chk_Pos_Control   ;AN000; Do process for positional
29414 00001AF8 EB53                     jmp     short _$P_Return_to_Caller ;AN000; and return to the caller
29415
29416                                 _$P_Too_Many_Error:                     ;AN000;
29417 00001AFA 2EC706[7119]0100         mov     word [cs:_$P_RC],_$P_Too_Many ;AC034; set exit code
29418 00001B01 EB4A                     jmp     short _$P_Return_to_Caller ;AN000; and return to the caller
29419
29420                                   ; 07/09/2023 - Retro DOSD v4.2 IO.SYS (Optimization)
29421                                   ; (PCDOS 7.1 IBMBIO.COM - SYSINIT:1E06h)
29422                                 get_maxp:
29423                                   ;mov    al,[es:bx+1]
29424 00001B03 268A4701                 mov     al,[es:bx+_$P_PARMSX_Blk.MaxP] ;AN000; get maxp
29425                                   ; 07/09/2023
29426                                   ; ah=0 ; *
29427                                   ;xor    ah,ah ; 0             ;AN000; ax = maxp
29428 00001B07 30ED                     xor     ch,ch ; **
29429 00001B09 40                       inc     ax                    ;AN000;
29430 00001B0A D1E0                     shl     ax,1                  ;AN000; ax = (ax+1)*2
29431 00001B0C 01C3                     add     bx,ax                 ;AN000; now bx points to maxs
29432 00001B0E C3                       retn
29433
29434                                 _$P_SW_Manager:                         ;AN000;
29435                                   ; 07/09/2023
29436                                   ;mov    al,[es:bx+_$P_PARMSX_Blk.MaxP] ;AN000; get maxp
29437                                   ;xor    ah,ah                 ;AN000; ax = maxp
29438                                   ;inc    ax                    ;AN000;
29439                                   ;shl    ax,1                  ;AN000; ax = (ax+1)*2
29440                                   ;add    bx,ax                 ;AN000; now bx points to maxs
29441 00001B0F E8F1FF                   call    get_maxp ; 07/09/2023
29442
29443 00001B12 268A0F                   mov     cl,[es:bx]            ;AN000;
29444                                   ; 07/09/2023
29445                                   ;xor    ch,ch ; ** (ch=0)     ;AN000; cx = maxs
29446                                   ;or     cx,cx                 ;AN000; at least one switch ?
29447                                   ;jz     short _$P_SW_Not_Found        ;AN000;
29448                                   ; 07/07/2023
29449 00001B15 E30F                     jcxz    _$P_SW_Not_Found      ; no
29450
29451 00001B17 43                       inc     bx                    ;AN000; now bx points to 1st CONTROL address
29452
29453                                 _$P_SW_Mgr_Loop:                        ;AN000;
29454 00001B18 53                       push    bx                    ;AN000;
29455 00001B19 268B1F                   mov     bx,[es:bx]            ;AN000; bx points to Switch CONTROL itself
29456 00001B1C E8A900                   call    _$P_Chk_SW_Control    ;AN000; do process for switch
29457 00001B1F 5B                       pop     bx                    ;AN000;
29458 00001B20 732B                     jnc     short _$P_Return_to_Caller ;AN000; if the CONTROL is for the switch, exit
29459
29460 00001B22 43                       inc     bx                    ;AC035; add '2' to
29461 00001B23 43                       inc     bx                    ;AC035;  BX reg
29462                                                                 ;AN000; else bx points to the next CONTROL
29463 00001B24 E2F2                     loop    _$P_SW_Mgr_Loop               ;AN000; and loop
29464
29465                                 _$P_SW_Not_Found:                       ;AN000;
29466 00001B26 2EC706[7119]0300         mov     word [cs:_$P_RC],_$P_Not_In_SW ;AC034; here no CONTROL for the switch has
29467 00001B2D EB1E                     jmp     short _$P_Return_to_Caller ;AN000; not been found, means error.
29468
29469                                 _$P_Key_Manager:                        ;AN000;
29470                                   ; 07/09/2023
29471                                   ;mov    al,[es:bx+_$P_PARMSX_Blk.MaxP] ;AN000; get maxp
29472                                   ;xor    ah,ah                 ;AN000; ax = maxp
29473                                   ;inc    ax                    ;AN000;
29474                                   ;shl    ax,1                  ;AN000; ax = (ax+1)*2
29475                                   ;add    bx,ax                 ;AN000; now bx points to maxs
29476 00001B2F E8D1FF                   call    get_maxp ; 07/09/2023
29477
29478 00001B32 268A07                   mov     al,[es:bx]            ;AN000;
29479 00001B35 30E4                     xor     ah,ah ; 0             ;AN000; ax = maxs
29480 00001B37 D1E0                     shl     ax,1                  ;AN000;
29481 00001B39 40                       inc     ax                    ;AN000; ax = ax*2+1
29482 00001B3A 01C3                     add     bx,ax                 ;AN000; now bx points to maxk
29483 00001B3C 268A0F                   mov     cl,[es:bx]            ;AN000;
29484                                   ; 07/09/2023
29485                                   ;xor    ch,ch ; ** (ch=0)     ;AN000; cx = maxk
29486                                   ;or     cx,cx                 ;AN000; at least one keyword ?
29487                                   ;jz     short _$P_Key_Not_Found       ;AN000;
29488                                   ; 07/07/2023
29489 00001B3F E305                     jcxz    _$P_Key_Not_Found     ; no
29490
29491 00001B41 43                       inc     bx                    ;AN000; now bx points to 1st CONTROL
29492
29493                                 _$P_Key_Mgr_Loop:                       ;AN000;
29494                                   ; 07/09/2023
```

```
                                    ; ('_$P_Chk_Key_Control' contains only 'stc' instruction)
                                    ; (always returns with cf=1)
                                    ;push   bx                      ;AN000;
                                    ;mov    bx,[es:bx]              ;AN000; bx points to keyword CONTROL itself
                                    ;call   _$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
00001C21 E80300                     call    _$P_Fill_Result             ;AN000; set result buffer
00001C24 58                         pop     ax                          ;AN000;
00001C25 F8                         clc                                 ;AN000;
                                _$P_Chk_SW_Single_Exit:                 ;AN000;
00001C26 C3                         retn                                ;AN000;
                                ;ELSE                                   ;AN000;(of IF SwSW)
                                ;   stc                                 ;AN000; this logic works when the SwSW
                                ;   retn                                ;AN000; is reset.

                                ;******************************************************************************
                                ; _$P_Fill_Result
                                ;
                                ; Function: Fill the result buffer
                                ;
                                ; Input:    AH = Item tag
                                ;       AL = type
                                ;            AL = 1: CX,DX has 32bit number (CX = high)
                                ;            AL = 2: DX has index(offset) into value list
                                ;            AL = 6: DL has driver # (1-A, 2-B, ... , 26 - Z)
                                ;            AL = 7: DX has year, CL has month and CH has date
                                ;            AL = 8: DL has hours, DH has minutes, CL has seconds,
                                ;                    amd CH has hundredths
                                ;            AL = else: cs:SI points to returned string buffer
                                ;       ES:BX -> CONTROL block
                                ;
                                ; Output:   None
                                ;
                                ; Use:      _$P_Do_CAPS_String, _$P_Remove_Colon, _$P_Found_SYNONYM
                                ;
                                ; Vars: _$P_DX(W)
                                ;******************************************************************************

                                _$P_Fill_Result:
00001C27 57                         push    di                          ;AN000;
00001C28 268B7F04                   mov     di,[es:bx+_$P_Control_Blk.Result_Buf]
                                                                        ;AN000; di points to result buffer
00001C2C 2E893E[7519]               mov     [cs:_$P_DX],di              ;AC034; set returned result address
                                    ;mov    [es:di+_$P_Result_Blk.Type],al ;AN000; store type
                                    ;mov    [es:di+_$P_Result_Blk.Item_Tag],ah ;AN000; store item tag
                                    ; 07/09/2023
                                    ;mov    [es:di+_$P_Result_Blk.Type], ax
00001C31 268905                     mov     [es:di],ax                  ; store type (al) and item tag (ah)

00001C34 50                         push    ax                          ;AN000;
00001C35 2EA1[8419]                 mov     ax,[cs:_$P_Found_SYNONYM] ;AC034; if yes,
00001C39 26894502                   mov     [es:di+_$P_Result_Blk.SYNONYM_Ptr],ax
                                                                        ;AN000;   then set it to the result
00001C3D 58                         pop     ax                          ;AN000;
                                _$P_RLT04:                              ;AN000;
00001C3E 3C01                       cmp     al,_$P_Number               ;AN000; if number
00001C40 750A                       jne     short _$P_RLT00                     ;AN000;

                                _$P_RLT02:                              ;AN000;
00001C42 26895504                   mov     [es:di+_$P_Result_Blk.Picked_Val],dx ;AN000; then store 32bit
00001C46 26894D06                   mov     [es:di+_$P_Result_Blk.Picked_Val+2],cx ;AN000; number
00001C4A EB5A                       jmp     short _$P_RLT_Exit          ;AN000;

                                _$P_RLT00:                              ;AN000;
00001C4C 3C02                       cmp     al,_$P_List_Idx                     ;AN000; if list index
00001C4E 7506                       jne     short _$P_RLT01                     ;AN000;

00001C50 26895504                   mov     [es:di+_$P_Result_Blk.Picked_Val],dx
                                                                        ;AN000; then store list index
00001C54 EB50                       jmp     short _$P_RLT_Exit          ;AN000;

                                _$P_RLT01:                              ;AN000;
00001C56 3C07                       cmp     al,_$P_Date_F               ;AN000; Date format ?
00001C58 74E8                       je      short _$P_RLT02                     ;AN000;

00001C5A 3C08                       cmp     al,_$P_Time_F               ;AN000; Time format ?
00001C5C 74E4                       je      short _$P_RLT02                     ;AN000;

00001C5E 3C06                       cmp     al,_$P_Drive                ;AN000; drive format ?
00001C60 7506                       jne     short _$P_RLT03                     ;AN000;

00001C62 26885504                   mov     [es:di+_$P_Result_Blk.Picked_Val],dl ;AN000; store drive number
00001C66 EB3E                       jmp     short _$P_RLT_Exit          ;AN000;

                                _$P_RLT03:                              ;AN000;
00001C68 3C04                       cmp     al,_$P_Complex              ;AN000; complex format ?
00001C6A 750F                       jne     short _$P_RLT05                     ;AN000;

00001C6C 2EA1[7E19]                 mov     ax,[cs:_$P_SaveSI_Cmpx] ;AC034; then get pointer in command buffer
00001C70 40                         inc     ax                          ;AN000; skip left Parentheses
00001C71 26894504                   mov     [es:di+_$P_Result_Blk.Picked_Val],ax ;AN000; store offset
00001C75 268C5D06                   mov     [es:di+_$P_Result_Blk.Picked_Val+2],ds ;AN000; store segment
00001C79 EB2B                       jmp     short _$P_RLT_Exit          ;AN000;

                                _$P_RLT05:                              ;AN000;
                                ;-----------------------  AL = 3, 5, or 9
00001C7B 26897504                   mov     [es:di+_$P_Result_Blk.Picked_Val],si
                                                                        ;AN000; store offset of STRING_BUF
00001C7F 268C4D06                   mov     [es:di+_$P_Result_Blk.Picked_Val+2],cs
                                                                        ;AN031; store segment of STRING_BUF
00001C83 50                         push    ax                          ;AN000;
00001C84 26F6470201                 test    byte [es:bx+_$P_Control_Blk.Function_Flag],_$P_CAP_File
                                                                        ;AN000; need CAPS by file table?
00001C89 7404                       jz      short _$P_RLT_CAP00         ;AN000;

00001C8B B004                       mov     al,_$P_DOSTBL_File          ;AN000; use file upper case table
00001C8D EB09                       jmp     short _$P_RLT_CAP02         ;AN000;

                                _$P_RLT_CAP00:                                  ;AN000;
00001C8F 26F6470202                 test    byte [es:bx+_$P_Control_Blk.Function_Flag],_$P_CAP_Char
                                                                        ;AN000; need CAPS by char table ?
00001C94 7405                       jz      short _$P_RLT_CAP01         ;AN000;

00001C96 B002                       mov     al,_$P_DOSTBL_Char          ;AN000; use character upper case table
                                _$P_RLT_CAP02:                                  ;AN000;
00001C98 E8DF00                     call    _$P_Do_CAPS_String          ;AN000;  process CAPS along the table
                                _$P_RLT_CAP01:                                  ;AN000;
00001C9B 58                         pop     ax                          ;AN000;
00001C9C 26F6470210                 test    byte [es:bx+_$P_Control_Blk.Function_Flag],_$P_Rm_Colon
                                                                        ;AN000; removing colon at end ?
00001CA1 7403                       jz      short _$P_RLT_Exit          ;AN000;

00001CA3 E8AE00                     call    _$P_Remove_Colon            ;AN000; then process it.
                                _$P_RLT_Exit:                                   ;AN000;
00001CA6 5F                         pop     di                          ;AN000;
00001CA7 C3                         retn                                ;AN000;

                                ;******************************************************************************
```

```
                                ; _$P_Check_Match_Flags
                                ;
                                ; Function:  Check the mutch_flags and make the exit code and set the
                                ;        result buffer
                                ;
                                ;       Check for types in this order:
                                ;          Complex
                                ;          Date
                                ;          Time
                                ;          Drive
                                ;          Filespec
                                ;          Quoted String
                                ;          Simple String
                                ;
                                ; Input:     cs:SI -> _$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 ?
```

```
30113                                   ;lea    di,[_$P_Char_CAP_Ptr]
30114                                   ; 07/09/2023
30115 00001DAB BF[061A]                 mov     di,_$P_Char_CAP_Ptr
30116                                 _$P_DCC00:                      ;AN000;
30117 00001DAE 2E3815                   cmp     [cs:di],dl              ;AN000; already got table address ?
30118 00001DB1 7415                     je      short _$P_DCC01                 ;AN000; if no,
30119
30120                                 ;In this next section, ES will be used to pass a 5 byte workarea to INT 21h,
30121                                 ; the GET COUNTYRY INFO call. This usage of ES is required by the function
30122                                 ; call, regardless of what base register is currently be defined as cs.
30123
30124 00001DB3 50                       push    ax                      ;AN000; get CAPS table thru DOS call
30125 00001DB4 51                       push    cx                      ;AN000;
30126 00001DB5 52                       push    dx                      ;AN000;
30127
30128 00001DB6 0E                       push    cs                      ;AC036; pass current base seg into
30129                                                                   ;(Note: this used to push CS.  BUG...
30130 00001DB7 07                       pop     es                      ;AN000;   ES reg, required for
30131                                                                   ;get extended country information
30132                                   ;mov    al,dl ; function        ;AN000; upper case table
30133                                   ; 07/07/2023
30134 00001DB8 92                       xchg    ax,dx
30135 00001DB9 B465                     mov     ah,_$P_DOS_Get_TBL ; 65h ;AN000; get extended CDI
30136 00001DBB BBFFFF                   mov     bx,_$P_DOSTBL_Def ; -1;AN000; get active CON
30137 00001DBE B90500                   mov     cx,_$P_DOSTBL_BL ; 5    ;AN000; buffer length
30138                                   ;mov    dx,_$P_DOSTBL_Def       ;AN000; get for default code page
30139                                   ; 07/07/2023
30140 00001DC1 89DA                     mov     dx,bx ; 0FFFFh
30141                                                                   ;DI already set to point to buffer
30142 00001DC3 CD21                     int     21h                     ;AN000; es:di point to buffer that
30143                                                                   ;now has been filled in with info
30144 00001DC5 5A                       pop     dx                      ;AN000;
30145 00001DC6 59                       pop     cx                      ;AN000;
30146 00001DC7 58                       pop     ax                      ;AN000;
30147
30148                                 _$P_DCC01:                      ;AN000;
30149
30150                                 ;In this next section, ES will be used as the base of the XLAT table, provided
30151                                 ; by the previous GET COUNTRY INFO DOS call.  This usage of ES is made
30152                                 ; regardless of which base reg is currently the cs reg.
30153
30154                                     ; 14/04/2024
30155                                     ;mov    bx,[cs:di+_$P_DOS_TBL.Off] ;AN000; get offset of table
30156                                     ;mov    es,[cs:di+_$P_DOS_TBL.Seg] ;AN000; get segment of table
30157                                     ; 07/07/2023
30158 00001DC8 2EC45D01                 les     bx,[cs:di+_$P_DOS_TBL.Off]
30159 00001DCC 43                       inc     bx                      ;AC035; add '2' to
30160 00001DCD 43                       inc     bx                      ;AC035;  BX reg
30161                                                                   ;AN000; skip length field
30162 00001DCE 2C80                     sub     al,_$P_ASCII80 ; 80h    ;AN000; make char to index
30163                                   ;xlat   es:[bx]                 ;AN000; perform case map
30164 00001DD0 26                       es
30165 00001DD1 D7                       xlat
30166 00001DD2 5F                       pop     di                      ;AN000;
30167 00001DD3 07                       pop     es                      ;AN000;
30168 00001DD4 5B                       pop     bx                      ;AN000;
30169                                 _$P_CAPS_Ret:                           ;AN000;
30170 00001DD5 C3                       retn                            ;AN000;
30171
30172                                 ;****************************************************************************
30173                                 ; _$P_Value / _$P_SValue
30174                                 ;
30175                                 ; Function:  Make 32bit value from cs:SI and see value list
30176                                 ;       and make result buffer.
30177                                 ;       _$P_SValue is an entry point for the signed value
30178                                 ;       and this will simply call _$P_Value after the handling
30179                                 ;       of the sign character, "+" or "-"
30180                                 ;
30181                                 ; Input:     cs:SI -> _$P_STRING_BUF
30182                                 ;       ES:BX -> CONTROL block
30183                                 ;
30184                                 ; Output:    None
30185                                 ;
30186                                 ; Use:      _$P_Fill_Result, _$P_Check_OVF
30187                                 ;
30188                                 ; Vars: _$P_RC(W), _$P_Flags(RW)
30189                                 ;****************************************************************************
30190
30191                                     ; 26/10/2022 - Retro DOS v4.0
30192                                     ; (MSDOS 5.0 IO.SYS - SYSINIT:1B0Bh)
30193
30194                                     ; 07/07/2023 - Retro DOS v4.2 IO.SYS (optimization)
30195                                     ; (MSDOS 6.21 IO.SYS - SYSINIT:1C46h)
30196                                 _$P_SValue:                     ;AN000; when signed value here
30197 00001DD6 50                       push    ax                      ;AN000;
30198 00001DD7 2E800E[7D19]80           or      byte [cs:_$P_Flags2],_$P_Signed ;AC034; indicate a signed numeric
30199 00001DDD 2E8026[7D19]FD           and     byte [cs:_$P_Flags2],0FFh-_$P_Neg ;AC034; assume positive value
30200                                   ;and    byte [cs:_$P_Flags2],~_$P_Neg ; 07/07/2023
30201 00001DE3 2E8A04                   mov     al,[cs:si]              ;AN000; get sign
30202 00001DE6 3C2B                     cmp     al,_$P_Plus             ;AN000; "+" ?
30203 00001DE8 740A                     je      short _$P_SVal00        ;AN000;
30204
30205 00001DEA 3C2D                     cmp     al,_$P_Minus            ;AN000; "-" ?
30206 00001DEC 7507                     jne     short _$P_SVal01        ;AN000; else
30207
30208 00001DEE 2E800E[7D19]02           or      byte [cs:_$P_Flags2],_$P_Neg ;AC034; set this is negative value
30209                                 _$P_SVal00:                     ;AN000;
30210 00001DF4 46                       inc     si                      ;AN000; skip sign char
30211                                 _$P_SVal01:                     ;AN000;
30212 00001DF5 E80200                   call    _$P_Value               ;AN000; and process value
30213 00001DF8 58                       pop     ax                      ;AN000;
30214 00001DF9 C3                       retn
30215
30216                                 ;****************************************************************************
30217
30218                                     ; 07/07/2023 - Retro DOS v4.2 IO.SYS (optimization)
30219                                     ; (MSDOS 6.21 IO.SYS - SYSINIT:1C6Ah)
30220
30221                                     ; 26/10/2022
30222                                 _$P_Value:                      ;AN000;
30223 00001DFA 50                       push    ax                      ;AN000;
30224 00001DFB 51                       push    cx                      ;AN000;
30225 00001DFC 52                       push    dx                      ;AN000;
30226 00001DFD 56                       push    si                      ;AN000;
30227 00001DFE 31C9                     xor     cx,cx                   ;AN000; cx = higher 16 bits
30228 00001E00 31D2                     xor     dx,dx                   ;AN000; dx = lower 16 bits
30229 00001E02 53                       push    bx                      ;AN000; save control pointer
30230                                 _$P_Value_Loop:                         ;AN000;
30231 00001E03 2E8A04                   mov     al,[cs:si]              ;AN000; get character
30232 00001E06 08C0                     or      al,al                   ;AN000; end of line ?
30233 00001E08 7438                     jz      short _$P_Value00       ;AN000;
30234
30235 00001E0A E8DC00                   call    _$P_0099                ;AN000; make asc(0..9) to bin(0..9)
30236 00001E0D 722F                     jc      short _$P_Value_Err0    ;AN000;
```

```
30237
30238 00001E0F 30E4                      xor     ah,ah                   ;AN000;
30239 00001E11 89C5                      mov     bp,ax                   ;AN000; save binary number
30240
30241                                  ; 08/07/2023 - Retro DOS v4.2 IO.SYS (optimization)
30242                                  ; Ref: Disassembled PCDOS 7.1 IBMBIO.COM SYSINIT code
30243                                  ;                               Erdogan Tan - July 2023
30244                                  %if 0
30245                                      shl     dx,1                    ;AN000; to have 2*x
30246                                      rcl     cx,1                    ;AN000; shift left w/ carry
30247                                      call    _$P_Check_OVF           ;AN000; Overflow occurred ?
30248                                      jc      short _$P_Value_Err0    ;AN000; then error, exit
30249
30250                                      mov     bx,dx                   ;AN000; save low(2*x)
30251                                      mov     ax,cx                   ;AN000; save high(2*x)
30252                                      shl     dx,1                    ;AN000; to have 4*x
30253                                      rcl     cx,1                    ;AN000; shift left w/ carry
30254                                      call    _$P_Check_OVF           ;AN000; Overflow occurred ?
30255                                      jc      short _$P_Value_Err0    ;AN000; then error, exit
30256
30257                                      shl     dx,1                    ;AN000; to have 8*x
30258                                      rcl     cx,1                    ;AN000; shift left w/ carry
30259                                      call    _$P_Check_OVF           ;AN000; Overflow occurred ?
30260                                      jc      short _$P_Value_Err0    ;AN000; then error, exit
30261
30262                                      add     dx,bx                   ;AN000; now have 10*x
30263                                      adc     cx,ax                   ;AN000; 32bit ADD
30264                                      call    _$P_Check_OVF           ;AN000; Overflow occurred ?
30265                                      jc      short _$P_Value_Err0    ;AN000; then error, exit
30266
30267                                      add     dx,bp                   ;AN000; Add the current one degree decimal
30268                                      adc     cx,0                    ;AN000; if carry, add 1 to high 16bit
30269                                      call    _$P_Check_OVF           ;AN000; Overflow occurred ?
30270                                      jc      short _$P_Value_Err0    ;AN000; then error, exit
30271
30272                                      inc     si                      ;AN000; update pointer
30273                                      jmp     short _$P_Value_Loop    ;AN000; loop until NULL encountered
30274                                  ;_$P_Value_Err0:
30275                                  %endif
30276                                  ;****
30277                                  %if 1
30278                                  ; 08/07/2023 - Retro DOS v4.2 IO.SYS (optimization)
30279                                      ; (PCDOS 7.1 IBMBIO.COM - SYSINIT:2130h)
30280
30281                                      ; 14/04/2024 - Retro DOS v5.0
30282                                      ;xor     ah,ah
30283                                      ;mov     bp,ax                  ; save binary number
30284
30285 00001E13 E81C00                    call    _$P_Value_2x_OVF        ; multiply cx:dx by 2 and then check overflow
30286 00001E16 89D3                      mov     bx,dx                   ; ax:bx = 2*(cx:dx)
30287 00001E18 89C8                      mov     ax,cx
30288 00001E1A E81500                    call    _$P_Value_2x_OVF        ; multiply cx:dx by 2 and then check overflow
30289 00001E1D E81200                    call    _$P_Value_2x_OVF        ; multiply cx:dx by 2 and then check overflow
30290 00001E20 01DA                      add     dx,bx                   ; 8*(cx:dx)+2*(cx:dx) = 10*(cx:dx)
30291 00001E22 11C1                      adc     cx,ax
30292 00001E24 E80F00                    call    _$P_Value_Chk_Add_OVF
30293 00001E27 01EA                      add     dx,bp                   ; Add the current one degree decimal
30294                                                                    ; if carry, add 1 to high 16bit
30295 00001E29 83D100                    adc     cx,0
30296 00001E2C E80700                    call    _$P_Value_Chk_Add_OVF ; Overflow occurred ?
30297                                                                    ; then error, exit (without return here)
30298 00001E2F 46                        inc     si                      ; update pointer
30299 00001E30 EBD1                      jmp     short _$P_Value_Loop
30300
30301                                  _$P_Value_2x_OVF:
30302 00001E32 D1E2                      shl     dx,1                    ; to have 2*x
30303 00001E34 D1D1                      rcl     cx,1                    ; shift left w/ carry
30304                                  _$P_Value_Chk_Add_OVF:
30305 00001E36 E89E00                    call    _$P_Check_OVF           ; check overflow (for the last shift or add)
30306 00001E39 7201                      jc      short _$P_Value_OVF
30307 00001E3B C3                        retn
30308                                  _$P_Value_OVF:
30309 00001E3C 44                        inc     sp                      ; skip "call" return address to the caller
30310 00001E3D 44                        inc     sp
30311
30312                                  ;_$P_Value_Err0:
30313                                  %endif
30314                                  ;****
30315
30316                                  _$P_Value_Err0:                         ;AN000;
30317 00001E3E 5B                        pop     bx                      ;AN000;
30318 00001E3F E98300                    jmp     _$P_Value_Err           ;AN000; Bridge
30319                                  ;
30320                                  _$P_Value00:                            ;AN000;
30321 00001E42 5B                        pop     bx                      ;AN000; restore control pointer
30322 00001E43 2EF606[7D19]02            test    byte [cs:_$P_Flags2],_$P_Neg ;AC034; here cx,dx = 32bit value
30323 00001E49 740A                      jz      short _$P_Value01       ;AN000; was it negative ?
30324
30325 00001E4B F7D1                      not     cx                      ;AN000; +
30326 00001E4D F7D2                      not     dx                      ;AN000; |- Make 2's complement
30327 00001E4F 83C201                    add     dx,1                    ;AN000; |
30328 00001E52 83D100                    adc     cx,0                    ;AN000; +
30329
30330                                  _$P_Value01:                            ;AN000; / nval = 0
30331 00001E55 268B7706                  mov     si,[es:bx+_$P_Control_Blk.Value_List] ;AN000; si points to value list
30332 00001E59 268A04                    mov     al,[es:si]              ;AN000; get nval
30333                                    ; 07/09/2023
30334                                    ;cmp    al,_$P_nval_None ; 0    ;AN000; no value list ?
30335                                    ;;*jne  short _$P_Value02       ;AN000;
30336                                    ;;* 07/07/2023
30337                                    ;je     short _$P_Value05
30338                                    ; 07/09/2023
30339 00001E5C 08C0                      or      al,al
30340 00001E5E 7459                      jz      short _$P_Value05 ; _$P_nval_None
30341
30342                                    ;mov    al,_$P_Number           ;AN000; Set type
30343                                    ;mov    ah,_$P_No_Tag           ;AN000; No ITEM_TAG set
30344                                    ; 07/07/2023
30345                                    ;*mov   ax,(_$P_No_Tag<<8)|_$P_Number
30346                                    ;*jmp   short _$P_Value_Exit    ;AN000;
30347
30348                                    ; 26/10/2022 (MSDOS 5.0 IO.SYS, SYSINIT compatibility)
30349                                    ; (SYSINIT:1BA5h)
30350                                    ; 12/12/2022
30351                                    ;nop    ; db  90h
30352
30353                                  _$P_Value02:                            ;AN000; / nval = 1
30354                                  ;IF Val1SW                              ;AN000;(Check if value list id #1 is supported)
30355                                  ;(tm07) cmp al,_$P_nval_Range           ;AN000; have range list ?
30356                                  ;(tm07) jne short _$P_Value03           ;AN000;
30357
30358 00001E60 46                        inc     si                      ;AN000;
30359 00001E61 268A04                    mov     al,[es:si]              ;AN000; al = number of range
30360
```

```
                                    ; 07/09/2023
                                    ;cmp    al,_$P_No_nrng          ;AN000; (tm07)
                                    ;je     short _$P_Value03       ;AN000; (tm07)
00001E64 08C0                       or      al,al
00001E66 745D                       jz      short _$P_Value03 ; _$P_No_nrng

00001E68 46                         inc     si                      ;AN000; si points to 1st item_tag
                                _$P_Val02_Loop:                             ;AN000;
00001E69 2EF606[7D19]80             test    byte [cs:_$P_Flags2],_$P_Signed ;AC034;
00001E6F 751E                       jnz     short _$P_Val02_Sign    ;AN000;

00001E71 263B4C03                   cmp     cx,[es:si+_$P_Val_List.Val_XH] ;AN000; comp cx with XH
00001E75 7234                       jb      short _$P_Val02_Next    ;AN000;
00001E77 7706                       ja      short _$P_Val_In        ;AN000;

00001E79 263B5401                   cmp     dx,[es:si+_$P_Val_List.Val_XL] ;AN000; comp dx with XL
00001E7D 722C                       jb      short _$P_Val02_Next    ;AN000;

                                _$P_Val_In:                             ;AN000;
00001E7F 263B4C07                   cmp     cx,[es:si+_$P_Val_List.Val_YH] ;AN000; comp cx with YH (tm01)
00001E83 7726                       ja      short _$P_Val02_Next    ;AN000;
00001E85 7237                       jb      short _$P_Val_Found     ;AN000;

00001E87 263B5405                   cmp     dx,[es:si+_$P_Val_List.Val_YL] ;AN000; comp dx with YL
00001E8B 771E                       ja      short _$P_Val02_Next    ;AN000;

00001E8D EB2F                       jmp     short _$P_Val_Found     ;AN000;

                                _$P_Val02_Sign:                             ;AN000;
00001E8F 263B4C03                   cmp     cx,[es:si+_$P_Val_List.Val_XH]      ;AN000; comp cx with XH
00001E93 7C16                       jl      short _$P_Val02_Next    ;AN000;
00001E95 7F06                       jg      short _$P_SVal_In       ;AN000;

00001E97 263B5401                   cmp     dx,[es:si+_$P_Val_List.Val_XL]      ;AN000; comp dx with XL
00001E9B 7C0E                       jl      short _$P_Val02_Next    ;AN000;

                                _$P_SVal_In:                                ;AN000;
00001E9D 263B4C07                   cmp     cx,[es:si+_$P_Val_List.Val_YH]      ;AN000; comp cx with YH
00001EA1 7F08                       jg      short _$P_Val02_Next    ;AN000;

00001EA3 7C19                       jl      short _$P_Val_Found     ;AN000;

00001EA5 263B5405                   cmp     dx,[es:si+_$P_Val_List.Val_YL]      ;AN000; comp dx with YL
                                    ;jg     short _$P_Val02_Next    ;AN000;
                                    ;jmp    short _$P_Val_Found     ;AN000;
                                    ; 07/07/2023
00001EA9 7E13                       jng     short _$P_Val_Found

                                _$P_Val02_Next:                             ;AN000;
00001EAB 83C609                     add     si,_$P_Len_Range        ;AN000;
00001EAE FEC8                       dec     al                      ;AN000; loop nrng times in AL
00001EB0 75B7                       jne     short _$P_Val02_Loop    ;AN000;
                                                                    ; / Not found
00001EB2 2EC706[7119]0600           mov     word [cs:_$P_RC],_$P_Out_Of_Range ;AC034;
                                    ;mov    al,_$P_Number           ;AN000;
                                    ;mov    ah,_$P_No_Tag           ;AN000; No ITEM_TAG set
                                _$P_Value05:                ;* 07/07/2023
                                    ; 07/07/2023
00001EB9 B801FF                     mov     ax,(_$P_No_Tag<<8)|_$P_Number
00001EBC EB11                       jmp     short _$P_Value_Exit    ;AN000;

                                _$P_Val_Found:                              ;AN000;
00001EBE B001                       mov     al,_$P_Number           ;AN000;
00001EC0 268A24                     mov     ah,[es:si]              ;AN000; found ITEM_TAG set
00001EC3 EB0A                       jmp     short _$P_Value_Exit    ;AN000;

                                _$P_Value03:                                ;AN000; / nval = 2

                                ;IF Val2SW                          ;AN000;(Check if value list id #2 is supported)
                                ;;;;cmp     al,$P_nval_Value        ; have match list ? ASSUME nval=2,
                                ;;;;jne     $P_Value04              ; even if it is 3 or more.
                                ;(tm07) inc si                      ;AN000;
                                ;(tm07) mov al,es:[si]              ;AN000; al = nrng
                                ;   mov     ah,$P_Len_Range         ;AN000;
                                ;   mul     ah                      ;AN000;  Skip nrng field
                                ;   inc     ax                      ;AN000;
                                ;   add     si,ax                   ;AN000; si points to nnval
                                ;   mov     al,es:[si]              ;AN000; get nnval
                                ;   inc     si                      ;AN000; si points to 1st item_tag
                                ;$P_Val03_Loop:                             ;AN000;
                                ;   cmp     cx,es:[si+$P_Val_XH]    ;AN000; comp cx with XH
                                ;   jne     $P_Val03_Next           ;AN000;
                                ;
                                ;   cmp     dx,es:[si+$P_Val_XL]    ;AN000; comp dx with XL
                                ;   je      $P_Val_Found            ;AN000;
                                ;
                                ;$P_Val03_Next:                             ;AN000;
                                ;   add     si,$P_Len_Value         ;AN000; points to next value choice
                                ;   dec     al                      ;AN000; loop nval times in AL
                                ;   jne     $P_Val03_Loop           ;AN000;
                                ;                                   ;AN000; / Not found
                                ;   mov     psdata_seg:$P_RC,$P_Not_in_Val ;AC034;
                                ;   mov     al,$P_Number            ;AN000;
                                ;   mov     ah,$P_No_Tag            ;AN000; No ITEM_TAG set
                                ;   jmp     short $P_Value_Exit     ;AN000;
                                ;
                                ;ENDIF                                  ;AN000;(of Val2SW)
                                ;$P_Value04:

                                _$P_Value_Err:                              ;AN000;
00001EC5 2EC706[7119]0900           mov     word [cs:_$P_RC],_$P_Syntax ;AC034;
                                    ;mov    al,_$P_String           ;AN000; Set type
                                    ;mov    ah,_$P_No_Tag           ;AN000; No ITEM_TAG set
                                    ; 07/09/2023
                                    ; 07/07/2023
00001ECC B803FF                     mov     ax,(_$P_No_Tag<<8)|_$P_String
                                _$P_Value_Exit:                             ;AN000;
00001ECF E855FD                     call    _$P_Fill_Result             ;AN000;
00001ED2 5E                         pop     si                      ;AN000;
00001ED3 5A                         pop     dx                      ;AN000;
00001ED4 59                         pop     cx                      ;AN000;
00001ED5 58                         pop     ax                      ;AN000;
00001ED6 C3                         retn                            ;AN000;

                                ; 28/03/2019 - Retro DOS v4.0

                                ;--------------------------------------------------------------------------
                                ; _$P_Check_OVF
                                ;
                                ; Function:  Check if overflow is occurred with consideration of
                                ;        signed or un-signed numeric value
                                ;
                                ; Input:     Flag register
                                ;
```

```
                                ; Output:    CY = 1  :    Overflow
                                ;
                                ; 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
```

```
30609 00001F29 E83200                   call    _$P_String_Comp             ;AN000; compare it with operand
30610 00001F2C 7312                     jnc     short _$P_Sim_Found   ;AN000; found on list ?
30611
30612 00001F2E 83C703                   add     di,_$P_Len_String ; 3 ;AN000; if no, point to next choice
30613 00001F31 FEC8                     dec     al                    ;AN000; loop nstval times in AL
30614 00001F33 75F1                     jne     short _$P_Sim_Loop    ;AN000;
30615                                                                 ;AN000; / Not found
30616 00001F35 2EC706[7119]0800         mov     word [cs:_$P_RC],_$P_Not_In_Str ;AC034;
30617 00001F3C B4FF                     mov     ah,_$P_No_Tag         ;AN000; No ITEM_TAG set
30618 00001F3E EB14                     jmp     short _$P_Sim_Exit    ;AN000;
30619
30620                                 _$P_Sim_Found:                          ;AN000;
30621 00001F40 268A65FF                 mov     ah,[es:di-1]          ;AN000; set item_tag
30622 00001F44 B002                     mov     al,_$P_List_Idx               ;AN000;
30623 00001F46 268B15                   mov     dx,[es:di]            ;AN000; get address of STRING
30624 00001F49 EB0B                     jmp     short _$P_Sim_Exit0   ;AN000;
30625                                 ;ENDIF                                  ;AN000;(of Val3SW+KeySW)
30626                                 _$P_Sim01:                      ;AN000;
30627 00001F4B 2EC706[7119]0900         mov     word [cs:_$P_RC],_$P_Syntax ;AC034;
30628 00001F52 B4FF                     mov     ah,_$P_No_Tag         ;AN000; No ITEM_TAG set
30629                                 _$P_Sim_Exit:                           ;AN000;
30630 00001F54 B003                     mov     al,_$P_String         ;AN000; Set type
30631                                 _$P_Sim_Exit0:                          ;AN000;
30632 00001F56 E8CEFC                   call    _$P_Fill_Result               ;AN000;
30633 00001F59 5F                       pop     di                    ;AN000;
30634 00001F5A 5A                       pop     dx                    ;AN000;
30635 00001F5B 5B                       pop     bx                    ;AN000;
30636 00001F5C 58                       pop     ax                    ;AN000;
30637 00001F5D C3                       retn                          ;AN000;
30638
30639                                 ;******************************************************************************
30640                                 ; _$P_String_Comp:
30641                                 ;
30642                                 ; Function:  Compare two string
30643                                 ;
30644                                 ; Input:     cs:SI -> 1st string
30645                                 ;       ES:BP -> 2nd string  (Must be upper case)
30646                                 ;       ES:BX -> CONTROL block
30647                                 ;
30648                                 ; Output:    CY = 1 if not match
30649                                 ;
30650                                 ; Use:      _$P_Chk_DBCS, _$P_Do_CAPS_Char
30651                                 ;
30652                                 ; Vars: _$P_KEYor_SW_Ptr(W), _$P_Flags(R). _$P_KEYorSW_Ptr
30653                                 ;******************************************************************************
30654
30655                                 _$P_String_Comp:
30656 00001F5E 50                       push    ax                    ;AN000;
30657 00001F5F 55                       push    bp                    ;AN000;
30658 00001F60 52                       push    dx                    ;AN000;
30659 00001F61 56                       push    si                    ;AN000;
30660 00001F62 B202                     mov     dl,_$P_DOSTBL_Char    ;AN000; use character case map table
30661                                 _$P_SCOM_Loop:                          ;AN000;
30662 00001F64 2E8A04                   mov     al,[cs:si]            ;AN000; get command character
30663 00001F67 E81502                   call    _$P_Chk_DBCS          ;AN000; DBCS ?
30664 00001F6A 723A                     jc      short _$P_SCOM00      ;AN000; yes,DBCS
30665
30666 00001F6C E82AFE                   call    _$P_Do_CAPS_Char      ;AN000; else, upper case map before comparison
30667                                 ;IF KeySW+SwSW                          ;AN000;(Check if keyword or switch is supported)
30668 00001F6F 2EF606[7D19]08           test    byte [cs:_$P_Flags2],_$P_Key_Cmp ;AC034; keyword search ?
30669 00001F75 740D                     jz      short _$P_SCOM04      ;AN000;
30670
30671 00001F77 3C3D                     cmp     al,_$P_Keyword        ;AN000; "=" is delimiter
30672 00001F79 751F                     jne     short _$P_SCOM03      ;AN000; IF "=" on command line AND  (bp+1=> char after the "=" in synonym
list)
30673
30674 00001F7B 26807E0100               cmp     byte [es:bp+1],_$P_NULL ;AN021;  at end of keyword string in the control block THEN
30675 00001F80 756D                     jne     short _$P_SCOM_Differ ;AN021;
30676
30677 00001F82 EB13                     jmp     short _$P_SCOM05      ;AN000; keyword found in synonym list
30678
30679                                 _$P_SCOM04:                             ;AN000;
30680 00001F84 2EF606[7D19]10           test    byte [cs:_$P_Flags2],_$P_SW_Cmp ;AC034; switch search ?
30681 00001F8A 740E                     jz      short _$P_SCOM03      ;AN000;
30682
30683 00001F8C 3C3A                     cmp     al,_$P_Colon          ;AN000; ":" is delimiter, at end of switch on command line
30684 00001F8E 750A                     jne     short _$P_SCOM03      ;AN000; continue compares
30685
30686                                   ; 26/10/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
30687                                   ;cmp    byte [es:bp+0],_$P_NULL
30688                                   ; 11/12/2022
30689 00001F90 26807E0000               cmp     byte [es:bp],_$P_NULL ;AN021; IF at end of switch on command AND
30690 00001F95 7558                     jne     short _$P_SCOM_Differ ;AN021;   at end of switch string in the control block THEN
30691
30692                                 _$P_SCOM05:                             ;AN000;   found a match
30693 00001F97 46                       inc     si                    ;AN000; si points to just after "=" or ":"
30694 00001F98 EB58                     jmp     short _$P_SCOM_Same   ;AN000; exit
30695
30696                                 _$P_SCOM03:                             ;AN000;
30697                                 ;ENDIF                                  ;AN000;(of KeySW+SwSW)
30698                                   ; 26/10/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
30699                                   ;cmp    al,[es:bp+0]
30700                                   ; 11/12/2022
30701 00001F9A 263A4600                 cmp     al,[es:bp]            ;AN000; compare operand w/ a synonym
30702 00001F9E 751B                     jne     short _$P_SCOM_Differ0        ;AN000; if different, check ignore colon option
30703
30704 00001FA0 08C0                     or      al,al                 ;AN000; end of line
30705 00001FA2 744E                     jz      short _$P_SCOM_Same   ;AN000; if so, exit
30706
30707                                   ; 12/12/2022
30708                                   ;inc    si                    ;AN000; update operand pointer
30709                                   ;inc    bp                    ;AN000;    and synonym pointer
30710                                   ; 26/10/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
30711 00001FA4 EB11                     jmp     short _$P_SCOM01      ;AN000; loop until NULL or "=" or ":" found in case
30712
30713                                 _$P_SCOM00:                             ;AN000; Here al is DBCS leading byte
30714                                   ; 26/10/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
30715                                   ;cmp    al,[es:bp+0]
30716                                   ; 11/12/2022
30717 00001FA6 263A4600                 cmp     al,[es:bp]            ;AN000; compare leading byte
30718 00001FAA 7543                     jne     short _$P_SCOM_Differ ;AN000; if not match, say different
30719
30720 00001FAC 46                       inc     si                    ;AN000; else, load next byte
30721 00001FAD 2E8A04                   mov     al,[cs:si]            ;AN000; and
30722 00001FB0 45                       inc     bp                    ;AN000;
30723                                   ; 26/10/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
30724                                   ;cmp    al,[es:bp+0]
30725                                   ; 11/12/2022
30726 00001FB1 263A4600                 cmp     al,[es:bp]            ;AN000; compare 2nd byte
30727 00001FB5 7538                     jne     short _$P_SCOM_Differ ;AN000; if not match, say different, too
30728
30729                                   ; 26/10/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
30730                                   ; 12/12/2022
30731                                 _$P_SCOM01:
```

```
00001FB7 46                             inc     si                      ;AN000; else update operand pointer
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)
```

```
30856                                   ;test  word [es:bx],_$P_Optional
30857 00002028 7532                     jnz    short _$P_FileF02     ;AN000;
30858
30859 0000202A 2EC706[7119]0200         mov    word [cs:_$P_RC],_$P_Op_Missing ;AC034; 3/17/87
30860 00002031 EB29                     jmp    short _$P_FileF02     ;AN000;
30861
30862                                 _$P_FileF03:                           ;AN000;
30863 00002033 58                       pop    ax                    ;AN000; discard save si
30864 00002034 56                       push   si                    ;AN000; save new si
30865                                 _$P_FileF_Loop1:                       ;AN000;
30866 00002035 2E8A04                   mov    al,[cs:si]            ;AN000; load character (not special char)
30867 00002038 08C0                     or     al,al                 ;AN000; end of line ?
30868 0000203A 741E                     jz     short _$P_FileF_RLT   ;AN000;
30869
30870 0000203C E82C00                   call   _$P_FileSp_Chk        ;AN000; File special character ?
30871 0000203F 740B                     je     short _$P_FileF00     ;AN000;
30872
30873 00002041 E83B01                   call   _$P_Chk_DBCS          ;AN000; no, then DBCS ?
30874 00002044 7302                     jnc    short _$P_FileF01     ;AN000;
30875 00002046 47                       inc    di                    ;AN000; if yes, skip next byte
30876 00002047 46                       inc    si                    ;AN000;
30877                                 _$P_FileF01:                           ;AN000;
30878 00002048 47                       inc    di                    ;AN000;
30879 00002049 46                       inc    si                    ;AN000;
30880 0000204A EBE9                     jmp    short _$P_FileF_Loop1 ;AN000;
30881                                 ;
30882                                 _$P_FileF00:                           ;AN000;
30883 0000204C 2EA2[7719]               mov    [cs:_$P_Terminator],al;AC034;
30884 00002050 2EC60400                 mov    byte [cs:si],_$P_NULL ;AN000; update end of string
30885 00002054 47                       inc    di                    ;AN000;
30886 00002055 2E893E[7319]             mov    [cs:_$P_SI_Save],di   ;AC034; update next pointer in command line
30887                                 _$P_FileF_RLT:                         ;AN000;
30888 0000205A 5E                       pop    si                    ;AN000;
30889 0000205B 5F                       pop    di                    ;AN000;
30890                                 _$P_FileF02:                           ;AN000;
30891 0000205C 58                       pop    ax                    ;AN000; (tm14)
30892                                   ;test  ax,_$P_File_Spc       ; 200h ;AN000; (tm14)
30893                                   ; 08/07/2023
30894 0000205D F6C402                   test   ah,(_$P_File_Spc>>8) ; 2
30895 00002060 7408                     jz     short _$P_Drv_Only_Exit       ;AN000; (tm14)
30896
30897 00002062 50                       push   ax                    ;AN000; (tm14)
30898                                   ;mov   ah,_$P_No_Tag         ;AN000; set
30899                                   ;mov   al,_$P_File_Spec      ;AN000; result
30900                                   ; 08/07/2023
30901 00002063 B805FF                   mov    ax,(_$P_No_Tag<<8)|_$P_File_Spec ; 0FF05h
30902                                                                ; set result
30903 00002066 E8BEFB                   call   _$P_Fill_Result               ;AN000; buffer to file spec
30904 00002069 58                       pop    ax                    ;AN000;
30905
30906                                 _$P_Drv_Only_Exit:                     ;AN000; (tm14)
30907 0000206A C3                       retn                         ;AN000;
30908
30909                                 ;******************************************************************************
30910                                 ; _$P_FileSp_Chk
30911                                 ;
30912                                 ; Function:  Check if the input byte is one of file special characters
30913                                 ;
30914                                 ; Input:     cs:SI -> _$P_STRING_BUF
30915                                 ;        AL = character code to be examineed
30916                                 ;
30917                                 ; Output:    ZF = 1 , AL is one of special characters
30918                                 ;******************************************************************************
30919
30920                                 _$P_FileSp_Chk:
30921 0000206B 53                       push   bx                    ;AN000;
30922 0000206C 51                       push   cx                    ;AN000;
30923                                   ;;lea  bx,[cs:_$P_FileSp_Char] ;AC034; special character table
30924                                   ;lea   bx,[_$P_FileSp_Char]  ; "[]|<>+=;\"" at
30925                                                                ; MSDOS 6.21 IO.SYS - SYSINIT:1838h
30926                                   ; 07/09/2023
30927 0000206D BB[0B1A]                 mov    bx,_$P_FileSp_Char
30928 00002070 B90900                   mov    cx,_$P_FileSp_Len ; 9 ;AN000; load length of it
30929                                 _$P_FileSp_Loop:                       ;AN000;
30930 00002073 2E3A07                   cmp    al,[cs:bx]            ;AN000; is it one of special character ?
30931 00002076 7404                     je     short _$P_FileSp_Exit ;AN000;
30932
30933 00002078 43                       inc    bx                    ;AN000;
30934 00002079 E2F8                     loop   _$P_FileSp_Loop               ;AN000;
30935
30936 0000207B 41                       inc    cx                    ;AN000; reset ZF
30937                                 _$P_FileSp_Exit:                       ;AN000;
30938 0000207C 59                       pop    cx                    ;AN000;
30939 0000207D 5B                       pop    bx                    ;AN000;
30940 0000207E C3                       retn
30941
30942                                 ;ENDIF                                 ;AN000;(of FileSW+DrvSW)
30943
30944                                 ;IF DrvSW                        ;AN000;(Check if drive only is supported)
30945
30946                                 ;******************************************************************************
30947                                 ; _$P_Drive_Format;
30948                                 ;
30949                                 ; Function:  Check if the input string is valid drive only format.
30950                                 ;        And set the result buffer.
30951                                 ;
30952                                 ; Input:     cs:SI -> _$P_STRING_BUF
30953                                 ;        ES:BX -> CONTROL block
30954                                 ;
30955                                 ; Output:    None
30956                                 ;
30957                                 ; Use:      _$P_Fill_Result, _$P_Chk_DBCS
30958                                 ;
30959                                 ; Vars: _$P_RC(W)
30960                                 ;******************************************************************************
30961
30962                                 _$P_Drive_Format:
30963 0000207F 50                       push   ax                    ;AN000;
30964 00002080 52                       push   dx                    ;AN000;
30965 00002081 2E8A04                   mov    al,[cs:si]            ;AN000;
30966 00002084 08C0                     or     al,al                 ;AN000; if null string
30967 00002086 7436                     je     short _$P_Drv_Exit    ;AN000; do nothing
30968
30969 00002088 E8F400                   call   _$P_Chk_DBCS          ;AN000; is it leading byte ?
30970 0000208B 722A                     jc     short _$P_Drv_Err     ;AN000;
30971
30972 0000208D 2E837C013A               cmp    word [cs:si+1],_$P_Colon ;AN000; "d", ":", 0  ?
30973 00002092 740D                     je     short _$P_DrvF00      ;AN000;
30974
30975                                   ;test  word [es:bx+_$P_Control_Blk.Match_Flag],_$P_Ig_Colon
30976                                   ;test  byte [es:bx+_$P_Control_Blk.Match_Flag],_$P_Ig_Colon ;AN000; colon can be ignored?
30977                                   ; 12/12/2022
30978 00002094 26F60710                 test   byte [es:bx],_$P_Ig_Colon
30979                                   ; 26/10/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
```

```
30980                                   ;test   word [es:bx],_$P_Ig_Colon
30981 00002098 741D                     jz      short _$P_Drv_Err       ;AN000;
30982
30983 0000209A 2E807C0100               cmp     byte [cs:si+1],_$P_NULL ;AN000; "d", 0  ?
30984 0000209F 7516                     jne     short _$P_Drv_Err       ;AN000;
30985
30986                                   _$P_DrvF00:                       ;AN000;
30987 000020A1 0C20                     or      al,_$P_Make_Lower       ;AN000; lower case
30988 000020A3 3C61                     cmp     al,"a"                    ;AN000; drive letter must
30989 000020A5 7210                     jb      short _$P_Drv_Err       ;AN000; in range of
30990
30991 000020A7 3C7A                     cmp     al,"z"                    ;AN000; "a"-"z"
30992 000020A9 770C                     ja      short _$P_Drv_Err       ;AN000; if no, error
30993
30994 000020AB 2C60                     sub     al,"a"-1                  ;AN000; make text drive to binary drive
30995 000020AD 88C2                     mov     dl,al                   ;AN000; set
30996                                   ;mov    ah,_$P_No_Tag           ;AN000; result
30997                                   ;mov    al,_$P_Drive            ;AN000; buffer
30998                                   ; 08/07/2023
30999 000020AF B806FF                   mov     ax,(_$P_No_Tag<<8)|_$P_Drive ; 0FF06h
31000                                                                   ; set result buffer
31001 000020B2 E872FB                   call    _$P_Fill_Result             ;AN000; to drive
31002 000020B5 EB07                     jmp     short _$P_Drv_Exit      ;AN000;
31003
31004                                   _$P_Drv_Err:                        ;AN000;
31005 000020B7 2EC706[7119]0900         mov     word [cs:_$P_RC],_$P_Syntax ;AC034;
31006                                   _$P_Drv_Exit:                       ;AN000;
31007 000020BE 5A                       pop     dx                      ;AN000;
31008 000020BF 58                       pop     ax                      ;AN000;
31009 000020C0 C3                       retn                            ;AN000;
31010
31011                                   ;ENDIF                              ;AN000;(of DrvSW)
31012
31013                                   ;******************************************************************************
31014                                   ; _$P_Skip_Delim;
31015                                   ;
31016                                   ; Function: Skip delimiters specified in the PARMS list, white space
31017                                   ;       and comma.
31018                                   ;
31019                                   ; Input:    DS:SI -> Command String
31020                                   ;       ES:DI -> Parameter List
31021                                   ;
31022                                   ; Output:   CY = 1 if the end of line encounterd
31023                                   ;       CY = 0 then SI move to 1st non-delimiter character
31024                                   ;       AL = Last examineed character
31025                                   ;
31026                                   ; Use:          _$P_Chk_EOL, _$P_Chk_Delim,
31027                                   ;
31028                                   ; Vars:     _$P_Flags(R)
31029                                   ;******************************************************************************
31030
31031                                   _$P_Skip_Delim:
31032                                   _$P_Skip_Delim_Loop:                    ;AN000;
31033 000020C1 AC                         lodsb                           ;AN000;
31034 000020C2 E81E00                     call    _$P_Chk_EOL             ;AN000; is it EOL character ?
31035 000020C5 7416                       jz      short _$P_Skip_Delim_CY       ;AN000; if yes, exit w/ CY on
31036
31037 000020C7 E84E00                     call    _$P_Chk_Delim           ;AN000; is it one of delimiters ?
31038 000020CA 7514                       jnz     short _$P_Skip_Delim_NCY ;AN000; if no, exit w/ CY off
31039
31040 000020CC 2EF606[7D19]20             test    byte [cs:_$P_Flags2],_$P_Extra ;AC034; extra delim or comma found ?
31041 000020D2 74ED                       jz      short _$P_Skip_Delim_Loop ;AN000; if no, loop
31042
31043 000020D4 2EF606[7D19]41             test    byte [cs:_$P_Flags2],_$P_SW+_$P_equ ;AC034; /x , or xxx=zzz , (tm08)
31044                                     ;jz     short _$P_Exit_At_Extra       ;AN000; no switch, no keyword (tm08)
31045                                     ; 08/07/2023
31046                                     ; cf=0
31047 000020DA 7505                       jnz     short _$P_Skip_Delim_Exit
31048 000020DC C3                         retn
31049
31050                                     ;dec    si                      ;AN000; backup si for next call (tm08)
31051                                     ;jmp    short _$P_Exit_At_Extra       ;AN000; else exit w/ CY off
31052                                     ; 12/12/2022
31053                                     ; cf=0
31054                                     ; 08/07/2023
31055                                     ;jmp    short _$P_Skip_Delim_Exit
31056
31057                                   _$P_Skip_Delim_CY:                ;AN000;
31058 000020DD F9                         stc                             ;AN000; indicate EOL
31059 000020DE EB01                       jmp     short _$P_Skip_Delim_Exit ;AN000;
31060
31061                                   _$P_Skip_Delim_NCY:                     ;AN000;
31062 000020E0 F8                         clc                             ;AN000; indicate non delim
31063                                   _$P_Skip_Delim_Exit:                    ;AN000; in this case, need
31064 000020E1 4E                         dec     si                      ;AN000;  backup index pointer
31065                                     ; 08/07/2023
31066                                     ; 12/12/2022
31067                                   ;_$P_Exit_At_Extra:         ; cf=0
31068 000020E2 C3                         retn                            ;AN000;
31069
31070                                     ; 12/12/2022
31071                                   ;_$P_Exit_At_Extra:                     ;AN000;
31072                                     ;clc                            ;AN000; indicate extra delim
31073                                     ;retn                           ;AN000;
31074
31075                                   ;******************************************************************************
31076                                   ; _$P_Chk_EOL;
31077                                   ;
31078                                   ; Function: Check if AL is one of End of Line characters.
31079                                   ;
31080                                   ; Input:    AL = character code
31081                                   ;       ES:DI -> Parameter List
31082                                   ;
31083                                   ; Output:   ZF = 1 if one of End of Line characters
31084                                   ;******************************************************************************
31085
31086                                   _$P_Chk_EOL:
31087 000020E3 53                         push    bx                      ;AN000;
31088 000020E4 51                         push    cx                      ;AN000;
31089 000020E5 3C0D                       cmp     al,_$P_CR               ;AN000; Carriage return ?
31090 000020E7 742C                       je      short _$P_Chk_EOL_Exit        ;AN000;
31091 000020E9 3C00                       cmp     al,_$P_NULL             ;AN000; zero ?
31092 000020EB 7428                       je      short _$P_Chk_EOL_Exit        ;AN000;
31093                                   ;IF LFEOLSW                         ;AN028; IF LF TO BE ACCEPTED AS EOL
31094 000020ED 3C0A                       cmp     al,_$P_LF               ;AN000; Line feed ?
31095 000020EF 7424                       je      short _$P_Chk_EOL_Exit        ;AN000;
31096                                   ;ENDIF                              ;AN028;
31097 000020F1 26807D0202                 cmp     byte [es:di+_$P_PARMS_Blk.Num_Extra],_$P_I_Have_EOL
31098                                                                     ;AN000; EOL character specified ?
31099 000020F6 721D                       jb      short _$P_Chk_EOL_Exit        ;AN000;
31100 000020F8 31DB                       xor     bx,bx                   ;AN000;
31101 000020FA 268A5D03                   mov     bl,[es:di+_$P_PARMS_Blk.Len_Extra_Delim]
31102                                                                     ;AN000; get length of delimiter list
31103 000020FE 83C304                     add     bx,_$P_Len_PARMS        ;AN000; skip it
```

```
                                    ; 08/07/2023
00002101 31C9                       xor     cx,cx ; *
00002103 26803900                   cmp     byte [es:bx+di],_$P_I_Use_Default ;AN000; No extra EOL character ?
00002107 740B                       je      short _$P_Chk_EOL_NZ  ;AN000;
                                    ; 08/07/2023
                                    ;;xor   cx,cx                 ;AN000; Get number of extra character
                                    ;xor    ch,ch ; *
00002109 268A09                     mov     cl,[es:bx+di]         ;AN000;
                                _$P_Chk_EOL_Loop:                 ;AN000;
0000210C 43                         inc     bx                    ;AN000;
0000210D 263A01                     cmp     al,[es:bx+di]         ;AN000; Check extra EOL character
00002110 7403                       je      short _$P_Chk_EOL_Exit        ;AN000;
00002112 E2F8                       loop    _$P_Chk_EOL_Loop      ;AN000;
                                    ; 08/07/2023
                                    ; cx=0
                                _$P_Chk_EOL_NZ:                           ;AN000;
                                    ;cmp    al,_$P_CR             ;AN000; reset ZF
                                    ; 08/07/2023
00002114 41                         inc     cx  ; zf=0  (cx=1) ; *
                                _$P_Chk_EOL_Exit:                 ;AN000;
00002115 59                         pop     cx                    ;AN000;
00002116 5B                         pop     bx                    ;AN000;
00002117 C3                         retn

                                ;****************************************************************************
                                ; _$P_Chk_Delim;
                                ;
                                ; Function: Check if AL is one of delimiter characters.
                                ;       if AL+[si] is DBCS blank, it is replaced with two SBCS
                                ;       blanks.
                                ;
                                ; Input:    AL = character code
                                ;       DS:SI -> Next Character
                                ;       ES:DI -> Parameter List
                                ;
                                ; Output:   ZF = 1 if one of delimiter characters
                                ;       SI points to the next character
                                ; Vars:  _$P_Terminator(W), _$P_Flags(W)
                                ;****************************************************************************

                                    ; 08/07/2023 - Retro DOS v4.2 IO.SYS (optimization)
                                    ; MSDOS 6.21 IO.SYS - SYSINIT:1FAEh
                                    ; (PCDOS 7.1 IBMBIO.COM - SYSINIT:2451h) ; (Retro DOS v5.0)

                                _$P_Chk_Delim:
00002118 53                         push    bx                    ;AN000;
00002119 51                         push    cx                    ;AN000;
0000211A 2EC606[7719]20             mov     byte [cs:_$P_Terminator],_$P_Space
                                                                  ;AC034; Assume terminated by space
                                    ;and    byte [cs:_$P_Flags20,0DFh
00002120 2E8026[7D19]DF             and     byte [cs:_$P_Flags2],0FFh-_$P_Extra ;AC034;
00002126 3C20                       cmp     al,_$P_Space ; 20h    ;AN000; Space ?
00002128 7423                       je      short _$P_Chk_Delim_Exit ;AN000;

0000212A 3C09                       cmp     al,_$P_TAB            ;AN000; TAB ?
0000212C 741F                       je      short _$P_Chk_Delim_Exit ;AN000;

0000212E 3C2C                       cmp     al,_$P_Comma          ;AN000; Comma ?
00002130 741E                       je      short _$P_Chk_Delim_Exit0 ;AN000;

                                ; Note: _$P_Chk_Delim00 part of code is nonsense here
                                ;       because _$P_Space = _$P_DBSP1 = 20h
                                ;       Erdogan Tan - 08/07/2023 - Retro DOS v4.2 IO.SYS (optimization)
                                ;_$P_Chk_Delim00:
                                    ; (PCDOS 7.1 IBMBIO.COM - SYSINIT:246Bh)
                                    ; (MSDOS 6.21 IO.SYS - SYSINIT:1FC8h)
                                %if 0
                                    ; 26/10/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
                                _$P_Chk_Delim00:                  ;AN000;
                                    cmp     al,_$P_DBSP1   ; 20h  ;AN000; 1st byte of DBCS Space ?
                                    jne     short _$P_Chk_Delim01 ;AN000;

                                    cmp     byte [si],_$P_DBSP2 ; 20h ;AN000; 2nd byte of DBCS Space ?
                                    jne     short _$P_Chk_Delim01 ;AN000;

                                    mov     al,_$P_Space          ;AN000;
                                    inc     si                    ;AN000; make si point to next character
                                    cmp     al,al                 ;AN000; Set ZF
                                    jmp     short _$P_Chk_Delim_Exit ;AN000;
                                %endif

                                _$P_Chk_Delim01:                  ;AN000;
00002132 26807DFE01                 cmp     byte [es:di-_$P_PARMS_Blk.Num_Extra],_$P_I_Have_Delim
                                                                  ;AN000; delimiter character specified ?
00002137 7214                       jb      short _$P_Chk_Delim_Exit ;AN000;

                                    ;xor    cx,cx                 ;AN000;
00002139 30ED                       xor     ch,ch
                                    ;mov    cl,[es:di+3]
0000213B 268A4D03                   mov     cl,[es:di+_$P_PARMS_Blk.Len_Extra_Delim]
                                                                  ;AN000; get length of delimiter list
                                    ;or     cx,cx                 ;AN000; No extra Delim character ?
                                    ;jz     short _$P_Chk_Delim_NZ        ;AN000;
                                    ; 08/07/2023
0000213F E30B                       jcxz    _$P_Chk_Delim_NZ

00002141 BB0300                     mov     bx,_$P_Len_PARMS-1 ; 3;AN000; set bx to 1st extra delimiter
                                _$P_Chk_Delim_Loop:                       ;AN000;
00002144 43                         inc     bx                    ;AN000;
00002145 263A01                     cmp     al,[es:bx+di]         ;AN000; Check extra Delim character
00002148 7406                       je      short _$P_Chk_Delim_Exit0 ;AN000;

0000214A E2F8                       loop    _$P_Chk_Delim_Loop    ;AN000; examine all extra delimiter

                                _$P_Chk_Delim_NZ:                 ;AN000;
                                    ;cmp    al,_$P_Space          ;AN000; reset ZF
                                    ; 08/07/2023
                                    ; cx=0 here
0000214C 41                         inc     cx ; cx=1, zf=0
                                _$P_Chk_Delim_Exit:                       ;AN000;
                                _$P_ChkDfin:                      ;AN000;
0000214D 59                         pop     cx                    ;AN000;
0000214E 5B                         pop     bx                    ;AN000;
0000214F C3                         retn                          ;AN000;

                                _$P_Chk_Delim_Exit0:                      ;AN000;
00002150 2EA2[7719]                 mov     [cs:_$P_Terminator],al ;AC034; keep terminated delimiter
00002154 2EF606[7D19]01             test    byte [cs:_$P_Flags2],_$P_equ ;AN027;AC034;; if terminating a key=
0000215A 7506                       jnz     short _$P_No_Set_Extra        ;AN027; then do not set the EXTRA bit

0000215C 2E800E[7D19]20             or      byte [cs:_$P_Flags2],_$P_Extra
                                                                  ;AC034; flag terminated extra delimiter or comma
                                _$P_No_Set_Extra:                 ;AN027;
00002162 38C0                       cmp     al,al                 ;AN000; set ZF
```

```
00002164 EBE7                       jmp     short _$P_Chk_Delim_Exit ;AN000;

                                ;*****************************************************************************
                                ; _$P_Chk_Switch;
                                ;
                                ; Function: Check if AL is the switch character not in first position of
                                ;       _$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:
00002205 0000                       dw      0
00002207 0000                       dw      0
00002209 [1722]                     dw      result_val
0000220B [1622]                     dw      noval
0000220D 01                         db      1       ; 1 switch

                                switch_x:
0000220E 2F5800                     db      '/X',0          ; M016

                                p_buffers:
00002211 0000                       dw      0       ; local variables
                                p_h_buffers:
00002213 0000                       dw      0
                                    ; 26/10/2022  (MSDOS 5.0 IO.SYS SYSINIT compatibility)
                                p_buffer_slash_x:
00002215 00                         db      0 ; 31/03/2019

                                ;-- common definitions ---------------------------------------------------

00002216 00                     noval:      db      0

                                result_val:         ;label  byte
00002217 00                         db      0               ; type returned
                                result_val_itag:
00002218 00                         db      0               ; item tag returned
                                result_val_swoff:
00002219 0000                       dw      0               ; es:offset of the switch defined
                                rv_byte:    ;label  byte
0000221B 00000000               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:
0000221F [2422]                     dw      brk_parmsx
00002221 01                         db      1       ; an extra delimiter list
00002222 01                         db      1       ; length is 1
00002223 3B                         db      ';'     ; delimiter

                                brk_parmsx:
00002224 0101[2A22]                 dw      101h,brk_pos    ; min,max = 1 positional
00002228 00                         db      0               ; no switches
00002229 00                         db      0               ; no keywords

                                ;brk_pos p_pos <2000h,0,result_val,on_off_string> ; simple string
                                brk_pos:
0000222A 0020                       dw      2000h
0000222C 0000                       dw      0
0000222E [1722]                     dw      result_val
00002230 [3322]                     dw      on_off_string
00002232 00                         db      0

                                on_off_string:      ;label  byte
00002233 03                         db      3               ; signals that there is a string choice
00002234 00                         db      0               ; no range definition
00002235 00                         db      0               ; no numeric values choice
00002236 02                         db      2               ; 2 strings for choice
00002237 01                         db      1               ; the 1st string tag
00002238 [3D22]                     dw      on_string
0000223A 02                         db      2               ; the 2nd string tag
0000223B [4022]                     dw      off_string

                                on_string:
0000223D 4F4E00                     db      "ON",0
                                off_string:
00002240 4F464600                   db      "OFF",0

                                p_ctrl_break:
00002244 00                         db      0       ; local variable

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

                                    ; 27/10/2022

                                ; country = n {m {path}}
                                ; or
                                ; country = n,,path

                                ;cntry_parms p_parms <cntry_parmsx>
                                cntry_parms:
00002245 [4A22]                     dw      cntry_parmsx
00002247 01                         db      1
00002248 01                         db      1
00002249 3B                         db      ';'

                                cntry_parmsx:
0000224A 0103[5422][6822]-          dw      301h,cntry_pos1,cntry_pos2,cntry_pos3 ; min 1, max 3 pos.
00002250 [7122]
00002252 00                         db      0               ; no switches
00002253 00                         db      0               ; no keywords

                                ;cntry_pos1 p_pos <8000h,0,result_val,cc_range> ; numeric value
                                cntry_pos1:
00002254 0080                       dw      8000h
00002256 0000                       dw      0
00002258 [1722]                     dw      result_val
0000225A [5D22]                     dw      cc_range
0000225C 00                         db      0

                                ;cc_range p_range <,,,1,999>
                                cc_range:
0000225D 01                         db      1
0000225E 01                         db      1
0000225F 01                         db      1
00002260 01000000                   dd      1
00002264 E7030000                   dd      999

                                ;cntry_pos2 p_pos <8001h,0,result_val,cc_range> ; optional num.
                                cntry_pos2:
00002268 0180                       dw      8001h
0000226A 0000                       dw      0
0000226C [1722]                     dw      result_val
0000226E [5D22]                     dw      cc_range
00002270 00                         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
```

```
000022E6 [1622]                     dw      noval
000022E8 00                         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]
```

```
31846 00002355 01                       db      1
31847 00002356 2F4B00                   swit_k:     db      '/K',0
31848
31849                                   ; 01/01/2023 - Retro DOS v4.2 (MSDOS 6.21 IO.SYS)
31850                                   ; (SYSINIT:220Ch) ; *
31851
31852                                   ; 27/10/2022 - Retro DOS v4.0 (MSDOS 5.0 IO.SYS, SYSINIT)
31853                                   ;
31854                                   ;swit_n_ctrl p_pos <0,0,result_val,noval,1> ; switch string follows
31855                                   swit_n_ctrl: ; *
31856 00002359 00000000[1722]-              dw      0,0,result_val,noval
31856 0000235F [1622]
31857 00002361 01                       db      1
31858 00002362 2F4E00                   swit_n: db  '/N',0
31859
31860                                   ;swit_f_ctrl p_pos <0,0,result_val,noval,1> ; switch string follows
31861                                   swit_f_ctrl: ; *
31862 00002365 00000000[1722]-              dw      0,0,result_val,noval
31862 0000236B [1622]
31863 0000236D 01                       db      1
31864 0000236E 2F4600                   swit_f: db  '/F',0
31865
31866                                       ; 27/10/2022
31867
31868                                   ;swit_t_ctrl p_pos <0,0,result_val,noval,1> ; switch string follows   M059
31869                                   swit_t_ctrl:
31870 00002371 00000000[1722]-              dw      0,0,result_val,noval
31870 00002377 [1622]
31871 00002379 01                       db      1
31872 0000237A 2F5400                   swit_t:     db      '/T',0                  ;                       M059
31873                                   ;swit_w_ctrl p_pos <0,0,result_val,noval,1> ; switch string follows   M063
31874                                   swit_w_ctrl:
31875 0000237D 00000000[1722]-              dw      0,0,result_val,noval
31875 00002383 [1622]
31876 00002385 01                       db      1
31877 00002386 2F5700                   swit_w:     db      '/W',0                  ;                       M063
31878
31879                                   ; 14/04/2024 - Retro DOS v5.0 (PCDOS 7.1 IBMBIO.COM, SYSINIT)
31880                                   ;;;
31881                                   ;swit_i_ctrl p_pos <0,0,result_val,noval,1> ; switch string follows
31882                                   swit_i_ctrl:
31883 00002389 0000                         dw      0
31884 0000238B 0000                         dw      0
31885 0000238D [1722]                       dw      result_val
31886 0000238F [1622]                       dw      noval
31887 00002391 01                           db      1
31888 00002392 2F4900                   swit_i:     db      '/I',0
31889                                   ;;;
31890
31891                                   ;   There doesn't need to be p_swit_n or p_swit_f because /N and /F are
31892                                   ;   acted upon during MULTI_CONFIG processing; we only needed entries
31893                                   ;   in the above table to prevent the parsing code from complaining about them
31894
31895 00002395 00                       p_swit_k:   db      0       ; local variable
31896 00002396 00                       p_swit_t:   db      0       ; local variable                    M059
31897 00002397 00                       p_swit_w:   db      0       ; local variable                    M063
31898                                   ; 14/04/2024 - Retro DOS v5.0 (PCDOS 7.1 IBMBIO.COM)
31899 00002398 00                       p_swit_i:   db      0
31900
31901                                   ;--------------------------------------------------------------------------
31902
31903                                       ; 27/10/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 IO.SYS)
31904                                       ; (SYSINIT:20E8h)
31905
31906                                   ; DOS = [ high | low ]
31907
31908                                   ;dos_parms p_parms  <dos_parmsx>
31909                                   dos_parms:
31910 00002399 [9E23]                       dw      dos_parmsx
31911 0000239B 01                           db      1
31912 0000239C 01                           db      1
31913 0000239D 3B                           db      ';'
31914                                   dos_parmsx:
31915 0000239E 01                           db      1               ; min parameters
31916 0000239F 02                           db      2               ; max parameters
31917 000023A0 [A623]                       dw      dos_pos         ;
31918 000023A2 [A623]                       dw      dos_pos         ;
31919 000023A4 00                           db      0               ; no switches
31920 000023A5 00                           db      0               ; no keywords
31921
31922                                   ;dos_pos p_pos     <2000h,0,result_val,dos_strings> ; simple string
31923                                   ;        p_pos     <2000h,0,result_val,dos_strings> ; simple string
31924                                   dos_pos:
31925 000023A6 00200000[1722]-              dw      2000h,0,result_val,dos_strings
31925 000023AC [B823]
31926 000023AE 00                           db      0
31927 000023AF 00200000[1722]-              dw      2000h,0,result_val,dos_strings
31927 000023B5 [B823]
31928 000023B7 00                           db      0
31929
31930                                   dos_strings:        ;label  byte
31931 000023B8 03                           db      3               ; signals that there is a string choice
31932 000023B9 00                           db      0               ; no range definition
31933 000023BA 00                           db      0               ; no numeric values choice
31934 000023BB 04                           db      4               ; 4 strings for choice
31935 000023BC 01                           db      1               ; the 1st string tag
31936 000023BD [E623]                       dw      hi_string
31937 000023BF 02                           db      2               ; the 2nd string tag
31938 000023C0 [EB23]                       dw      lo_string
31939 000023C2 03                           db      3
31940 000023C3 [EF23]                       dw      umb_string
31941 000023C5 04                           db      4
31942 000023C6 [F323]                       dw      noumb_string
31943
31944                                   ; 14/04/2024 - Retro DOS v5.0
31945                                   ; (PCDOS 7.1 IBMDOS.COM - SYSINIT:273Eh)
31946                                   ;;;
31947                                   dosdata_parms:
31948 000023C8 [CD23]                       dw      dosdata_parmsx ; DOSDATA = UMB|NOUMB
31949 000023CA 01                           db      1
31950 000023CB 01                           db      1
31951 000023CC 3B                           db      ';'
31952                                   dosdata_parmsx:
31953 000023CD 01                           db      1
31954 000023CE 01                           db      1               ; min,max = 1 positional
31955 000023CF [D323]                       dw      dosdata_pos
31956 000023D1 00                           db      0               ; no switches
31957 000023D2 00                           db      0               ; no keywords
31958
31959                                       ; dosdata_pos p_pos <2000h,0,result_val,dosdata_strings>
31960                                   dosdata_pos:
31961 000023D3 0020                         dw      2000h           ; simple string
31962 000023D5 0000                         dw      0
31963 000023D7 [1722]                       dw      result_val
```

```
31964 000023D9 [DC23]                      dw      dosdata_strings
31965 000023DB 00                          db      0
31966                                   dosdata_strings:
31967 000023DC 03                          db      3               ; signals that there is a string choice
31968 000023DD 00                          db      0               ; no range definition
31969 000023DE 00                          db      0               ; no numeric values choice
31970 000023DF 02                          db      2               ; 2 strings for choice
31971 000023E0 01                          db      1               ; the 1st string tag
31972 000023E1 [EF23]                      dw      umb_string      ; "UMB"
31973 000023E3 02                          db      2               ; the 2nd string tag
31974 000023E4 [F323]                      dw      noumb_string    ; "NOUMB"
31975                                   ;;;
31976
31977 000023E6 4849474800               hi_string:  db      "HIGH",0
31978 000023EB 4C4F5700                 lo_string:  db      "LOW",0
31979 000023EF 554D4200                 umb_string: db      "UMB",0
31980 000023F3 4E4F554D4200             noumb_string:       db      "NOUMB",0
31981
31982                                   p_dos_hi:
31983 000023F9 00                          db      0       ; local variable
31984                                                      ; BUGBUG : I dont know whether PARSER uses
31985                                                      ;          this variable or not
31986                                      ; 14/04/2024 (PCDOS 7.1 IBMBIO.COM)
31987 000023FA 00                          db      0
31988
31989                                   ; 01/01/2023 - Retro DOS v4.2 (Modified MSDOS 6.21 IO.SYS)
31990                                   ; 27/10/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 IO.SYS)
31991                                   ;%if 0
31992
31993                                   ;************************************************************** RICHID ****
31994
31995                                   ;include    highvar.inc    ; devicehigh variables (used by loadhigh also)
31996
31997                                   ; 30/03/2019 - Retro DOS v4.0
31998                                   ;--------------------------------------------------------------------------
31999
32000                                   ;   Module:   HIGHVAR.INC - Data common to LOADHIGH and DEVICEHIGH, res seg
32001                                   ;
32002                                   ;   Date:     May 14, 1992
32003                                   ;
32004                                   ;**************************************************************************
32005                                   ;
32006                                   ;   Modification log:
32007                                   ;
32008                                   ;     DATE    WHO      DESCRIPTION
32009                                   ;   --------  -------  -------------------------------------------------
32010                                   ;   05/14/92  t-richj  Original
32011                                   ;   06/21/92  t-richj  Final revisions before check-in
32012                                   ;
32013                                   ;**************************************************************************
32014                                   ;
32015                                   ; There are two primary definitions which need to be made, selectively, before
32016                                   ; this include file should be used.  These are:
32017                                   ;    HV_Extern - If this has been defined, variables for this module will be
32018                                   ;                declared as external.  Otherwise, variables will be declared
32019                                   ;                public, as well as defined, here.  LoadHigh declares HV_Extern
32020                                   ;                in stub.asm and loadhi.asm, and does not declare it in
32021                                   ;                rdata.asm... DeviceHigh does not declare HV_Extern anywhere
32022                                   ;                (as only one module, sysconf.asm, includes this file).
32023                                   ;    HV_LoadHigh - This should be defined when this module is going into
32024                                   ;                  command.com, for LoadHigh.  All of loadhi.asm, stub.asm and
32025                                   ;                  rdata.asm define this, while io.sys' sysconf.asm does not.
32026                                   ;
32027                                   ;**************************************************************************
32028
32029                                   ; To keep track of which UMBs were specified on the DH/LH command lines, and
32030                                   ; to keep track of the minimum sizes given for each, there're two arrays kept
32031                                   ; in { IO.SYS: sysinitseg / COMMAND.COM: DATARES }... each is MAXUMB elements
32032                                   ; big.  16 should be around 14 too many for most users, so there's no expected
32033                                   ; space problem (it's just such a nice round number, eh?).
32034
32035                                   MAXUMB      equ     16
32036
32037                                   ; Memory elements owned by the system are marked as PSP address 8 in both the
32038                                   ; USA and Japan; Japanese systems also use 9 under more bizzarre conditions.
32039
32040                                   FreePSPOwner        equ     0       ; Free MCBs all have an owner PSP address of 0
32041                                   SystemPSPOwner      equ     8
32042                                   ;JapanPSPOwner      equ     9
32043
32044                                   ; for LoadHigh and DeviceHigh:
32045                                   ;
32046                                   ;   fInHigh - Is set to 1 during HideUMBs(), and back to zero in
32047                                   ;             UnHideUMBs().
32048                                   ;   fUmbTiny - Is set to 1 iff the user has specified /S on the command-
32049                                   ;              line.
32050                                   ;   SegLoad - Segment address for first UMB specified; set automatically.
32051                                   ;   UmbLoad - The load UMB number; for example, this is 3 if the user has
32052                                   ;             given a command-line like "/L:3,500;4"
32053                                   ;   UmbUsed - An array of characters, each of which is 1 iff the UMB
32054                                   ;             matching its index number was specified on the command-line;
32055                                   ;             for example, after "/L:3,500;4;7", UmbUsed[3], [4] and [7]
32056                                   ;             will be set to 1.  All others will be set to 0.
32057                                   ;   UmbSize - An array of words, each of which is interpereted as a size
32058                                   ;             specified by the user for a UMB (in the above example, all
32059                                   ;             elements would be zero save UmbSize[3], which would be 500.
32060                                   ;   fm_umb - Set to the old UMB link-state (0x80 or 0x00)
32061                                   ;   fm_strat - Set to the old memory-allocation strategy (0$00000???)
32062                                   ;   fm_argc  - Number of arguments received by ParseVar() (see ParseVar()
32063                                   ;              for details).
32064
32065 000023FB 00                       fInHigh:  db        0
32066 000023FC 00                       fUmbTiny: db        0
32067 000023FD 0000                     SegLoad:  dw        0
32068 000023FF 00                       UmbLoad:  db        0
32069 00002400 00<rep 10h>              UmbUsed:  times MAXUMB db 0 ; times 16 db 0  ; db 16 dup(?)
32070 00002410 0000<rep 10h>            UmbSize:  times MAXUMB dw 0 ; times 16 dw 0  ; dw 16 dup(?)
32071 00002430 00                       fm_umb:   db        0
32072 00002431 00                       fm_strat: db        0
32073 00002432 00                       fm_argc:  db        0
32074
32075                                   ; UmbLoad is set to UNSPECIFED, below, until /L:umb is read; at which point
32076                                   ; UmbLoad is set to the UMB number given.
32077
32078                                   UNSPECIFIED equ     -1
32079
32080                                   ;%endif ; 27/10/2022
32081
32082                                   ;************************************************************** RICHID ****
32083
32084                                   ; 30/03/2019 - Retro DOS v4.0 (MSDOS 6.0, SYSCONF.ASM)
32085                                   ; ((MSDOS 6.21 IO.SYS -> SYNINIT:22BAh))
32086
32087                                   ; 27/10/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 IO.SYS)
```

```
                                ; (SYSINIT:212Bh)
                                
                                        ;public DevEntry
                                
00002433 0000                   DevSize:    dw      0       ; size of the device driver being loaded(paras)
00002435 0000                   DevLoadAddr:        dw      0       ; Mem addr where the device driver is 2 b loaded
00002437 0000                   DevLoadEnd: dw      0       ; MaxAddr to which device can be loaded
00002439 00000000               DevEntry:   dd      0       ; Entry point to the device driver
0000243D 00000000               DevBrkAddr: dd      0       ; Break address of the device driver
                                ; 30/12/2022
                                ; 27/10/2022
00002441 00                     ConvLoad:   db      0       ; Use conventional (dos 5 -style) InitDevLoad?
                                ;
00002442 00                     DevUMB:             db      0       ; byte indicating whether to load DDs in UMBs
00002443 0000                   DevUMBAddr: dw      0       ; cuurent UMB used fro loading devices (paras)
00002445 0000                   DevUMBSize: dw      0       ; Size of the current UMB being used   (paras)
00002447 0000                   DevUMBFree: dw      0       ; Start of free are in the current UMB (paras)
                                ;
00002449 00000000               DevXMSAddr: dd      0
                                ;
0000244D 0000                   DevExecAddr:        dw      0       ; Device load address parameter to Exec call
0000244F 0000                   DevExecReloc:       dw      0       ; Device load relocation factor
                                ;
00002451 00                     DeviceHi:   db      0       ; Flag indicating whther the current device
                                                            ;  is being loaded into UMB
00002452 0000                   DevSizeOption:      dw      0       ; SIZE= option
                                ;
00002454 00                     Int12Lied:  db      0       ; did we trap int 12 ?
00002455 0000                   OldInt12Mem:        dw      0       ; value in 40:13h (int 12 ram)
00002457 50524F544D414E24       ThreeComName:       db      'PROTMAN$'      ; 3Com Device name
                                ;
0000245F 00                     FirstUMBLinked:     db      0
00002460 0000                   DevDOSData: dw      0       ; segment of DOS Data
00002462 00000000               DevCmdLine: dd      0       ; Current Command line
00002466 00                     DevSavedDelim:      db      0       ; The delimiter which was replaced with null
                                                            ; to use the file name in the command line
                                ; 13/04/2024 - Retro DOS v5.0 (PCDOS 7.1 IBMBIO.COM)
;   ifdef   dblspace_hooks
00002467 00                     MagicHomeFlag:      db      0       ; set non-zero when MagicDrv is final placed
;   endif

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

                                ; 31/03/2019 - Retro DOS v4.0

                                ; 27/10/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 IO.SYS)
                                ; (SYSINIT:215Eh)

                                ;--------------------------------------------------------------------------------
                                ;
                                ; procedure : doconf
                                ;
                                ;             Config file is parsed initially with this routine. For the
                                ;             Subsequent passes 'multi_pass' entry is used .
                                ;
                                ;--------------------------------------------------------------------------------

                                    ; 27/10/2022
                                doconf:
0000246A 0E                         push    cs
0000246B 1F                         pop     ds

0000246C B80037                     mov     ax,3700h
                                        ;mov       ax,(CHAR_OPER<<8)     ; get switch character
0000246F CD21                       int     21h
00002471 8816[BC4B]                 mov     [command_line+1],dl   ; set in default command line

                                ; 31/12/2022 - Retro DOS v4.2 (Modified MSDOS 6.21  IO.SYS)
                                ; 27/10/2022
                                ;;ifdef     MULTI_CONFIG
                                ;   ;mov    [command_line-1],dl     ; save default switchchar
00002475 8816[BA4B]                 mov     [def_swchr],dl ; 31/03/2019
                                ;;endif     ;MULTI_CONFIG

00002479 BA[D14A]                   mov     dx,config ;'\CONFIG.SYS' ;now pointing to file description
0000247C B8003D                     mov     ax,3D00h
                                    ;mov    ax,OPEN<<8              ;open file "config.sys"
0000247F F9                         stc                             ;in case of int 24
00002480 CD21                       int     21h                     ;function request
00002482 7309                       jnc     short noprob            ; brif opened okay

                                ; 31/12/2022
                                ; 27/10/2022
                                ;;ifdef     MULTI_CONFIG
00002484 E8A019                     call    kbd_read                ; we still want to give the guy
                                ;                                   ; a chance to select clean boot!
                                ;;endif                                  ; (ie, no autoexec.bat processing)
00002487 C606[CD02]0B               mov     byte [multi_pass_id],11     ; set it to unreasonable number
0000248C C3                         retn
                                noprob:                             ;get file size (note < 64k!!)
0000248D 89C3                       mov     bx,ax  ; File handle
0000248F 31C9                       xor     cx,cx                   ; 0
00002491 31D2                       xor     dx,dx                   ; 0
                                    ;mov    ax,4202h
00002493 B80242                     mov     ax,(LSEEK<<8)|2
00002496 CD21                       int     21h
00002498 A3[5603]                   mov     [count],ax              ; dx:ax = file size ; 08/09/2023
                                                                    ; 08/09/2023 - Erdogan Tan - Note:
0000249B 31D2                       xor     dx,dx                   ; dx already must be 0 here ; 08/09/2023
                                                                    ; I am not removing 'xor dx,dx' here
                                                                    ; for MSDOS compatibility.
                                                                    ; ((Also PCDOS 7.1 has 'xor dx,dx' here))
                                                                    ; (Error will be same if CONGIG.SYS file
                                                                    ;  size > 64KB)
                                    ;mov    ax,4200h
0000249D B80042                     mov     ax,LSEEK<<8             ;reset pointer to beginning of file
000024A0 CD21                       int     21h

                                    ; 31/12/2022 - Retro DOS v4.2
000024A2 8B16[A502]                 mov     dx,[ALLOCLIM]           ;use current alloclim value
                                    ; 27/10/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
                                    ;mov    dx,[top_of_cdss]

000024A6 A1[5603]                   mov     ax,[count]
000024A9 A3[D002]                   mov     [config_size],ax        ;save the size of config.sys file.
000024AC E868EE                     call    ParaRound
000024AF 29C2                       sub     dx,ax

                                ; 31/12/2022
                                ; 27/10/2022
                                ;ifdef     MULTI_CONFIG
                                ;
                                ;  The size of the CONFIG.SYS workspace (for recreating the in-memory
                                ;  CONFIG.SYS image, and later for building the initial environment) need
```

```
32212                                   ;  not be any larger than CONFIG.SYS itself, EXCEPT for the fact that
32213                                   ;  we (may) add a variable to the environment that does not explicity appear
32214                                   ;  in CONFIG.SYS, and that variable is CONFIG (as in CONFIG=COMMON).
32215                                   ;  The default setting for CONFIG cannot result in more than 1 paragraph
32216                                   ;  of extra space, so here we account for it (the worst case of course is
32217                                   ;  when CONFIG.SYS is some very small size, like 0 -JTP)
32218                                   ;
32219 000024AF 4A                          dec     dx                      ;reserve 1 additional paragraph
32220 000024B0 8916[6219]                mov     [config_wrkseg],dx      ;this is the segment to be used for
32221 000024B4 29C2                      sub     dx,ax                   ;rebuilding the config.sys memory image
32222                                   ;;endif     ;MULTI_CONFIG
32223
32224 000024B6 83EA11                    sub     dx,11h                  ;room for header
32225
32226                                   ; 31/12/2022
32227 000024B9 8916[A502]                mov     [ALLOCLIM],dx           ;config starts here. new alloclim value.
32228 000024BD 8916[A302]                mov     [CONFBOT],dx
32229
32230                                   ; 27/10/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
32231                                   ;mov    [top_of_cdss],dx
32232                                   ;call    TempCDS
32233                                   ; 31/12/2022
32234                                   ; 11/12/2022
32235                                   ; ds <> cs
32236                                   ;mov    dx,[cs:top_of_cdss]
32237
32238                                   ; 08/09/2023
32239                                   ; ds = cs
32240 000024C1 8B0E[5603]                mov     cx,[count]
32241
32242 000024C5 8EDA                      mov     ds,dx
32243 000024C7 8EC2                      mov     es,dx
32244
32245 000024C9 31D2                      xor     dx,dx
32246                                   ; 08/09/2023
32247                                   ;mov    cx,[cs:count]
32248 000024CB B43F                      mov     ah,3Fh
32249                                   ;mov    ah,READ  ; 3Fh
32250 000024CD F9                        stc                             ;in case of int 24
32251 000024CE CD21                      int     21h                     ;function request
32252 000024D0 9C                        pushf
32253
32254                                   ; find the eof mark in the file. if present,then trim length.
32255
32256 000024D1 50                        push    ax
32257 000024D2 57                        push    di
32258 000024D3 51                        push    cx
32259 000024D4 B01A                      mov     al,1Ah                  ; eof mark
32260 000024D6 89D7                      mov     di,dx                   ; point to buffer
32261 000024D8 E305                      jcxz    puteol                  ; no chars
32262 000024DA F2AE                      repnz   scasb                   ; find end
32263 000024DC 7501                      jnz     short puteol            ; none found and count exhausted
32264
32265                                   ; we found a 1a. back up
32266
32267 000024DE 4F                        dec     di                      ; backup past 1Ah
32268
32269                                   ;  just for the halibut, stick in an extra eol
32270
32271                                   puteol:
32272 000024DF B00D                      mov     al,cr ; 0Dh
32273 000024E1 AA                        stosb
32274 000024E2 B00A                      mov     al,lf  ;0Ah
32275 000024E4 AA                        stosb
32276 000024E5 29D7                      sub     di,dx                   ; difference moved
32277                                   ; 27/10/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
32278                                   ;mov    [cs:count],di           ; new count
32279
32280                                   ; 11/12/2022
32281                                   ; 31/03/2019 - Retro DOS v4.0
32282 000024E7 0E                        push    cs
32283 000024E8 1F                        pop     ds
32284
32285 000024E9 893E[5603]                mov     [count],di              ; new count
32286
32287 000024ED 59                        pop     cx
32288 000024EE 5F                        pop     di
32289 000024EF 58                        pop     ax
32290
32291                                   ; 11/12/2022
32292                                   ; 27/10/2022
32293                                   ;push   cs
32294                                   ;pop    ds
32295
32296 000024F0 50                        push    ax
32297                                   ;mov    ah,CLOSE
32298 000024F1 B43E                      mov     ah,3Eh
32299 000024F3 CD21                      int     21h
32300 000024F5 58                        pop     ax
32301 000024F6 9D                        popf
32302 000024F7 7204                      jc      short conferr           ;if not we've got a problem
32303 000024F9 39C1                      cmp     cx,ax
32304 000024FB 742C                      jz      short getcom            ;couldn't read the file
32305                                   conferr:
32306 000024FD BA[D14A]                  mov     dx,config               ;want to print config error
32307 00002500 E82525                    call    badfil
32308                                   ; 14/04/2024
32309                                   endconv:    ; 01/01/2023
32310 00002503 C3                        retn
32311
32312                                   ;--------------------------------------------------------------------------
32313                                   ;
32314                                   ; entry : multi_pass
32315                                   ;
32316                                   ;             called to execute device=,install= commands
32317                                   ;
32318                                   ;--------------------------------------------------------------------------
32319
32320                                   ; 27/10/2022
32321                                   multi_pass:
32322 00002504 0E                        push    cs
32323 00002505 1F                        pop     ds
32324
32325 00002506 803E[CD02]0A              cmp     byte [multi_pass_id],10
32326                                   ;jae_endconv:
32327 0000250B 73F6                      jae     short endconv           ; do nothing. just return.
32328
32329                                   ; 31/12/2022 - Retro DOS v4.2 (Modified MSDOS 6.21 IO.SYS)
32330 0000250D FF36[A302]                push    word [CONFBOT]
32331                                   ; 27/10/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
32332                                   ;push   word [top_of_cdss]
32333 00002511 07                        pop     es                      ; es -> confbot
32334
32335 00002512 8B36[5803]                mov     si,[org_count]
```

```
32336 00002516 8936[5603]                   mov     [count],si              ; set count
32337 0000251A 31F6                         xor     si,si ; 0
32338 0000251C 8936[5A03]                       mov     [chrptr],si             ; reset chrptr
32339 00002520 8936[AF02]                       mov     [linecount],si          ; reset linecount
32340
32341 00002524 E88822                       call    getchr
32342 00002527 EB06                         jmp     short conflp
32343
32344                                       ; 14/04/2024
32345                                       ; 01/01/2023
32346                                     ;endconv:
32347                                       ;retn
32348
32349                                     getcom:
32350                                       ; 03/01/2023
32351                                       ; ds = cs
32352 00002529 E8C016                           call    organize                ; organize the file
32353 0000252C E88022                       call    getchr
32354                                     conflp:
32355 0000252F 72D2                         jc      short endconv
32356
32357 00002531 FF06[AF02]                       inc     word [linecount]  ; increase linecount
32358
32359                                       ; 08/09/2023
32360 00002535 30E4                         xor     ah,ah ; 0
32361                                       ;mov    byte [multdeviceflag],0      ; reset multdeviceflag.
32362                                       ;mov    byte [setdevmarkflag],0      ; reset setdevmarkflag.
32363 00002537 8826[6619]                   mov     [multdeviceflag],ah ; 0
32364 0000253B 8826[6919]                   mov     [setdevmarkflag],ah ; 0
32365
32366 0000253F 3C0A                         cmp     al,lf                   ; linefeed?
32367 00002541 7448                         je      short blank_line        ;  then ignore this line.
32368
32369                                     ; 31/12/2022 - Retro DOS v4.2 (Modified MSDOS 6.21 IO.SYS)
32370                                     ; (SYSINIT:23CCh)
32371                                     ; 27/10/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
32372                                     ;%if 0
32373
32374                                     ;ifdef      MULTI_CONFIG
32375
32376                                     ;   If this is a genuine CONFIG.SYS command, then there should be a line
32377                                     ;   number immediately following it....
32378
32379 00002543 A2[6419]                       mov     [config_cmd],al         ; save original command code
32380                                       ;and    al,NOT CONFIG_OPTION_QUERY
32381 00002546 247F                         and     al,~CONFIG_OPTION_QUERY ; and al,7Fh
32382
32383                                       ; 08/09/2023
32384 00002548 3826[6519]                   cmp     [config_multi],ah ; 0
32385                                       ;cmp    byte [config_multi],0 ; is this a multi-config config.sys?
32386 0000254C 7427                             je      short not_final         ; no, line number is not embedded
32387
32388 0000254E 50                               push    ax                      ;
32389 0000254F E85D22                           call    getchr                  ; ignore end-of-image errors,
32390 00002552 88C4                             mov     ah,al                   ; because if there's an error
32391 00002554 E85822                           call    getchr                  ; fetching the line number that's
32392 00002557 86E0                             xchg    al,ah                   ; supposed to be there, the next
32393 00002559 A3[AF02]                         mov     [linecount],ax          ; getchr call will get the same error
32394 0000255C 58                               pop     ax
32395                                     ;
32396                                     ;   HACK: when 4DOS.COM is the shell and it doesn't have an environment from
32397                                     ;   which to obtain its original program name, it grovels through all of
32398                                     ;   memory to find the filename that was used to exec it; it wants to find
32399                                     ;   the SHELL= line in the in-memory copy of CONFIG.SYS, and it knows that
32400                                     ;   sysinit converts the SHELL= keyword to an 'S', so it expects to find an 'S'
32401                                     ;   immediately before the filename, but since we are now storing line # info
32402                                     ;   in the config.sys memory image, 4DOS fails to find the 'S' in the right
32403                                     ;   spot.
32404                                     ;
32405                                     ;   So, on the final pass of CONFIG.SYS, copy the command code (eg, 'S')
32406                                     ;   over the line number info, since we no longer need that info anyway. This
32407                                     ;   relies on the fact that getchr leaves ES:SI pointing to the last byte
32408                                     ;   retrieved.
32409                                     ;
32410 0000255D 803E[CD02]02                     cmp byte [multi_pass_id],2; final pass?
32411 00002562 7211                             jb  short not_final             ; no
32412                                           ;test      word [install_flag],have_install_cmd
32413 00002564 F606[CE02]01                 test    byte [install_flag],have_install_cmd ; 1
32414 00002569 7407                             jz  short final         ; no install cmds, so yes it is
32415 0000256B 803E[CD02]03                     cmp byte [multi_pass_id],3; final pass?
32416 00002570 7203                             jb  short not_final             ; no
32417                                     final:                                  ;
32418 00002572 268804                       mov     [es:si],al              ; save backward-compatible command code
32419                                     not_final:                              ;
32420                                     ;endif
32421
32422                                     ; 31/12/2022
32423                                     ;%endif ; 27/10/2022
32424
32425 00002575 88C4                         mov     ah,al
32426 00002577 E83522                       call    getchr
32427 0000257A 7314                         jnc     short tryi
32428
32429 0000257C 803E[CD02]02                 cmp     byte [multi_pass_id],2
32430                                       ;jae    short jae_endconv     ; do not show badop again for multi_pass.
32431                                       ; 27/10/2022
32432 00002581 7380                         jnb     short endconv
32433 00002583 E90009                       jmp     badop
32434
32435                                     coff:
32436                                       ; 11/12/2022
32437                                       ; ds = cs
32438                                       ;push   cs
32439                                       ;pop    ds
32440 00002586 E81D22                       call    newline
32441 00002589 EBA4                         jmp     short conflp   ; 13/05/2019
32442
32443                                     blank_line:
32444 0000258B E82122                       call    getchr
32445 0000258E EB9F                         jmp     short conflp
32446
32447                                       ; 27/10/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
32448                                     ; 11/12/2022
32449                                     ; (there is not a jump or call to here from anywhere!)
32450                                     ;coff_p:
32451                                       ;push   cs
32452                                       ;pop    ds
32453
32454                                     ;to handle install= commands,we are going to use multi-pass.
32455                                     ;the first pass handles the other commands and only set install_flag when
32456                                     ;it finds any install command. the second pass will only handle the
32457                                     ;install= command.
32458
32459                                     ;--------------------------------------------------------------------------------
```

```
                                ;install command
                                ;--------------------------------------------------------------------------------
                                
                                    ; 27/10/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 IO.SYS)
                                    ; (SYSINIT:2250h)
                                tryi:
00002590 803E[CD02]00               cmp     byte [multi_pass_id],0; the initial pass for DOS=HI
00002595 7503                       jne     short not_init_pass
00002597 E97F01                     jmp     multi_try_doshi
                                not_init_pass:
0000259A 803E[CD02]02               cmp     byte [multi_pass_id],2; the second pass was for ifs=
                                        ; 11/12/2022
                                    ;je     short multi_pass_coff2; now it is NOPs
0000259F 74E5                       je      short coff
                                    ; 27/10/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
                                    ;je     short multi_pass_coff
                                                                    ; This pass can be made use of if
                                                                    ; we want do some config.sys process
                                                                    ; after device drivers are loaded
                                                                    ; and before install= commands
                                                                    ; are processed
                                
000025A1 803E[CD02]03               cmp     byte [multi_pass_id],3; the third pass for install= ?
000025A6 741D                       je      short multi_try_i
000025A8 80FC48                         cmp     ah, CONFIG_DOS  ; 'H'
                                    ; 11/12/2022
                                    ;je     short multi_pass_coff2
000025AB 74D9                       je      short coff
                                    ; 27/10/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
                                    ;je     short multi_pass_coff
                                
                                ;       make note of any INSTALL= or INSTALLHIGH= commands we find,
                                ;       but don't process them now.
                                
000025AD 80FC49                         cmp     ah,CONFIG_INSTALL ; 'I'  ; install= command?
                                    ; 31/12/2022 - Retro DOS v4.2 (Modified MSDOS 6.21 IO.SYS)
000025B0 7507                       jne     short precheck_installhigh ; the first pass is for normal operation.
                                    ; 27/10/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
                                    ;jne    short tryb
                                
                                    ;or     word [install_flag],have_install_cmd ; set the flag
000025B2 800E[CE02]01               or      byte [install_flag],have_install_cmd ; 1
                                multi_pass_coff2:
000025B7 EBCD                       jmp     short coff ; 13/05/2019      ; and handles the next command
                                
                                ; 31/12/2022 - Retro DOS v4.2 (Modified MSDOS 6.21 IO.SYS)
                                ; (SYSINIT:2448h)
                                ; 27/10/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
                                ;%if 0
                                precheck_installhigh:
000025B9 80FC57                     cmp     ah,CONFIG_INSTALLHIGH ; 'W' ; signifier for INSTALLHIGH
000025BC 756B                       jne     short tryb          ; carry on with normal processing
                                    ;or     word [install_flag],have_install_cmd
000025BE 800E[CE02]01               or      byte [install_flag],have_install_cmd ; 1
000025C3 EBC1                       jmp     short coff
                                ;%endif ; 27/10/2022
                                
                                multi_try_i:
000025C5 80FC49                         cmp     ah,CONFIG_INSTALL ; 'I' ; install= command?
                                    ; 31/12/2022 - Retro DOS v4.2
000025C8 750A                       jne     short multi_try_n     ; no, check for installhigh
                                    ; 27/10/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
                                    ;jne    short multi_pass_filter
                                
                                ; 31/12/2022
                                ;%if 1
                                ; 27/10/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
                                ;%if 0
                                ;ifdef      MULTI_CONFIG
000025CA E84F20                     call    query_user              ; query the user if config_cmd
000025CD 7241                       jc      short multi_pass_filter       ; has the CONFIG_OPTION_QUERY bit set
                                ;endif
                                ;%endif ; 27/10/2022
                                
000025CF E8C7EC                     call    do_install_exec       ;install it.
000025D2 EBB2                       jmp     short coff            ;to handle next install= command.
                                
                                ; 31/12/2022 - Retro DOS v4.2 (Modified MSDOS 6.21 IO.SYS)
                                ; (SYSINIT:2463h)
                                ;%if 1
                                ; 27/10/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
                                ;%if 0
                                
                                multi_try_n:
000025D4 80FC57                         cmp     ah,CONFIG_INSTALLHIGH   ; installhigh= command?
000025D7 7537                           jne short multi_pass_filter       ; no. ignore this.
                                ;ifdef      MULTI_CONFIG
000025D9 E84020                         call    query_user              ; query the user if config_cmd
000025DC 7232                           jc      short multi_pass_filter  ; has the CONFIG_OPTION_QUERY bit set
                                ;endif
                                
                                ;       The memory environment is in its normal DOS state, so do
                                ;       the standard calls to set the alloc strategy for loading high
                                
000025DE B80058                     mov     ax,(ALLOCOPER<<8)|0 ; 5800h
000025E1 CD21                       int     21h                     ;get alloc strategy
000025E3 89C3                       mov     bx,ax
000025E5 53                             push    bx                      ; save for the return
                                
000025E6 81CB8000                       or  bx,HIGH_FIRST  ; 80h  ;set alloc to HighFirst
000025EA B80158                     mov     ax,(ALLOCOPER<<8)|1 ; 5801h
000025ED CD21                       int     21h                     ;set alloc strategy
                                
000025EF B80258                     mov      ax,(ALLOCOPER<<8)|2 ; 5802h
000025F2 CD21                           int     21h                     ; get link state
000025F4 30E4                           xor     ah,ah                   ; clear top byte
000025F6 50                             push    ax                      ; save for return
                                
000025F7 B80358                         mov ax,(ALLOCOPER<<8)|3 ; 5803h
000025FA BB0100                     mov     bx,1
000025FD CD21                       int     21h                     ;link in UMBs
                                
000025FF E897EC                     call    do_install_exec       ;install it.
                                
00002602 B80358                         mov     ax,(ALLOCOPER<<8)|3
00002605 5B                             pop     bx                      ; recover original link state
00002606 CD21                           int     21h
00002608 5B                             pop     bx                      ; recover original alloc strategy
00002609 B80158                         mov     ax,(ALLOCOPER<<8)|1
0000260C CD21                           int     21h
                                
                                    ;jmp    short coff            ;to handle next install= command.
                                    ; 01/01/2023
0000260E EBA7                       jmp     short multi_pass_coff2
```

```
                                ;%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
```

```
32708                                   ;   jne     short if18
32709
32710 0000266A E82508                       call    badparm_p
32711 0000266D C706[1322]0000               mov     word [p_h_buffers],0
32712 00002673 EB12                         jmp     short sr7
32713                                   if18:
32714 00002675 A1[1122]                     mov     ax,[p_buffers] ; we don't have any problem.
32715 00002678 A3[9902]                     mov     [buffers],ax   ; now,let's set it really.
32716
32717 0000267B A1[1322]                     mov     ax,[p_h_buffers]
32718 0000267E A3[9B02]                     mov     [h_buffers],ax
32719
32720                                   ;   mov     al,[p_buffer_slash_x] ; M016
32721                                   ;   mov     [buffer_slash_x],al
32722
32723 00002681 A1[AF02]                     mov     ax,[linecount]
32724 00002684 A3[B902]                     mov     [buffer_linenum],ax ; save the line number for the future use.
32725                                       ; 31/12/2022
32726                                       ;jmp    short sr7
32727                                       ; 03/01/2023
32728                                   sr7:
32729 00002687 E9FCFE                       jmp     coff
32730
32731                                   ;---------------------------------------------------------------------------
32732                                   ; break command
32733                                   ;---------------------------------------------------------------------------
32734
32735                                   ;**************************************************************************
32736                                   ;                                                                         *
32737                                   ; function: parse the parameters of break = command.                      *
32738                                   ;                                                                         *
32739                                   ; input :                                                                 *
32740                                   ;   es:si -> parameters in command line.                                 *
32741                                   ; output:                                                                 *
32742                                   ;   turn the control-c check on or off.                                  *
32743                                   ;                                                                         *
32744                                   ; subroutines to be called:                                               *
32745                                   ;   sysinit_parse                                                        *
32746                                   ; logic:                                                                  *
32747                                   ; {                                                                       *
32748                                   ;   set di to brk_parms;                                                 *
32749                                   ;   set dx,cx to 0;                                                      *
32750                                   ;   while (end of command line)                                         *
32751                                   ;   { sysinit_parse;                                                    *
32752                                   ;     if (no error) then                                                  *
32753                                   ;          if (result_val._$P_item_tag == 1) then        /*on          */ *
32754                                   ;             set p_ctrl_break,on;                                       *
32755                                   ;          else                                  /*off         */ *
32756                                   ;             set p_ctrl_break,off;                                      *
32757                                   ;     else {show message;error_exit};                                     *
32758                                   ;   };                                                                    *
32759                                   ;   if (no error) then                                                   *
32760                                   ;      dos function call to set ctrl_break check according to           *
32761                                   ; };                                                                      *
32762                                   ;                                                                         *
32763                                   ;**************************************************************************
32764
32765                                       ; 27/10/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 IO.SYS)
32766                                       ; (SYSINIT:22FFh)
32767                                   tryc:
32768 0000268A 80FC43                         cmp     ah,CONFIG_BREAK ; 'C'
32769 0000268D 7539                         jne     short trym
32770
32771                                   ; 31/12/2022 - Retro DOS v4.2 (Modified MSDOS 6.21 IO.SYS)
32772                                   ; (SYSINIT:2527h)
32773                                   ; 27/10/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
32774                                   ;%if 0
32775                                   ;ifdef      MULTI_CONFIG
32776 0000268F E88A1F                       call    query_user              ; query the user if config_cmd
32777 00002692 7234                         jc      short trym              ; has the CONFIG_OPTION_QUERY bit set
32778                                   ;endif
32779                                   ;%endif ; 27/10/2022
32780
32781 00002694 BF[1F22]                     mov     di,brk_parms
32782 00002697 31C9                         xor     cx,cx
32783                                       ; 03/01/2023
32784                                       ;mov    dx,cx
32785                                   do22:
32786 00002699 E8CB07                       call    sysinit_parse
32787 0000269C 7303                         jnc     short if22              ; parse error
32788                                       ;call   badparm_p               ;  show message and end the search loop.
32789                                       ;;jmp   short sr22
32790                                       ; 31/12/2022
32791                                   ;sr22:
32792                                       ;jmp    coff
32793                                       ; 03/01/2023
32794 0000269E E9B506                       jmp     badparm_p_coff
32795                                   if22:
32796 000026A1 83F8FF                       cmp     ax,_$P_RC_EOL           ; end of line?
32797 000026A4 7415                         je      short en22              ; then end the $endloop
32798
32799                                       ;cmp    byte [result_val_itag],1
32800 000026A6 803E[1822]01                 cmp     byte [result_val+_$P_Result_Blk.Item_Tag],1
32801 000026AB 7507                         jne     short if26
32802
32803 000026AD C606[4422]01                 mov     byte [p_ctrl_break],1 ; turn it on
32804                                       ;jmp    short en26
32805                                       ; 31/12/2022
32806 000026B2 EBE5                         jmp     short do22
32807                                   if26:
32808 000026B4 C606[4422]00                 mov     byte [p_ctrl_break],0 ; turn it off
32809                                   en26:
32810 000026B9 EBDE                         jmp     short do22              ; we actually set the ctrl break
32811                                   en22:
32812 000026BB B433                         mov     ah,SET_CTRL_C_TRAPPING ; if we don't have any parse error.
32813 000026BD B001                         mov     al,1
32814 000026BF 8A16[4422]                   mov     dl,[p_ctrl_break]
32815 000026C3 CD21                         int     21h
32816                                       ; 31/12/2022
32817                                       ;jmp    short sr22
32818                                       ; 03/01/2023
32819                                   sr22:
32820 000026C5 E9BEFE                       jmp     coff
32821
32822                                   ;---------------------------------------------------------------------------
32823                                   ; multitrack command
32824                                   ;---------------------------------------------------------------------------
32825
32826                                   ;**************************************************************************
32827                                   ;                                                                         *
32828                                   ; function: parse the parameters of multitrack= command.                  *
32829                                   ;                                                                         *
32830                                   ; input :                                                                 *
32831                                   ;   es:si -> parameters in command line.                                 *
```

```
                                ; output:                                                       *
                                ;   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
00002780 A2[6624]                   mov     [DevSavedDelim],al
                                    ;mov    [cs:DevSavedDelim],al ; Save the delimiter before replacing
                                                                  ;  it with null
                                    ; 18/12/2022
00002783 29DB                       sub     bx,bx
00002785 26881C                     mov     [es:si],bl ; 0
                                    ;mov    byte [es:si],0

00002788 07                         pop     es
00002789 5E                         pop     si      ; 14/04/2024

                                ;--------------------------------------------------------------------------
                                ; BEGIN PATCH TO CHECK FOR NON-EXISTANT UMBs   -- t-richj 7-21-92
                                ;--------------------------------------------------------------------------

                                ; 31/12/2022 - Retro DOS v4.2 (Modified MSDOS 6.21 IO.SYS)
                                ; (SYSINIT:2642h)
                                ; 28/10/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
                                ;%if 0
                                ; 10/07/2023 - Retro DOS v4.2 IO.SYS (optimized)
                                ; MSDOS 6.21 IO.SYS - SYSINIT:2642h
                                %if 1
                                    ; 01/01/2023
                                    ; ds = cs
0000278A E8F00C                     call    UmbTest                 ; See if UMBs are around...
                                    ; 01/01/2023
                                    ;jnc    short NrmTst            ; ...yep. So do that normal thang.

                                    ;mov    byte [cs:DeviceHi],0    ; ...nope... so load low.
                                    ; 31/12/2022
                                    ; ds = cs, bx = 0
                                    ;mov    byte [DeviceHi],bl ; 0
                                    ;jmp    short LoadDevice
                                    ; 01/01/2023
0000278D 7222                       jc      short LoadDevice ; bl = 0
                                %endif
                                ;%endif
                                ;--------------------------------------------------------------------------
                                ; END PATCH TO CHECK FOR NON-EXISTANT UMBs   -- t-richj 7-21-92
                                ;--------------------------------------------------------------------------

                                NrmTst:
                                    ; 11/12/2022
                                    ; ds = cs
                                    ;;mov   byte [cs:DeviceHi],0
                                    ;mov    byte [DeviceHi],0
                                    ; 18/12/2022
                                    ; bx = 0
0000278F 381E[4224]                 cmp     [DevUMB],bl ; 0
                                    ;cmp    byte [DevUMB],0
                                    ;;cmp   byte [cs:DevUMB],0    ; do we support UMBs
00002793 741C                       je      short LoadDevice      ; no, we don't
                                    ;mov    byte [cs:DeviceHi],1
                                    ; 11/12/2022
                                    ;mov    byte [DeviceHi],1
                                    ; 18/12/2022
00002795 FEC3                       inc     bl ; mov bl,1 ; (*)
                                    ; 11/12/2022
                                    ;jmp    short LoadDevice2     ; 11/12/2022
00002797 EB18                       jmp     short LoadDevice

                                ;--------------------------------------------------------------------------
                                ; device command
                                ;--------------------------------------------------------------------------

                                    ; 31/12/2022 - Retro DOS v4.2 (Modified MSDOS 6.21 IO.SYS)
                                    ; (SYSINIT:2665h)

                                    ; 28/10/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 IO.SYS)
                                    ; (SYSINIT:2401h)
                                tryd:
                                    ; 11/12/2022
                                    ;xor    bx,bx ; 31/12/2022
                                    ;
00002799 80FC44                        cmp     ah,CONFIG_DEVICE ; 'D'
0000279C 7403                       je      short gotd
                                skip_it2:
0000279E E9FC02                     jmp     tryq
                                gotd:

                                ; 31/12/2022 - Retro DOS v4.2
                                ; 28/10/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
                                ;%if 0
                                ;ifdef      MULTI_CONFIG
000027A1 E8781E                     call    query_user              ; query the user if config_cmd
000027A4 72F8                       jc      short skip_it2          ; has the CONFIG_OPTION_QUERY bit set
                                ;endif
                                ;%endif ; 28/10/2022

                                    ; 31/12/2022
000027A6 29DB                       sub     bx,bx
                                    ; bx = 0
                                    ; 11/12/2022
                                    ; ds = cs
                                    ;mov    byte [DeviceHi],0
                                    ;mov    word [DevSizeOption],0
000027A8 891E[5224]                 mov     [DevSizeOption],bx ; 0
000027AC C606[6624]20               mov     byte [DevSavedDelim],' '
                                    ;mov    byte [cs:DeviceHi],0  ; not to be loaded in UMB ;M007
                                    ;mov    word [cs:DevSizeOption],0
                                    ;mov    byte [cs:DevSavedDelim],' ' ; In case of DEVICE= the null has to
                                                                        ;  be replaced with a ' '
                                LoadDevice:                                 ; device= or devicehigh= command.
                                    ; 11/12/2022
                                    ;mov    byte [DeviceHi],0
000027B1 881E[5124]                 mov     byte [DeviceHi],bl    ; 0 or 1 (*)
                                LoadDevice2:
                                    ; 28/10/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
                                    ;
                                    ;push    cs
                                        ;pop     ds
                                    ;
                                    ;mov    [bpb_addr],si         ; pass the command line to the dvice
                                    ;mov    [bpb_addr+2],es
                                    ;
                                    ;mov    [DevCmdLine],si               ; save it for ourself
                                    ;mov    [DevCmdLine+2],es
                                    ;
                                    ;mov    byte [driver_units],0 ; clear total block units for driver

                                    ; 11/12/2022
                                    ; ds = cs
```

```
                                    ;mov    bx,cs
                                    ;mov    ds,bx
                                    ;mov    [cs:bpb_addr],si      ; pass the command line to the dvice
000027B5 8936[8103]                 mov     [bpb_addr],si
                                    ;mov    [cs:bpb_addr+2],es
000027B9 8C06[8303]                 mov     [bpb_addr+2],es
                                    ;mov    [cs:DevCmdLine],si    ; save it for ourself
000027BD 8936[6224]                 mov     [DevCmdLine],si
                                    ;mov    [cs:DevCmdLine+2],es
000027C1 8C06[6424]                 mov     [DevCmdLine+2],es
                                    ; 31/12/2022 - Retro DOS v4.2
000027C5 C606[6A19]00               mov     byte [driver_units],0 ; clear total block units for driver
000027CA E82520                     call    round
000027CD E8910E                     call    SizeDevice
000027D0 723F                       jc      short BadFile
                                    ; 11/12/2022
                                    ; ds = cs
                                ; - Begin DeviceHigh primary logic changes ---------------------------------
                                ; 31/12/2022 - Retro DOS v4.2 (Modified MSDOS 6.21 IO.SYS)
                                ; (SYSINIT:26A4h)
                                ; 28/10/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
                                ;%if 0
000027D2 C606[4124]01               mov     byte [ConvLoad],1     ; Doesn't matter if DeviceHi==0
                                    ; 22/07/2023
                                    ;mov    al,[DeviceHi]         ; If not using upper memory,
000027D7 800E[5124]00               or      byte [DeviceHi],0     ; Skip all this and go on to
                                    ; 10/07/2023
                                    ;or     al,al
000027DC 741E                       jz      short DevConvLoad     ; the actual load.
                                    ;call   GetLoadUMB            ; Returns first UMB spec'ed in AX
000027DE A0[FF23]                   mov     al,[UmbLoad]   ; 19/04/2019 - Retro DOS v4.0
000027E1 3CFF                       cmp     al,-1                 ; If umb0 not specified, it's old style
000027E3 7417                       jz      short DevConvLoad     ; so load high even if SIZE= is smaller
000027E5 FE0E[4124]                 dec     byte [ConvLoad] ; 0   ; They specified /L, so use new loader
000027E9 E8590A                     call    GetLoadSize           ; Returns size of first UMB specified
000027EC 09C0                       or      ax,ax
000027EE 7406                       jz      short tryd_1          ; If size1 not specified, nada to do:
000027F0 3B06[3324]                 cmp     ax,[DevSize]          ; /L:...,Size < DevSize?
000027F4 7D06                       jge     short DevConvLoad
                                tryd_1:
000027F6 A1[3324]                   mov     ax,[DevSize]          ; Size < DevSize, so write DevSize as
000027F9 E8550A                     call    StoLoadSize           ; minsize for load UMB.
                                ;%endif ; 28/10/2022
                                ; - End DeviceHigh primary logic changes -----------------------------------
                                DevConvLoad:
                                    ; 28/10/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
000027FC E8660D                     call    InitDevLoad
                                    ; 11/12/2022
                                    ; ds = cs
000027FF A1[3524]                   mov     ax,[DevLoadAddr]
00002802 0306[3324]                 add     ax,[DevSize]
00002806 7206                       jc      short NoMem
00002808 3906[3724]                 cmp     [DevLoadEnd],ax
0000280C 7315                       jae     short LoadDev
                                    ; 11/12/2022
                                    ;mov    ax,[cs:DevLoadAddr]
                                    ;add    ax,[cs:DevSize]
                                    ;jc     short NoMem
                                    ;cmp    [cs:DevLoadEnd],ax
                                    ;jae    short LoadDev
                                NoMem:
                                    ; 11/12/2022
                                    ; ds = cs
                                    ;jmp    mem_err
0000280E E92020                     jmp     mem_err2
                                BadFile:
                                    ; 28/10/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
                                    ;;call  RetFromUM              ; Does nothing if didn't call HideUMBs
                                    ;;cmp   byte [es:si],' '
                                       ;;jae      short tryd_2
                                    ; 31/12/2022
                                    ;cmp    byte [es:si],0Dh      ; cr
                                       ;jne       short tryd_2
                                    ;jmp    badop
                                    ; 31/12/2022
                                    ; ds = cs
                                    ; 31/12/2022 - Retro DOS v4.2 (Modified MSDOS 6.21 IO.SYS)
                                    ; (SYSINIT:26E6h)
00002811 E8AE0E                     call    RetFromUM             ; Does nothing if didn't call HideUMBs
00002814 26803C20                   cmp     byte [es:si],' '
                                    ;cmp    byte [es:si],20h ; space
00002818 7303                       jnb     short tryd_2
0000281A E96906                     jmp     badop
                                tryd_2:
0000281D E80C22                     call    badload
00002820 E963FD                     jmp     coff
                                LoadDev:
00002823 06                         push    es
00002824 1F                         pop     ds
00002825 89F2                       mov     dx,si                 ;ds:dx points to file name
00002827 E87C0E                     call    ExecDev               ; load device driver using exec call
                                badldreset:
0000282A 1E                         push    ds
0000282B 07                         pop     es                    ;es:si back to config.sys
0000282C 0E                         push    cs
0000282D 1F                         pop     ds                    ;ds back to sysinit
0000282E 72E1                       jc      short BadFile
                                goodld:
                                    ; 11/12/2022
                                    ; ds = cs
```

```
33328 00002830 06                       push    es ; +  ; 31/12/2022
33329 00002831 56                       push    si ; ++
33330 00002832 E89E0E                   call    RemoveNull
33331 00002835 06                       push    es
33332 00002836 56                       push    si
33333
33334 00002837 0E                       push    cs
33335 00002838 07                       pop     es
33336
33337 00002839 1E                       push    ds ; **  ; ds = cs
33338 0000283A 56                       push    si
33339
33340                                   ;lds    si,[cs:DevEntry]      ; peeks the header attribute
33341                                   ; 31/12/2022
33342                                   ; ds = cs
33343 0000283B C536[3924]               lds     si,[DevEntry]
33344
33345                                   ;test   word [si+4],8000h
33346                                   ; 11/12/2022
33347 0000283F F6440580                 test    byte [si+SYSDEV.ATT+1],DEVTYP>>8
33348                                   ;test   word [si+SYSDEV.ATT],DEVTYP ; block device driver?
33349 00002843 7514                     jnz     short got_device_com_cont   ; no.
33350
33351 00002845 2EC536[6D02]             lds     si,[cs:DOSINFO]               ; ds:si -> sys_var
33352                                   ;cmp    byte [si+32],26
33353 0000284A 807C201A                 cmp     byte [si+SYSI_NUMIO],26       ; no more than 26 drive number
33354 0000284E 7209                     jb      short got_device_com_cont
33355
33356 00002850 5E                       pop     si
33357 00002851 1F                       pop     ds ; **
33358
33359 00002852 5E                       pop     si                    ; clear the stack
33360 00002853 07                       pop     es
33361
33362                                   ; 28/10/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
33363                                   ;call   RetFromUM
33364                                   ; 31/12/2022
33365                                   ; ds = cs ; **
33366 00002854 E86B0E                   call    RetFromUM             ; Do this before we leave
33367
33368                                   ;jmp    short badnumblock
33369                                   ; 31/12/2022
33370 00002857 EB73                     jmp     short badnumblock2  ; ds = cs
33371
33372                                 got_device_com_cont:
33373 00002859 5E                       pop     si
33374 0000285A 1F                       pop     ds
33375
33376                                   ; 11/12/2022
33377                                   ; ds = cs
33378
33379 0000285B E8AE06                   call    LieInt12Mem
33380 0000285E E80B07                   call    UpdatePDB             ; update the PSP:2 value M020
33381
33382                                   ; 11/12/2022
33383                                   ; ds = cs
33384                                   ; 08/09/2023
33385 00002861 31C0                     xor     ax, ax ; 0
33386 00002863 3806[6619]               cmp     byte [multdeviceflag],al ; 0
33387                                   ;cmp    byte [multdeviceflag],0
33388                                   ;cmp    byte [cs:multdeviceflag],0 ; Pass limit only for the 1st device
33389                                                                      ;  driver in the file ; M027
33390 00002867 750B                     jne     short skip_pass_limit ;                     ; M027
33391
33392                                   ; 11/12/2022
33393                                   ; ds = cs
33394                                   ;mov    word [cs:break_addr],0; pass the limit to the DD
33395                                   ;mov    bx,[cs:DevLoadEnd]
33396                                   ;mov    [cs:break_addr+2],bx
33397
33398                                   ;mov    word [break_addr],0
33399                                   ; 08/09/2023
33400 00002869 A3[7D03]                 mov     [break_addr],ax ; 0
33401 0000286C 8B1E[3724]               mov     bx,[DevLoadEnd]
33402 00002870 891E[7F03]               mov     [break_addr+2],bx
33403
33404                                 skip_pass_limit:
33405                                 ;   Note: sysi_numio (in DOS DATA) currently reflects the REAL
33406                                 ;   number of installed devices (including DblSpace drives) where
33407                                 ;   "drivenumber" is the number that the next block device will
33408                                 ;   be assigned to. Because some naughty device drivers (like
33409                                 ;   interlnk) look at the internal DOS variable instead of the
33410                                 ;   value we pass it, we'll temporarily stick our value into
33411                                 ;   DOS DATA while we're initializing the device drivers.
33412                                 ;
33413                                 ;   Note that this will make it impossible for this device
33414                                 ;   driver to access the DblSpace drive letters, whether
33415                                 ;   they are swapped-hosts or unswapped compressed drives,
33416                                 ;   during its initialization phase.
33417
33418                                 ; 31/12/2022 - Retro DOS v4.2 (Modified MSDOS 6.21 IO.SYS)
33419                                 ; (SYSINIT:2752h)
33420                                 ; 29/10/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
33421                                 ;%if 0
33422                                     ; 31/12/2022
33423                                     ;push   ds
33424
33425                                     ;lds    bx,[cs:DOSINFO]               ; ds:bx -> sys_var
33426                                     ; 31/12/2022
33427                                     ; ds = cs
33428                                     ; 08/09/2023
33429                                     ;lds    bx,[DOSINFO]          ; ds:bx -> sys_var
33430
33431                                     ;mov    al,[cs:drivenumber]   ; temporarily use this next drv value
33432                                     ;mov    [cs:devdrivenum],al   ; pass drive number in packet to driver
33433                                     ;mov    ah,al
33434
33435                                     ; 08/09/2023
33436                                     ; ds = cs
33437 00002874 A0[8503]                 mov     al,[drivenumber]      ; temporarily use this next drv value
33438 00002877 A2[8503]                 mov     [devdrivenum],al      ; pass drive number in packet to driver
33439 0000287A 88C4                     mov     ah,al
33440 0000287C C51E[6D02]               lds     bx,[DOSINFO]          ; ds:bx -> sys_var
33441
33442 00002880 874720                   xchg    ax,[bx+SYSI_NUMIO]    ; swap with existing values
33443                                   ; 31/12/2022
33444                                   ;pop    ds
33445
33446 00002883 50                       push    ax                    ; save real sysi_numio/ncds in ax
33447
33448                                 ;%endif ; 29/10/2022
33449
33450                                     ; 29/10/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 IO.SYS)
33451                                     ; (SYSINIT:24B9h)
```

```
33452
33453 00002884 BB0600                      mov     bx,SYSDEV.STRAT ; 6
33454 00002887 E8B01F                      call    calldev                 ; calldev (sdevstrat);
33455 0000288A BB0800                      mov     bx,SYSDEV.INT ; 8
33456 0000288D E8AA1F                      call    calldev                 ; calldev (sdevint);
33457
33458                                      ; 11/12/2022
33459                                      ; ds <> cs (from calldev) ; 31/12/2022
33460
33461                                  ; 31/12/2022 - Retro DOS v4.2 (Modified MSDOS 6.21 IO.SYS)
33462                                  ; (SYSINIT:2773h)
33463                                  ; 29/10/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
33464                                  ;%if 0
33465 00002890 58                          pop     ax                      ; get real sysi_numio value
33466                                      ; 31/12/2022
33467                                      ;push  ds
33468 00002891 2EC51E[6D02]                lds     bx,[cs:DOSINFO]                 ; ds:bx -> sys_var
33469 00002896 894720                      mov     [bx+SYSI_NUMIO],ax      ; swap with existing values
33470                                      ; 31/12/2022
33471                                      ;pop   ds
33472
33473                                  ;%endif ; 29/10/2022
33474
33475                                      ; 11/12/2022
33476 00002899 0E                          push    cs
33477 0000289A 1F                          pop     ds
33478
33479 0000289B E89C06                      call    TrueInt12Mem
33480
33481                                      ; 11/12/2022
33482                                      ; ds = cs
33483                                      ;mov   ax,[cs:break_addr]      ; move break addr from the req packet
33484                                      ;mov   [cs:DevBrkAddr],ax
33485                                      ;mov   ax,[cs:break_addr+2]
33486                                      ;mov   [cs:DevBrkAddr+2],ax
33487 0000289E A1[7D03]                    mov     ax,[break_addr]
33488 000028A1 A3[3D24]                    mov     [DevBrkAddr],ax
33489 000028A4 A1[7F03]                    mov     ax,[break_addr+2]
33490 000028A7 A3[3F24]                    mov     [DevBrkAddr+2],ax
33491
33492                                      ; 29/10/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
33493                                      ;call  RetFromUM               ; There we go... all done.
33494                                      ; 31/12/2022 - Retro DOS v4.2 (Modified MSDOS 6.21 IO.SYS)
33495                                      ; (SYSINIT:2791h)
33496 000028AA E8150E                      call    RetFromUM               ; There we go... all done.
33497
33498                                      ; 31/12/2022
33499                                      ; ds = cs
33500
33501                                      ; 11/12/2022
33502 000028AD 803E[4224]00                cmp     byte [DevUMB],0
33503                                      ;cmp   byte [cs:DevUMB],0
33504 000028B2 7403                        je      short tryd_3
33505 000028B4 E80010                      call    AllocUMB
33506                                      ; 31/12/2022
33507                                      ; ds = cs
33508                                  tryd_3:
33509
33510                                  ;ifndef ROMDOS
33511                                  ;------ If we are waiting to be moved into hma lets try it now !!!
33512
33513                                      ; 11/12/2022
33514                                      ; ds = cs
33515
33516                                      ;cmp   byte [cs:runhigh],0FFh
33517 000028B7 803E[6C02]FF                cmp     byte [runhigh],0FFh ; 11/12/2022
33518 000028BC 7503                        jne     short tryd_4
33519
33520                                      ; 11/12/2022
33521                                      ; ds = cs
33522 000028BE E8D7E1                      call    TryToMovDOSHi           ; move DOS into HMA if reqd
33523                                  tryd_4:
33524                                  ;endif ; ROMDOS
33525
33526 000028C1 5E                           pop     si
33527 000028C2 1F                          pop     ds
33528 000028C3 C60400                      mov     byte [si],0             ; *p = 0;
33529
33530 000028C6 0E                           push    cs
33531 000028C7 1F                          pop     ds
33532
33533 000028C8 EB1F                        jmp     short was_device_com
33534
33535                                  ;--------------------------------------------------------------------------
33536
33537                                  ; 02/04/2019 - Retro DOS v4.0
33538
33539                                  ; 31/12/2022 - Retro DOS v4.2 (Modified MSDOS 6.21 IO.SYS)
33540                                  ; (SYSINIT:27B3h)
33541
33542                                  badnumblock:
33543 000028CA 0E                          push    cs
33544 000028CB 1F                          pop     ds
33545                                  badnumblock2:       ; 31/12/2022 (ds=cs)
33546 000028CC BA[7051]                    mov     dx,badblock
33547 000028CF E88221                      call    print
33548
33549                                  ;------ fall thru ---------------------------------------------------------
33550
33551                                      ; 31/12/2022 - Retro DOS v4.2
33552
33553                                  erase_dev_do:                           ; modified to show message "error in config.sys..."
33554
33555                                      ;call  CheckDoubleSpace ; MSDOS 6.21 IO.SYS SYSINIT:27BBh
33556                                                              ; (Note: 'call CheckDoubleSpace'
33557                                                              ; has been removed at 'erase_dev_do:' pos
33558                                                              ; in PCDOS 7.1 IBMBIO.COM - SYSINIT:2CBAh)
33559                                                              ; Erdogan Tan - 10/07/2023
33560 000028D2 5E                          pop     si ; ++
33561 000028D3 07                          pop     es ; + ; 31/12/2022
33562
33563 000028D4 0E                          push    cs
33564 000028D5 1F                          pop     ds
33565
33566                                  skip1_resetmemhi:
33567                                      ; 11/12/2022
33568                                      ; ds = cs
33569 000028D6 833E[8603]00                cmp     word [configmsgflag],0
33570                                      ;cmp   word [cs:configmsgflag],0
33571 000028DB 7409                        je      short no_error_line_msg
33572
33573 000028DD E8DA05                      call    error_line              ; no "error in config.sys" msg for device driver. dcr d493
33574                                      ; 11/12/2022
33575                                      ; ds = cs
```

```
33576                                   ;mov    word [cs:configmsgflag],0
33577 000028E0 C706[8603]0000           mov     word [configmsgflag],0; set the default value again.
33578
33579                                   no_error_line_msg:
33580 000028E6 E99DFC                   jmp     coff
33581
33582                                   ;--------------------------------------------------------------------------
33583
33584                                   was_device_com:
33585                                       ; 14/12/2022
33586                                       ; ds = cs
33587 000028E9 A1[3F24]                 mov     ax,[DevBrkAddr+2]
33588                                   ;mov    ax,[cs:DevBrkAddr+2] ; 13/05/2019
33589 000028EC 3B06[3724]               cmp     ax,[DevLoadEnd]
33590                                   ;cmp    ax,[cs:DevLoadEnd]
33591 000028F0 7605                     jbe     short breakok
33592
33593 000028F2 5E                       pop     si
33594 000028F3 07                       pop     es
33595 000028F4 E91AFF                   jmp     BadFile
33596
33597                                   breakok:
33598                                       ; 14/12/2022
33599                                       ; ds = cs
33600 000028F7 C43E[6D02]               les     di,[DOSINFO]
33601 000028FB C516[3924]               lds     dx,[DevEntry]
33602                                   ;lds    dx,[cs:DevEntry]      ;set ds:dx to header
33603 000028FF 89D6                     mov     si,dx
33604
33605                                       ; 14/11/2022
33606                                   ;les    di,[cs:DOSINFO]       ;es:di point to dos info
33607
33608                                       ; 14/12/2022
33609                                       ; ds <> cs
33610
33611                                       ;mov    ax,[si+4]
33612 00002901 8B4404                   mov     ax,[si+SYSDEV.ATT]    ;get attributes
33613                                   ; 12/12/2022
33614 00002904 F6C480                   test    ah,DEVTYP>>8 ; 80h
33615                                   ;test   ax,DEVTYP ; 8000h     ;test if block dev
33616 00002907 7426                     jz      short isblock
33617
33618                                   ;------ lets deal with character devices
33619
33620 00002909 2E800E[6919]02           or      byte [cs:setdevmarkflag],for_devmark ; 2
33621 0000290F E8F40D                   call    DevSetBreak           ;go ahead and alloc mem for device
33622                                   jc_edd:
33623 00002912 72BE                     jc      short erase_dev_do    ;device driver's init routine failed.
33624
33625                                       ; 12/12/2022
33626 00002914 A801                     test    al,ISCIN
33627                                   ;test   ax,ISCIN ; 1          ;is it a console in?
33628 00002916 7408                     jz      short tryclk
33629
33630 00002918 2689550C                 mov     [es:di+SYSI_CON],dx   ; es:di+12
33631 0000291C 268C5D0E                 mov     [es:di+SYSI_CON+2],ds ; es:di+14
33632                                   tryclk:
33633                                       ; 12/12/2022
33634 00002920 A808                     test    al,ISCLOCK
33635                                   ;test   ax,ISCLOCK ; 8        ;is it a clock device?
33636 00002922 7408                     jz      short golink
33637
33638 00002924 26895508                 mov     [es:di+SYSI_CLOCK],dx ; es:di+8
33639 00002928 268C5D0A                 mov     [es:di+SYSI_CLOCK+2],ds ; es:di+10
33640                                   golink:
33641 0000292C E9DF00                   jmp     linkit
33642
33643                                   ;------ deal with block device drivers
33644
33645                                   isblock:
33646 0000292F 2EA0[7C03]               mov     al,[cs:unitcount]     ;if no units found,erase the device
33647 00002933 08C0                     or      al,al
33648 00002935 749B                     jz      short erase_dev_do
33649                                   ;mov    [si+10],al
33650 00002937 88440A                   mov     [si+SYSDEV.NAME],al   ; number of units in name field
33651                                       ; 29/10/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
33652                                       ;add    [cs:driver_units],al
33653                                       ; 31/12/2022 - Retro DOS v4.2 (Modified MSDOS 6.21 IO.SYS)
33654 0000293A 2E0006[6A19]             add     [cs:driver_units],al  ; keep total for all drivers in file
33655                                   perdrv:
33656 0000293F 98                       cbw                           ; warning no device > 127 units
33657 00002940 89C1                     mov     cx,ax
33658 00002942 88E6                     mov     dh,ah
33659                                   ;mov    dl,[es:di+32]
33660 00002944 268A5520                 mov     dl,[es:di+SYSI_NUMIO] ;get number of devices
33661 00002948 88D4                     mov     ah,dl
33662 0000294A 00C4                     add     ah,al                 ; check for too many devices
33663 0000294C 80FC1A                   cmp     ah,26                 ; 'A' - 'Z' is 26 devices
33664 0000294F 7603                     jbe     short ok_block
33665 00002951 E976FF                   jmp     badnumblock
33666
33667                                   ok_block:
33668 00002954 2E800E[6919]02           or      byte [cs:setdevmarkflag],for_devmark ; 2
33669 0000295A E8A90D                   call    DevSetBreak           ; alloc the device
33670 0000295D 72B3                     jc      short jc_edd
33671 0000295F 26004520                 add     [es:di+SYSI_NUMIO],al ; update the amount
33672
33673 00002963 2E0006[8503]             add     [cs:drivenumber],al   ; remember amount for next device
33674 00002968 2EC51E[8103]             lds     bx,[cs:bpb_addr]      ; point to bpb array
33675                                   perunit:
33676 0000296D 2EC42E[6D02]             les     bp,[cs:DOSINFO]
33677                                   ;les    bp,[es:bp+SYSI_DPB]   ; get first dpb
33678                                       ; 11/12/2022
33679 00002972 26C46E00                 les     bp,[es:bp]
33680                                       ; 29/10/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
33681                                   ;les    bp,[es:bp+0]          ; [es:bp+SYSI_DPB]
33682                                   scandpb:
33683                                   ;cmp    word [es:bp+25],-1
33684 00002976 26837E19FF               cmp     word [es:bp+DPB.NEXT_DPB],-1
33685 0000297B 7406                     je      short foundpb
33686                                   ;les    bp,[es:bp+25]
33687 0000297D 26C46E19                 les     bp,[es:bp+DPB.NEXT_DPB]
33688 00002981 EBF3                     jmp     short scandpb
33689                                   foundpb:
33690 00002983 2EA1[3D24]               mov     ax,[cs:DevBrkAddr]
33691 00002987 26894619                 mov     [es:bp+DPB.NEXT_DPB],ax
33692 0000298B 2EA1[3F24]               mov     ax,[cs:DevBrkAddr+2]
33693 0000298F 2689461B                 mov     [es:bp+DPB.NEXT_DPB+2],ax
33694
33695 00002993 2EC42E[3D24]             les     bp,[cs:DevBrkAddr]
33696 00002998 2E8306[3D24]3D           add     word [cs:DevBrkAddr],DPBSIZ ; 33
33697                                                                 ; 08/09/2023
33698                                                                 ; (61 in PCDOS 7.1 IBMBIO.COM)
33699 0000299E E8440D                   call    RoundBreakAddr
```

```
33700
33701 000029A1 26C74619FFFF             mov     word [es:bp+DPB.NEXT_DPB],-1
33702 000029A7 26C64618FF               mov     byte [es:bp+DPB.FIRST_ACCESS],-1
33703
33704 000029AC 8B37                     mov     si,[bx]                 ;ds:si points to bpb
33705 000029AE 43                       inc     bx
33706 000029AF 43                       inc     bx                      ;point to next guy
33707                                   ;mov    [es:bp+DPB.DRIVE],dx
33708                                   ; 11/12/2022
33709 000029B0 26895600                 mov     [es:bp],dx ; 13/05/2019
33710                                   ; 29/10/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
33711                                   ;mov    [es:bp+0],dx            ; [es:bp+DPB.DRIVE]
33712
33713                                   ; 13/04/2024 - Retro DOS v5.0
33714                                   ; PCDOS 7.1 IBMBIO.COM
33715                                   ;;;
33716 000029B4 52                       push    dx
33717 000029B5 51                       push    cx              ; initialize FAT32 extended DPB parameters/fields
33718 000029B6 BA5241                   mov     dx,4152h        ; 'AR' signature for FAT32 extended DPB
33719 000029B9 31C9                     xor     cx,cx   ; 0
33720                                   ;mov    [es:bp+1Dh],cx
33721 000029BB 26894E1D                 mov     [es:bp+DPB.NEXT_FREE],cx ; last allocated cluster #
33722                                   ;cmp    [si+0Bh],cx     ; BPB.fatsecs16
33723 000029BF 394C0B                   cmp     [si+A_BPB.SECTORSPERFAT],cx ; 0
33724 000029C2 7514                     jnz     short set_dpb   ; FAT DPB (33 bytes)   -jnz-
33725                                                           ; FAT32 DPB (61 bytes) -jz-
33726                                   ;mov    [es:bp+39h],cx
33727 000029C4 26894E39                 mov     [es:bp+DPB.FAT32_NXTFREE],cx ; 0
33728                                   ;mov    [es:bp+3Bh],cx
33729 000029C8 26894E3B                 mov     [es:bp+DPB.FAT32_NXTFREE+2],cx ; 0
33730 000029CC 49                       dec     cx      ; 0FFFFh ; -1
33731                                   ;mov    [es:bp+1Fh],cx
33732 000029CD 26894E1F                 mov     [es:bp+DPB.FREE_CNT],cx ; -1 = unknown
33733                                   ;mov    [es:bp+21h],cx
33734 000029D1 26894E21                 mov     [es:bp+DPB.FREE_CNT+2],cx ; -1 = unknown
33735 000029D5 B95845                   mov     cx,4558h        ; 'EX' signature for FAT32 extended DPB
33736                                 set_dpb:
33737                                   ;;;
33738
33739 000029D8 B453                     mov     ah,SETDPB ; 53h                 ;hidden system call
33740 000029DA CD21                     int     21h
33741                                                   ; DOS - 2+ internal - TRANSLATE BIOS PARAMETER BLOCK
33742                                                   ; DS:SI -> BPB (BIOS Parameter Block)
33743                                                   ; ES:BP -> buffer for DOS Drive Parameter Block
33744                                   ; 13/04/2024
33745                                   ;;;
33746 000029DC 59                       pop     cx
33747 000029DD 5A                       pop     dx
33748                                   ;;;
33749
33750                                   ;mov    ax,[es:bp+2]
33751 000029DE 268B4602                 mov     ax,[es:bp+DPB.SECTOR_SIZE]
33752 000029E2 06                       push    es
33753 000029E3 2EC43E[6D02]             les     di,[cs:DOSINFO]         ;es:di point to dos info
33754                                   ;cmp    ax,[es:di+10h]
33755 000029E8 263B4510                 cmp     ax,[es:di+SYSI_MAXSEC]
33756 000029EC 07                       pop     es
33757                                   ; 13/04/2024
33758                                   ;jna    short iblk_1
33759                                   ;jmp    bad_bpb_size_sector
33760                                   ; 29/10/2022
33761 000029ED 777F                     ja      short bad_bpb_size_sector
33762                                 iblk_1:
33763 000029EF 1E                       push    ds
33764 000029F0 52                       push    dx
33765
33766 000029F1 2EC516[3924]             lds     dx,[cs:DevEntry]
33767                                   ;mov    [es:bp+13h],dx
33768 000029F6 26895613                 mov     [es:bp+DPB.DRIVER_ADDR],dx
33769                                   ;mov    [es:bp+15h],ds
33770 000029FA 268C5E15                 mov     [es:bp+DPB.DRIVER_ADDR+2],ds
33771
33772 000029FE 5A                       pop     dx
33773 000029FF 1F                       pop     ds
33774
33775 00002A00 42                       inc     dx
33776 00002A01 FEC6                     inc     dh
33777                                   ;loop   perunit
33778                                   ; 13/04/2024
33779                                   ;;;
33780 00002A03 49                       dec     cx              ; cx = cx - 1
33781                                                           ; cx = remain count from [cs:unitcount]
33782 00002A04 7403                     jz      short iblk_2    ; cx = 0 -> done
33783 00002A06 E964FF                   jmp     perunit         ; loop until cx is 0
33784                                 iblk_2:
33785                                   ;;;
33786
33787 00002A09 0E                       push    cs
33788 00002A0A 1F                       pop     ds
33789
33790 00002A0B E895E3                   call    TempCDS                 ; set cds for new drives
33791                                   ; 31/12/2022
33792                                   ; ds <> cs
33793                                 linkit:
33794 00002A0E 2EC43E[6D02]             les     di,[cs:DOSINFO]         ;es:di = dos table
33795 00002A13 268B4D22                 mov     cx,[es:di+SYSI_DEV]     ;dx:cx = head of list
33796 00002A17 268B5524                 mov     dx,[es:di+SYSI_DEV+2]
33797
33798 00002A1B 2EC536[3924]             lds     si,[cs:DevEntry]        ;ds:si = device location
33799 00002A20 26897522                 mov     [es:di+SYSI_DEV],si     ;set head of list in dos
33800 00002A24 268C5D24                 mov     [es:di+SYSI_DEV+2],ds
33801 00002A28 8B04                     mov     ax,[si]                 ;get pointer to next device
33802 00002A2A 2EA3[3924]               mov     [cs:DevEntry],ax        ;and save it
33803
33804 00002A2E 890C                     mov     [si],cx                 ;link in the driver
33805 00002A30 895402                   mov     [si+2],dx
33806                                 enddev:
33807 00002A33 5E                       pop     si
33808 00002A34 07                       pop     es
33809 00002A35 40                       inc     ax                      ;ax = ffff (no more devs if yes)?
33810 00002A36 740B                     jz      short coffj3
33811
33812 00002A38 2EFE06[6619]             inc     byte [cs:multdeviceflag] ; possibly multiple device driver.
33813 00002A3D E8E80C                   call    DevBreak                ; M009
33814                                   ; 11/12/2022
33815                                   ; ds = cs (DevBreak)
33816
33817                                   ; 03/04/2019 - Retro DOS v4.0
33818                                   ; MSDOS 6.21 IO.SYS - SYSINIT:290Dh
33819 00002A40 E9EDFD                   jmp     goodld                  ; otherwise pretend we loaded it in
33820                                 coffj3:
33821                                   ; 18/12/2022
33822                                   ; ax = 0
33823 00002A43 2EA2[6619]               mov     [cs:multdeviceflag],al ; 0
```

```
                                    ;mov    byte [cs:multdeviceflag],0 ; reset the flag
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
```

```
34072 00002B0B 803E[DD4A]00             cmp     byte [cntry_drv],0
34073                                   ;cmp    byte [cs:cntry_drv],0
34074 00002B10 7405                     je      short tryq_def
34075 00002B12 BA[DD4A]                 mov     dx,cntry_drv
34076 00002B15 EB03                     jmp     short tryq_openit
34077
34078                                 tryq_def:
34079 00002B17 BA[DF4A]                 mov     dx,cntry_root
34080                                 tryq_openit:
34081 00002B1A B8003D                   mov     ax,3D00h                ;open a file
34082 00002B1D F9                       stc
34083 00002B1E CD21                     int     21h
34084 00002B20 7242                     jc      short tryqfilebad       ;open failure
34085
34086                                   ; 14/12/2022
34087                                   ; ds = cs
34088 00002B22 A3[5C03]                 mov     [cntryfilehandle],ax
34089                                   ;mov    [cs:cntryfilehandle],ax       ;save file handle
34090 00002B25 89C3                     mov     bx,ax
34091 00002B27 A1[7A22]                 mov     ax,[p_cntry_code]
34092 00002B2A 8B16[7C22]               mov     dx,[p_code_page]
34093                                   ;mov    ax,[cs:p_cntry_code]
34094                                   ;mov    dx,[cs:p_code_page]   ;now,ax=country id,bx=filehandle
34095                                   ;mov    cx,[cs:memhi]
34096 00002B2E 8B0E[6403]               mov     cx,[memhi]
34097 00002B32 81C18001                 add     cx,384                  ;need 6k buffer to handle country.sys
34098                                                                   ;M023
34099                                   ; 14/12/2022
34100                                   ; ds = cs
34101 00002B36 3B0E[A502]               cmp     cx,[ALLOCLIM]
34102                                   ;cmp    cx,[cs:ALLOCLIM]
34103 00002B3A 7745                     ja      short tryqmemory        ;cannot allocate the buffer for country.sys
34104
34105 00002B3C BE[DD4A]                 mov     si,cntry_drv            ;ds:si -> cntry_drv
34106 00002B3F 803C00                   cmp     byte [si],0             ;default path?
34107 00002B42 7502                     jne     short tryq_set_for_dos
34108
34109 00002B44 46                       inc     si                      ;ds:si -> cntry_root
34110 00002B45 46                       inc     si                      ;ds:si -> cntry_root
34111
34112                                 tryq_set_for_dos:
34113                                   ; 14/12/2022
34114                                   ; ds = cs
34115 00002B46 C43E[7902]               les     di,[sysi_country]
34116                                   ;les    di,[cs:sysi_country]  ;es:di -> country info tab in dos
34117 00002B4A 57                       push    di                      ;save di
34118                                   ;add    di,8
34119 00002B4B 83C708                   add     di,country_cdpg_info.ccPath_CountrySys ; 8
34120 00002B4E E8D01E                   call    move_asciiz             ;set the path to country.sys in dos.
34121 00002B51 5F                       pop     di                      ;es:di -> country info tab again.
34122
34123                                   ; 14/12/2022
34124 00002B52 8B0E[6403]               mov     cx,[memhi]
34125                                   ;mov    cx,[cs:memhi]
34126 00002B56 8ED9                     mov     ds,cx
34127 00002B58 31F6                     xor     si,si                   ;ds:si -> 2k buffer to be used.
34128 00002B5A E8601D                   call    setdoscountryinfo       ;now do the job!!!
34129                                   ; ds <> cs ; 14/12/2022
34130 00002B5D 7325                     jnc     short tryqchkerr        ;read error or could not find country,code page combination
34131
34132 00002B5F 83F9FF                   cmp     cx,-1                   ;could not find matching country_id,code page?
34133 00002B62 74A1                     je      short tryqbad           ;then "invalid country code or code page"
34134
34135                                 tryqfilebad:
34136 00002B64 0E                       push    cs
34137 00002B65 07                       pop     es
34138 00002B66 2E803E[DD4A]00           cmp     byte [cs:cntry_drv],0 ;is the default file used?
34139 00002B6C 7405                     je      short tryqdefbad
34140
34141 00002B6E BE[DD4A]                 mov     si,cntry_drv
34142 00002B71 EB03                     jmp     short tryqbadload
34143
34144                                 tryqdefbad:                       ;default file has been used.
34145 00002B73 BE[DF4A]                 mov     si,cntry_root           ;es:si -> \country.sys in sysinit_seg
34146                                 tryqbadload:
34147 00002B76 E8B31E                   call    badload                 ;ds will be restored to sysinit_seg
34148                                   ; 31/12/2022 - Retro DOS v4.2 (Modified MSDOS 6.21 IO.SYS)
34149                                   ; (SYSINIT:2A69h)
34150 00002B79 8B0E[A302]               mov     cx,[CONFBOT] ; ds = cs (from badload)
34151                                   ;mov    cx,[cs:CONFBOT]
34152                                   ; 30/10/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
34153                                   ;mov    cx,[cs:top_of_cdss]
34154                                   ; 11/12/2022
34155                                   ; ds = cs
34156                                   ;mov    cx,[top_of_cdss]  ; mov cx,[CONFBOT]
34157 00002B7D 8EC1                     mov     es,cx                   ;restore es -> confbot.
34158 00002B7F EB13                     jmp     short coffj4
34159
34160                                 tryqmemory:
34161 00002B81 BA[1C51]                 mov     dx,insufmemory
34162                                 tryqchkerr:
34163                                   ;mov    cx,[cs:CONFBOT]
34164                                   ; 30/10/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
34165                                   ;mov    cx,[cs:top_of_cdss]
34166                                   ; 12/12/2022
34167 00002B84 0E                       push    cs
34168 00002B85 1F                       pop     ds
34169                                   ; 31/12/2022 - Retro DOS v4.2
34170 00002B86 8B0E[A302]               mov     cx,[CONFBOT] ; (MSDOS 6.21 IO.SYS, SYSINIT)
34171                                   ;mov    cx,[top_of_cdss]  ; mov cx,[CONFBOT]
34172 00002B8A 8EC1                     mov     es,cx                   ;restore es -> confbot seg
34173                                   ;push   cs
34174                                   ;pop    ds                      ;restore ds to sysinit_seg
34175 00002B8C 7306                     jnc     short coffj4            ;if no error,then exit
34176
34177 00002B8E E8C31E                   call    print                   ;else show error message
34178 00002B91 E82603                   call    error_line
34179                                 coffj4:
34180                                   ;mov    bx,[cs:cntryfilehandle]
34181                                   ; 11/12/2022
34182                                   ; ds = cs
34183 00002B94 8B1E[5C03]               mov     bx,[cntryfilehandle]
34184 00002B98 B43E                     mov     ah,3Eh
34185 00002B9A CD21                     int     21h                     ;close a file. don't care even if it fails.
34186 00002B9C E9E7F9                   jmp     coff
34187
34188                                 ;--------------------------------------------
34189
34190                                 cntry_error:
34191
34192                                 ;function: show "invalid country code or code page" messages,or
34193                                 ;          "error in country command" depending on the error code
34194                                 ;          in ax returned by sysparse;
34195                                 ;in:ax - error code
```

```
                                ;   ds - sysinitseg
                                ;   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                                            *
                                ;        { if (cx == 1) then /* first positional = stack count */       *
                                ;             p_stack_count = result_val._$P_picked_val;           *
                                ;          if (cx == 2) then /* second positional = stack size */      *
                                ;             p_stack_size = result_val._$P_picked_val;            *
                                ;        }                                                          *
                                ;     else /*error exit*/                                           *
                                ;        error exit;                                                *
                                ;   };                                                             *
                                ;   here check p_stack_count,p_stack_size if it meets the condition;  *
                                ;   if o.k.,then set stack_count,stack_size;                       *
                                ;    else error_exit;                                              *
                                ; };                                                               *
                                ;*******************************************************************************

                                tryk:
                                        ;if      stacksw

00002C27 80FC4B                         cmp     ah,CONFIG_STACKS ; 'K'
00002C2A 7402                       je      short do_tryk
                                skip_it4:
00002C2C EB79                       jmp     short trys      ; 15/12/2022
                                do_tryk:

                                ; 31/12/2022 - Retro DOS v4.2 (Modified MSDOS 6.21 IO.SYS)
                                ; (SYSINIT:2B33h)
                                ; 30/10/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
                                ;%if 0
                                ;ifdef      MULTI_CONFIG
00002C2E E8EB19                        call query_user              ; query the user if config_cmd
00002C31 72F9                          jc   short skip_it4        ; has the CONFIG_OPTION_QUERY bit set
                                ;endif
                                ;%endif     ; 30/10/2022

                                    ; 14/12/2022
                                    ; ds = cs

00002C33 BF[EA22]                   mov     di,stks_parms
00002C36 31C9                       xor     cx,cx
                                    ; 03/01/2023
                                    ;mov    dx,cx
                                do79:
00002C38 E82C02                     call    sysinit_parse
00002C3B 730B                       jnc     short if79              ; parse error

00002C3D BA[8B51]                   mov     dx,badstack             ; "invalid stack parameter"
00002C40 E8111E                     call    print                   ;  and show messages and end the search loop.
00002C43 E87402                     call    error_line
                                    ;jmp    sr79
                                    ; 11/12/2022
00002C46 EB39                       jmp     short sr79
                                if79:
00002C48 83F8FF                     cmp     ax,_$P_RC_EOL           ; end of line?
00002C4B 7412                       je      short en79              ; then end the $endloop

                                    ; 14/12/2022
                                    ; ds = cs

                                    ;;mov   ax,[cs:rv_dword]
                                    ;mov    ax,[cs:result_val+_$P_Result_Blk.Picked_Val]
                                    ;mov    ax,[rv_dword]
00002C4D A1[1B22]                   mov     ax,[result_val+_$P_Result_Blk.Picked_Val]

00002C50 83F901                     cmp     cx,1
00002C53 7505                       jne     short if83

                                    ; 14/12/2022
                                    ;mov    [cs:p_stack_count],ax
                                    ;jmp    short en83
00002C55 A3[1F23]                   mov     [p_stack_count],ax
00002C58 EBDE                       jmp     short do79
                                if83:
                                    ; 14/12/2022
                                    ;mov    [cs:p_stack_size],ax
00002C5A A3[2123]                   mov     [p_stack_size],ax
                                en83:
00002C5D EBD9                       jmp     short do79
                                en79:
                                    ; 14/12/2022
                                    ; ds = cs
00002C5F A1[1F23]                   mov     ax,[p_stack_count]
00002C62 09C0                       or      ax,ax
00002C64 741E                       jz      short if87

                                    ; 14/12/2022
                                    ;cmp    word [p_stack_count],0
                                    ;;cmp   word [cs:p_stack_count],0

                                    ;je     short if87

                                    ; 14/12/2022
00002C66 83F808                     cmp     ax, mincount ; 8
                                    ;cmp    word [cs:p_stack_count],mincount ; 8
                                    ; 15/12/2022
00002C69 721F                       jb      short en87
00002C6B 833E[2123]20               cmp     word [p_stack_size],minsize ; 32
                                    ;cmp    word [cs:p_stack_size],minsize ; 32
                                    ; 15/12/2022
00002C70 7218                       jb      short en87
                                if94:
                                    ; 14/12/2022
                                    ; ds = cs
                                    ; ax = [p_stack_count]
                                    ;mov    ax,[p_stack_count]
                                    ;;mov   ax,[cs:p_stack_count]
00002C72 A3[8C02]                   mov     [stack_count],ax
                                    ;mov    [cs:stack_count],ax
                                    ;mov    ax,[cs:p_stack_size]
00002C75 A1[2123]                   mov     ax,[p_stack_size]
                                    ;mov    [cs:stack_size],ax
00002C78 A3[8E02]                   mov     [stack_size],ax
                                    ;mov    word [cs:stack_addr],-1      ; stacks= been accepted.
00002C7B C706[9002]FFFF             mov     word [stack_addr],-1
                                sr79:
00002C81 E902F9                     jmp     coff

                                if87:
                                    ; 14/12/2022
00002C84 3906[2123]                 cmp     [p_stack_size],ax ; 0
00002C88 74E8                       je      short if94 ; ax = [p_stack_count] = 0
                                    ;cmp    word [cs:p_stack_size],0
                                    ;je     short if94
                                en87:
                                    ; 15/12/2022
```

```
                                    ; ([p_stack_count] is invalid, use default values)
                                    ; 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:
00002CA7 80FC53                         cmp     ah,CONFIG_SHELL ; 'S'
00002CAA 755A                       jne     short tryx

                                ; 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:                                                *
                                ; {                                                     *
                                ;   set di points to fcbs_parms;                       *
                                ;   set dx,cx to 0;                                     *
                                ;   while (end of command line)                         *
                                ;   { sysparse;                                         *
                                ;     if (no error) then                                *
                                ;        { if (cx == 1) then /* first positional = fcbs */          *
                                ;             fcbs = result_val._$P_picked_val;       *
                                ;          if (cx == 2) then /* second positional = keep */    *
                                ;             keep = result_val._$P_picked_val;       *
                                ;        }                                              *
                                ;     else /*error exit*/                               *
                                ;        error exit;                                    *
                                ;   };                                                  *
                                ; };                                                    *
                                ;*************************************************************************

                                tryx:
00002D06 80FC58                     cmp     ah,CONFIG_FCBS  ; 'X'
                                    ; 31/12/2022 - Retro DOS v4.2 (Modified MSDOS 6.21 IO.SYS)
00002D09 7534                       jne     short try1
                                    ; 30/10/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
                                    ;jne    short tryy      ; comment command

                                ; 31/12/2022 - Retro DOS v4.2
                                ; 30/10/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
                                ;%if 0
                                ;ifdef      MULTI_CONFIG
00002D0B E80E19                     call    query_user      ; query the user if config_cmd
00002D0E 722F                       jc      short try1      ; has the CONFIG_OPTION_QUERY bit set
                                ;endif
                                ;%endif ; 30/10/2022

00002D10 BF[9E22]                   mov     di,fcbs_parms
00002D13 31C9                       xor     cx,cx
                                    ; 03/01/2023
                                    ;mov    dx,cx
                                do98:
00002D15 E84F01                     call    sysinit_parse
                                        ; 03/01/2023
                                    ;jnc    short if98      ; parse error
                                       ;call      badparm_p       ;  and show messages and end the search loop.
                                    ;jmp    short sr98
                                    ;------------------------
                                    ; 03/01/2023 - Retro DOS v4.2 (Modified MSDOS 6.21 IO.SYS)
00002D18 723C                       jc      short badparm_p_coff
                                if98:
00002D1A 83F8FF                     cmp     ax,_$P_RC_EOL  ; end of line?
00002D1D 7412                       je      short en98      ; then end the $endloop

                                    ;;mov   al,[cs:rv_dword]
                                    ;mov    al,[cs:result_val+_$P_Result_Blk.Picked_Val]
                                    ; 15/12/2022
                                    ; ds = cs
00002D1F A0[1B22]                   mov     al,[result_val+_$P_Result_Blk.Picked_Val]
00002D22 83F901                     cmp     cx,1            ; the first positional?
00002D25 7505                       jne     short if102
                                    ;mov    [cs:p_fcbs],al
                                    ; 15/12/2022
00002D27 A2[D322]                   mov     [p_fcbs],al
                                    ;jmp    short en102
00002D2A EBE9                       jmp     short do98
                                if102:
                                    ;mov    [cs:p_keep],al
                                    ; 15/12/2022
00002D2C A2[D422]                   mov     [p_keep],al
                                en102:
00002D2F EBE4                       jmp     short do98
                                en98:
                                    ; 15/12/2022
                                    ; ds = cs
00002D31 A0[D322]                   mov     al,[p_fcbs]
00002D34 A2[A002]                   mov     [FCBS],al
00002D37 C606[A102]00               mov     byte [KEEP],0
                                    ;mov    al,[cs:p_fcbs]  ; M017
                                    ;mov    [cs:FCBS],al    ; M017
                                    ;mov    byte [cs:KEEP],0 ; M017
                                sr98:
00002D3C E947F8                     jmp     coff

                                ; 31/12/2022 - Retro DOS v4.2
                                %if 0

                                ; 30/10/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
                                ;-------------------------------------------------------------------------
                                ; comment= do nothing. just decrease chrptr,and increase count for correct
                                ;          line number
                                ;-------------------------------------------------------------------------

                                tryy:
                                    cmp     ah,CONFIG_COMMENT ; 'Y'
                                    jne     short try0

                                donothing:
                                    ; 15/12/2022
                                    ; ds = cs
                                    dec     word [chrptr]
                                    inc     word [count]
                                    ; 02/11/2022
                                    ;dec    word [cs:chrptr]
                                    ;inc    word [cs:count]

                                    jmp     coff

                                ;-------------------------------------------------------------------------
                                ; rem command
                                ;-------------------------------------------------------------------------

                                try0:                           ; do nothing with this line.
                                    cmp     ah,CONFIG_REM ; '0'
                                    je      short donothing

                                %endif

                                ; 07/04/2019 - Retro DOS v4.0

                                ;-------------------------------------------------------------------------
                                ; switches command
                                ;-------------------------------------------------------------------------

                                ;*************************************************************************
                                ;                                                                        *
                                ; function: parse the option switches specified.                        *
```

```
                                ; note - this command is intended for the future use also.            *
                                ; 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
```

```
35064 00002DA6 8ED8                     mov     ds,ax
35065 00002DA8 750A                     jne     short if117
35066                                   ; 14/04/2024
35067 00002DAA C606[7E04]00             mov     byte [keyrd_func],0 ; 4E5h ; use the conventional keyboard functions
35068                                                   ; BIOSDATA:047Eh for PCDOS 7.1 IBMBIO.COM
35069 00002DAF C606[7F04]01             mov     byte [keysts_func],1 ; 4E6h (for MSDOS 6.21 IO.SYS)
35070                                                   ; BIOSDATA:047Fh for PCDOS 7.1 IBMBIO.COM
35071                                 if117:
35072                                   ; 15/12/2022
35073                                   ; ds <> cs
35074 00002DB4 2EA0[9623]               mov     al,[cs:p_swit_t]                        ;M059
35075 00002DB8 A2[8B04]                 mov     [t_switch],al  ; 4F2h (for MSDOS 6.21 IO.SYS)       ;M059
35076                                           ; 14/04/2024   ; BIOSDATA:048Bh for PCDOS 7.1 IBMBIO.COM
35077 00002DBB 2E803E[9723]00           cmp     byte [cs:p_swit_w],0                    ;M063
35078 00002DC1 740E                     je      short skip_dos_flag                     ;M063
35079 00002DC3 06                       push    es
35080 00002DC4 53                       push    bx
35081 00002DC5 B452                     mov     ah,GET_IN_VARS ; 52h                    ;M063
35082 00002DC7 CD21                     int     21h                                     ;M063
35083                                           ; DOS - 2+ internal - GET LIST OF LISTS
35084                                           ; Return: ES:BX -> DOS list of lists
35085                                   ;or     bytes [es:86h],2
35086 00002DC9 26800E860002             or      byte [es:DOS_FLAG_OFFSET],SUPPRESS_WINA20 ; 2       ;M063
35087 00002DCF 5B                       pop     bx
35088 00002DD0 07                       pop     es
35089                                 skip_dos_flag:                                    ;M063
35090 00002DD1 1F                       pop     ds
35091                                 sr110:
35092 00002DD2 E9B1F7                   jmp     coff
35093
35094                                 ; 31/12/2022 - Retro DOS v4.2 (Modified MSDOS 6.21 IO.SYS)
35095                                 ; (SYSINIT:2D14h)
35096                                 ; 30/10/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
35097                                 ;%if 0
35098
35099                                 tryv:
35100
35101                                 ;ifdef  MULTI_CONFIG
35102                                 ;-----------------------------------------------------------------------
35103                                 ; set command (as in "set var=value<cr/lf>")
35104                                 ;-----------------------------------------------------------------------
35105
35106 00002DD5 80FC56                     cmp     ah,CONFIG_SET  ; 'V'
35107 00002DD8 750F                     jne     short tryn
35108 00002DDA E83F18                   call    query_user      ; query the user if config_cmd
35109 00002DDD 720A                     jc      short tryn     ; has the CONFIG_OPTION_QUERY bit set
35110 00002DDF E83614                   call    copy_envvar     ; copy var at ES:SI to "config_wrkseg"
35111 00002DE2 73EE                     jnc     short sr110    ; no error
35112                                 err:
35113 00002DE4 E8D300                   call    error_line      ; whoops, display error in line XXX
35114 00002DE7 EBE9                     jmp     short sr110     ; jump to coff (to skip to next line)
35115
35116                                 ;-----------------------------------------------------------------------
35117                                 ; numlock command (as in "numlock=on|off")
35118                                 ;-----------------------------------------------------------------------
35119                                 tryn:
35120 00002DE9 80FC4E                   cmp     ah,CONFIG_NUMLOCK  ;'N'
35121                                   ;jne    short tryy
35122                                   ; 14/04/2024 - Retro DOS v5.0 (PCDOS 7.1 IBMBIO.COM)
35123 00002DEC 750C                     jne     short tryt
35124
35125 00002DEE E82B18                   call    query_user      ; query the user if config_cmd
35126 00002DF1 7238                     jc      short tryy     ; has the CONFIG_OPTION_QUERY bit set
35127 00002DF3 E8B710                   call    set_numlock
35128 00002DF6 72EC                     jc      short err
35129 00002DF8 EBD8                     jmp     short sr110    ; all done
35130
35131                                 ;endif  ;MULTI_CONFIG
35132
35133                                 ; 14/04/2024 - Retro DOS v5.0 (PCDOS 7.1 IBMBIO.COM)
35134                                 ;-----------------------------------------------------------------------
35135                                 ; dosdata command
35136                                 ;-----------------------------------------------------------------------
35137                                 tryt:
35138                                   ;cmp    ah,54h ; 'T'
35139 00002DFA 80FC54                   cmp     ah,CONFIG_DOSDATA ; 'T' ; PCDOS 7 new config cmd
35140 00002DFD 752C                     jne     short tryy
35141
35142 00002DFF E81A18                   call    query_user
35143 00002E02 7227                     jc      short tryy
35144
35145 00002E04 BF[C823]                 mov     di,dosdata_parms
35146 00002E07 31C9                     xor     cx,cx
35147                                   ; 14/04/2024 - Retro DOS v5.0
35148                                   ;mov    dx,cx ; 0
35149                                 do120:
35150 00002E09 E85B00                   call    sysinit_parse
35151 00002E0C 7303                     jnc     short if120
35152
35153                                   ;call   badparm_p
35154                                   ;jmp    short en120
35155                                   ; 14/04/2024 - Retro DOS v5.0
35156 00002E0E E945FF                   jmp     badparm_p_coff
35157                                 if120:
35158                                   ;cmp    ax,0FFFFh
35159 00002E11 83F8FF                   cmp     ax,_$P_RC_EOL ; -1 ; end of line?
35160 00002E14 7422                     jz      short en120
35161 00002E16 803E[1822]01             cmp     byte [result_val_itag],1 ; tag 1 (UMB)
35162                                                   ; [result_val+_$P_Result_Blk.Item_Tag]
35163 00002E1B 7507                     jnz     short if121
35164 00002E1D C606[6E03]01             mov     byte [dosdata_umb],1 ; DOSDATA=UMB (1) NOUMB (0)
35165                                   ;jmp    short sr120
35166                                   ; 14/04/2024
35167 00002E22 EBE5                     jmp     short do120
35168                                 if121:
35169 00002E24 C606[6E03]00             mov     byte [dosdata_umb],0 ; DOSDATA=UMB (1) NOUMB (0)
35170                                 sr120:
35171 00002E29 EBDE                     jmp     short do120
35172                                   ; 14/04/2024
35173                                 ;en120:
35174                                   ;jmp    coff
35175
35176                                 ; 30/10/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
35177                                 ;-----------------------------------------------------------------------
35178                                 ; comment= do nothing. just decrease chrptr,and increase count for correct
35179                                 ;          line number
35180                                 ;-----------------------------------------------------------------------
35181
35182                                   ; 31/12/2022
35183                                 tryy:
35184 00002E2B 80FC59                   cmp     ah,CONFIG_COMMENT ; 'Y'
35185 00002E2E 750B                     jne     short try0
35186
35187                                 donothing:
```

```
                                    ; 15/12/2022
                                    ; ds = cs
00002E30 FF0E[5A03]                 dec     word [chrptr]
00002E34 FF06[5603]                 inc     word [count]
                                    ; 02/11/2022
                                    ;dec    word [cs:chrptr]
                                    ;inc    word [cs:count]
                                en120:              ; 14/04/2024
00002E38 E94BF7                     jmp     coff

                                ;--------------------------------------------------------------------------
                                ; rem command
                                ;--------------------------------------------------------------------------

                                try0:                               ; do nothing with this line.
00002E3B 80FC30                     cmp     ah,CONFIG_REM ; '0'
00002E3E 74F0                       je      short donothing

                                ;%endif     ; 30/10/2022

                                ; 30/10/2022
                                ; (MSSOS 5.0 IO.SYS - SYSINIT:29D7h)

                                ;--------------------------------------------------------------------------
                                ; bogus command
                                ;--------------------------------------------------------------------------

                                tryz:
00002E40 80FCFF                         cmp     ah,0FFh             ;null command? (BUGBUG - who sets FFh anyway?)
                                    ; 31/12/2022
00002E43 74EB                       je      short donothing
                                    ; 02/11/2022
                                    ;je     short tryz_donothing

00002E45 FF0E[5A03]                 dec     word [chrptr]
00002E49 FF06[5603]                 inc     word [count]
00002E4D EB37                       jmp     short badop

                                ; 31/12/2022
                                ;   ; 02/11/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 IO.SYS)
                                ;tryz_donothing:
                                ;   jmp     donothing

                                ; 07/04/2019 - Retro DOS v4.0

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

                                ; 31/12/2022 - Retro DOS v4.2 (Modified MSDOS 6.21 IO.SYS)
                                ; (SYSINIT:2D5Dh)

                                ; 11/07/2023 - Retro DOS v4.2 IO.SYS (optimization)

                                ; 30/10/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
                                ;
                                ;***CheckProtmanArena -- special hack for adjusting alloclim with Protman$
                                ;
                                ;   adjusts alloclim if Protman$ reduced our arena through a manual hack.
                                ;
                                CheckProtmanArena:
                                    ; 08/09/2023
                                    ; ds = cs
00002E4F 06                         push    es
                                    ;mov    ax,[cs:area]   ; get our arena header
00002E50 A1[6803]                   mov     ax,[area] ; 08/09/2023
00002E53 48                         dec     ax
00002E54 8EC0                       mov     es,ax
                                    ;add    ax,[es:ARENA.SIZE]
00002E56 2603060300                 add     ax,[es:3]       ; find end of arena
00002E5B 40                         inc     ax
                                    ; 08/09/2023
00002E5C 3B06[A502]                 cmp     ax,[ALLOCLIM]
                                    ;cmp    ax,[cs:ALLOCLIM] ; is it less than alloclim?
00002E60 7703                       ja      short CheckProtmanDone

                                    ;mov    [cs:ALLOCLIM],ax ; reduce alloclim then
                                    ; 08/09/2023
00002E62 A3[A502]                   mov     [ALLOCLIM],ax
                                CheckProtmanDone:
00002E65 07                         pop     es
00002E66 C3                         retn

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

                                sysinit_parse:

                                ;--------------------------------------------------------------------------
                                ;set up registers for sysparse
                                ;in)es:si -> command line in confbot
                                ;   di -> offset of the parse control definition.
                                ;
                                ;out)       calls sysparse.
                                ;   carry will set if parse error.
                                ;   *** the caller should check the eol condition by looking at ax
                                ;   *** after each call.
                                ;   *** if no parameters are found,then ax will contain a error code.
                                ;   *** if the caller needs to look at the synomym@ of the result,
                                ;   ***  the caller should use cs:@ instead of es:@.
                                ;   cx register should be set to 0 at the first time the caller calls this
                                ;    procedure.
                                ;   ax - exit code
                                ;   bl - terminated delimeter code
                                ;   cx - new positional ordinal
                                ;   si - set to pase scanned operand
                                ;   dx - selected result buffer
                                ;--------------------------------------------------------------------------

                                    ; 03/01/2023 - Retro DOS v4.2 (Modified MSDOS 6.21 IO.SYS)
                                    ; (SYSINIT:2D78h)

                                    ; 14/04/2024 - Retro DOS v5.0
                                    ; (PCDOS 7.1 IBMBIO.COM - SYSINIT:32F3h)

                                    ; 09/09/2023 - Retro DOS v4.2 IO.SYS (Optimization)
                                    ; ds = cs
00002E67 8C06[6D19]                 mov     [badparm_seg],es        ;save the pointer to the parm
00002E6B 8936[6B19]                 mov     [badparm_off],si        ;we are about to parse for badparm msg.

                                    ; 24/10/2022
00002E6F 06                         push    es                      ;save es,ds
00002E70 1E                         push    ds

00002E71 06                         push    es
00002E72 1F                         pop     ds                      ;now ds:si -> command line
```

```
35312 00002E73 0E                       push    cs
35313 00002E74 07                       pop     es                      ;now es:di -> control definition
35314
35315                                   ; 09/09/2023
35316                                   ;mov    [cs:badparm_seg],ds     ;save the pointer to the parm
35317                                   ;mov    [cs:badparm_off],si     ;we are about to parse for badparm msg.
35318
35319                                   ;mov    dx,0
35320                                   ; 04/01/2023
35321 00002E75 29D2                     sub     dx,dx ; 0
35322 00002E77 E89BEB                   call    SysParse
35323                                   ;cmp    ax,_$P_No_Error         ; 0     ;no error
35324                                   ; 06/09/2023
35325 00002E7A 21C0                     and     ax,ax
35326
35327                                 ;**cas note: when zero true after cmp, carry clear
35328
35329                                   ;je     short ll4
35330                                   ; 24/10/2022 (MSDOS 5.0 IO.SYS compatibility, SYSINIT:2A02h)
35331                                   ; 12/12/2022
35332 00002E7C 7405                     je      short en4 ; cf=0
35333 00002E7E 83F8FF                   cmp     ax,_$P_RC_EOL ; 0FFFFh;or the end of line?
35334                                   ;jne    short if4
35335                                   ; 12/12/2022
35336 00002E81 7400                     je      short en4 ; cf=0
35337                                   ; 06/09/2023
35338                                   ; cf=1
35339
35340                                 ; 12/12/2022
35341                                 ;ll4:
35342                                 ;   ; 12/12/2022
35343                                 ;   ; cf=0
35344                                 ;   ;clc
35345                                 ;   jmp     short en4
35346
35347                                 if4:
35348                                   ; 24/10/2022
35349                                   ; 06/09/2023 (cf=1)
35350                                   ;stc
35351                                 en4:
35352 00002E83 1F                       pop     ds
35353 00002E84 07                       pop     es
35354 00002E85 C3                       retn
35355
35356                                 ; 11/12/2022
35357                                 %if 0
35358
35359                                 ;--------------------------------------------------------------------------
35360                                 ;
35361                                 ; procedure : badop_p
35362                                 ;
35363                                 ;             same thing as badop,but will make sure to set ds register back
35364                                 ;             to sysinitseg and return back to the caller.
35365                                 ;
35366                                 ;--------------------------------------------------------------------------
35367
35368                                 badop_p:
35369                                   push    cs
35370                                   pop     ds              ;set ds to configsys seg.
35371                                   mov     dx,badopm
35372                                   call    print
35373                                       ;call       error_line
35374                                   ;retn
35375                                   ; 11/12/2022
35376                                   jmp     error_line
35377
35378                                 %endif
35379
35380                                 ;--------------------------------------------------------------------------
35381                                 ;
35382                                 ; label : badop
35383                                 ;
35384                                 ;--------------------------------------------------------------------------
35385
35386                                 badop:
35387 00002E86 BA[4C50]                 mov     dx,badopm       ;want to print command error "unrecognized command..."
35388 00002E89 E8C81B                   call    print
35389 00002E8C E82B00                   call    error_line      ;show "error in config.sys ..." .
35390 00002E8F E9F4F6                   jmp     coff
35391
35392                                 ;--------------------------------------------------------------------------
35393                                 ;
35394                                 ; procedure : badparm_p
35395                                 ;
35396                                 ;             show "bad command or parameters - xxxxxx"
35397                                 ;             in badparm_seg,badparm_off -> xxxxx
35398                                 ;
35399                                 ;--------------------------------------------------------------------------
35400
35401                                   ; 24/10/2022
35402                                 badparm_p:
35403                                   ; 11/12/2022
35404                                   ; ds = cs
35405                                   ; 11/12/2022
35406                                   ;push  ds ; *
35407 00002E92 52                       push    dx
35408 00002E93 56                       push    si
35409
35410                                   ; 11/12/2022
35411                                   ; ds = cs
35412                                   ;push  cs
35413                                   ;pop   ds
35414
35415 00002E94 BA[7350]                 mov     dx,badparm
35416 00002E97 E8BA1B                   call    print                   ; "bad command or parameters - "
35417 00002E9A C536[6B19]               lds     si,[badparm_ptr]
35418
35419                                 ;   print "xxxx" until cr.
35420
35421                                 do1:
35422 00002E9E 8A14                     mov     dl,[si]                 ; get next character
35423 00002EA0 80FA0D                   cmp     dl,cr ; 0Dh             ; is a carriage return?
35424 00002EA3 7407                     je      short en1               ; exit loop if so
35425
35426 00002EA5 B402                     mov     ah,2 ; STD_CON_OUTPUT ; function 2
35427 00002EA7 CD21                     int     21h                     ; display character
35428 00002EA9 46                       inc     si                      ; next character
35429 00002EAA EBF2                     jmp     short do1
35430                                 en1:
35431 00002EAC 0E                       push    cs
35432 00002EAD 1F                       pop     ds
35433
35434 00002EAE BA[7050]                 mov     dx,crlfm
35435 00002EB1 E8A01B                   call    print
```

```
00002EB4 E80300                     call    error_line

00002EB7 5E                         pop     si
00002EB8 5A                         pop     dx
                                    ; 11/12/2022
                                    ;pop    ds ; *
                                badparmp_ret:
00002EB9 C3                         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?
00002ECF 720C                       jb      short sln_last

00002ED1 31D2                       xor     dx,dx
00002ED3 F7F1                       div     cx      ; cx = 10
00002ED5 80CA30                     or      dl,30h          ; add "0" (= 30h) to make it an ascii.
00002ED8 8815                       mov     [di],dl
00002EDA 4F                         dec     di
00002EDB EBF0                       jmp     short sln_loop

                                sln_last:
00002EDD 0C30                       or      al,30h ; "0"
00002EDF 8805                       mov     [di],al
00002EE1 89FA                       mov     dx,di
00002EE3 E86E1B                     call    print           ; show it.
00002EE6 5F                         pop     di
                                    ; 11/12/2022
                                    ;pop    ds
00002EE7 07                         pop     es
00002EE8 C3                         retn

                                ; 07/04/2019 - Retro DOS v4.0
                                ; (MSDOS 6.21 IO.SYS, SYSINIT:2E44h)

                                ;--------------------------------------------------------------------------
                                ;
                                ; procedure : ProcDOS
                                ;
                                ;   Process the result of DOS= parsing
                                ;
                                ;   result_val._$P_item_tag       = 1 for DOS=HIGH
                                ;                                 = 2 for DOS=LOW
                                ;                                 = 3 for DOS=UMB
                                ;                                 = 4 for DOS=NOUMB
                                ;--------------------------------------------------------------------------

                                    ; 01/11/2022 - Retro DOS v4.0 (Modififed MSDOS 5.0 IO.SYS)
                                    ; (SYTSINIT:2AB5h)
                                ProcDOS:
                                    ; 01/01/2023
                                    ; ds = cs
00002EE9 30E4                       xor     ah,ah
                                    ;;mov   al,[cs:result_val_itag]
                                    ;mov    al,[cs:result_val+_$P_Result_Blk.Item_Tag]
                                    ; 01/01/2023
00002EEB A0[1822]                   mov     al,[result_val+_$P_Result_Blk.Item_Tag]
00002EEE 48                         dec     ax
00002EEF 7415                       jz      short pd_hi
00002EF1 48                         dec     ax
00002EF2 740E                       jz      short pd_lo
00002EF4 48                         dec     ax
00002EF5 7405                       jz      short pd_umb
                                    ;;mov   byte [cs:DevUMB],0
                                    ; 18/12/2022
                                    ;mov    byte [cs:DevUMB],ah ; 0
                                    ; 01/01/2023
00002EF7 8826[4224]                 mov     byte [DevUMB],ah ; 0
00002EFB C3                         retn
                                pd_umb:
                                    ; 01/01/2023
00002EFC C606[4224]FF               mov     byte [DevUMB],0FFh
                                    ;mov    byte [cs:DevUMB],0FFh
00002F01 C3                         retn
                                pd_lo:
                                    ; 01/01/2023
00002F02 A2[6C02]                   mov     [runhigh],al ; 0
                                    ; 18/12/2022
                                    ;mov    [cs:runhigh],al ; 0
                                    ;;mov   byte [cs:runhigh],0
00002F05 C3                         retn
                                pd_hi:
                                    ; 01/01/2023
00002F06 C606[6C02]FF               mov     byte [runhigh],0FFh
                                    ;mov    byte [cs:runhigh],0FFh
                                limx:       ; 11/12/2022
00002F0B C3                         retn

                                ;--------------------------------------------------------------------------
                                ;
                                ; procedure : LieInt12Mem
                                ;
                                ;   Input : DevEntry points to Device Start address (offset == 0)
                                ;           alloclim set to the limit of low memory.
                                ;
                                ;   Output : none
                                ;
                                ;   Changes the ROM BIOS variable which stores the total low memory
                                ;   If a 3com device driver (any character device with name 'PROTMAN$')
                                ;   is being loaded alloclim is converted into Ks and stored in 40:13h
                                ;   Else if a device driver being loaded into UMB the DevLoadEnd is
                                ;   converted into Ks and stored in 40:13h
                                ;
                                ;--------------------------------------------------------------------------

                                LieInt12Mem:
                                    ; 11/12/2022
                                    ; ds = cs
00002F0C A1[A502]                   mov     ax,[ALLOCLIM]
                                    ;mov    ax,[cs:ALLOCLIM]      ; lie INT 12 as alloclim
                                                                  ; assuming that it is 3Com
00002F0F E84200                     call    IsIt3Com              ; Is it 3Com driver?
00002F12 740A                       jz      short lim_set         ; yes, lie to him differently
                                    ; 13/05/2019
                                    ;cmp    byte [cs:DeviceHi],0  ; Is the DD being loaded in UMB
                                    ;je     short limx            ; no, don't lie
                                    ;mov    ax,[cs:DevLoadEnd]    ; lie INT 12 as end of UMB
                                    ; 11/12/2022
                                    ; ds = cs
00002F14 803E[5124]00               cmp     byte [DeviceHi],0
00002F19 74F0                       je      short limx
00002F1B A1[3724]                   mov     ax,[DevLoadEnd]
                                lim_set:
                                    ; 01/11/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
                                    ; 11/12/2022
                                    ;call   SetInt12Mem
                                ;limx:
                                    ;retn

                                    ;jmp    short SetInt12Mem

                                ;--------------------------------------------------------------------------
                                ;
                                ; procedure : SetInt12Mem
```

```
                                ;
                                ;   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

00003040 E80E00                     call   parseL
00003043 73D0                       jnc    short pv10     ; If no carry, go back and look for more

00003045 4E                         dec    si             ; Else, back up and exit.
00003046 EB03                       jmp    short pvErr    ; AX has already been set by parseL

                                pvE1:       ;mov   ax,3
00003048 B80300                     mov    ax,PV_InvSwt   ; Unrecognized switch passed
0000304B 4E                     pvErr:      dec    si
0000304C 4E                         dec    si
0000304D F9                         stc
                                pvX:;popreg <es, ds, di>
0000304E 07                         pop    es
0000304F 1F                         pop    ds
                                    ; 01/01/2023
                                    ;pop   di ; * ; (not required) ; 01/01/2023
00003050 C3                         retn

                                ; ---------------------------------------------------------------------------
                                ;*** parseL - parses ":nnnn[,nnnn][;nnnn[,nnnn]]*" for ParseVar
                                ; ---------------------------------------------------------------------------
                                ; ENTRY:    ES:SI points to colon
                                ; EXIT:     ES:SI points to first character not parsed
                                ; ERROR:    Carry set; rewind three characters and return (see ParseVar)
                                ; USES:     ES:SI, flags, AX, CX, DX, variables in highvar.inc
                                ; ---------------------------------------------------------------------------
                                ; If the string here is terminated with anything other than whitespace or a
                                ; switchchar (perhaps it's /S or another /L:... ), then we return with carry
                                ; set, indicating that they've screwed up the syntax.  The 3-character rewind
                                ; makes sure the app /L: is reported as being the culprit.
                                ; ---------------------------------------------------------------------------

                                parseL:
00003051 AC                         lodsb
00003052 3C3A                       cmp    al,':'         ; Make sure they did /L:
00003054 754E                       jne    short plE1     ; If they didn't, return with carry set.

                                ; ----------------------------------------
                                ; PL10--ES:SI = a UMB number, after /L: or ;
                                ; ----------------------------------------

00003056 E8DB00                 pl10:       call   GetXNum        ; After this, 'tis ",size" or ";umb" or " mod"
00003059 724F                       jc     short plE2     ; And error if it's a bad number.
0000305B E89D01                     call   convUMB        ; Convert any address to a UMB number

0000305E 88C1                       mov    cl,al          ; Remember the UMB number
00003060 E87600                     call   stowUMB        ; Mark this UMB # as used;
00003063 7245                       jc     short plE2     ; If it was already marked, it'll error

                                    ;call  incArgc        ; Each UMB number is another arg for LH to skip
00003065 2EFE06[3224]               inc    byte [cs:fm_argc] ; 08/04/2019 - Retro DOS v4.0

0000306A AC                         lodsb
0000306B 3C3B                       cmp    al,';'         ; Did "umb;" ?
0000306D 74E7                       je     short pl10     ; Yep: go back and get another UMB.

0000306F E84900                     call   isWhite        ; Did "umb " ?
00003072 743B                       jz     short plX      ; Yep: return (it'll go back to whitespace)

00003074 E83900                     call   isEOL          ; Did "umb" ?
00003077 7435                       jz     short plSwX    ; If so, backup and exit like everything's ok

                                    ;cmp   al,'/'
00003079 3C2F                       cmp    al,SWTCH       ; Did "umb/" ? (as in, "/L:1,100;2/S")
0000307B 7431                       je     short plSwX    ; If so, back up ES:SI one character and return

0000307D 3C2C                       cmp    al,','         ; Did "umb," ?
0000307F 7523                       jne    short plE1     ; Just what the heck DID they do? Return error.

                                ; --- Read a size -----------------------------------------------------------

00003081 E8B000                     call   GetXNum        ; Stop on "size;" or "size " or anything else
00003084 721E                       jc     short plE1     ; And error if it's a bad size.

00003086 E81601                     call   toPara         ; Convert from bytes to paragraphs

00003089 E8CE01                     call   stowSiz        ; CL still has the UMB number for this routine

                                    ;call  incArgc        ; Each UMB size is another arg for LH to skip
0000308C 2EFE06[3224]               inc    byte [cs:fm_argc] ; 08/04/2019 - Retro DOS v4.0

00003091 AC                         lodsb
00003092 3C3B                       cmp    al,';'         ; They did "umb,size;", so get another UMB.
00003094 74C0                       je     short pl10     ;

00003096 E82200                     call   isWhite        ; Did it end with whitespace?
00003099 7414                       jz     short plX      ; If so, we're done here--go back.

0000309B E81200                     call   isEOL          ; Did they do "umb,size" and end??? (stupid)
0000309E 740E                       jz     short plSwX    ; If so, backup and exit like everything's ok

                                    ;cmp   al,'/'
000030A0 3C2F                       cmp    al,SWTCH       ; Did they do "umb,size/" ?
000030A2 740A                       je     short plSwX    ; If so, again, we're done here.
                                plE1:
                                    ;mov   ax,1
000030A4 B80100                     mov    ax,PV_InvArg   ; If not, we don't know WHAT they did...
000030A7 4E                         dec    si
000030A8 F9                         stc
000030A9 C3                         retn

                                plE2:       ;mov   ax,2
000030AA B80200                     mov    ax,PV_BadUMB   ; In this case, they've specified a UMB twice
                                    ; 12/12/2022
                                    ; cf=1
                                    ;stc
000030AD C3                         retn
                                plSwX:
000030AE 4E                         dec    si             ; If we hit a '/' character, back up one char
                                                          ; so the whitespace checker will see it too.
                                plX:; 12/12/2022
                                    ; cf=0
                                    ;clc                  ; Then just return with carry clear, so
000030AF C3                         retn                  ; ParseVar will go about its business.

                                ; ---------------------------------------------------------------------------
                                ;*** incArgc - increments fm_argc, for use with LoadHigh command-line parsing
                                ; ---------------------------------------------------------------------------
                                ; ENTRY:    None
                                ; EXIT:     None
                                ; ERROR:    None
```

```
                                ; USES:     fm_argc, flags
                                ; ----------------------------------------------------------------------------

                                ;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 td3E      ; 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 87DA                       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 87DA                       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)
                                ; ----------------------------------------------

```

```
37048 00003225 41                      hus10:      inc     cx              ; For each UMB:
37049                                      ; 01/01/2023
37050 00003226 80F910                      cmp     cl,MAXUMB
37051                                      ;cmp    cx,MAXUMB ; 16
37052 00003229 730E                        jae     short hus20
37053
37054 0000322B 88C8                        mov     al,cl           ; (stopping as soon as we're outside of the
37055 0000322D 06                          push    es
37056 0000322E E8A200                      call    findUMB         ; valid range of UMBs)
37057 00003231 07                          pop     es              ; push/pop: trash what findumb finds. :-)
37058 00003232 7205                        jc      short hus20
37059
37060                                      ; 02/01/2023
37061                                      ;push   cx ; *
37062 00003234 E84F01                      call    _hideUMB_       ; hide what we need to hide.
37063                                      ;pop    cx ; *
37064
37065 00003237 EBEC                        jmp     short hus10
37066                                  hus20:
37067                                      ;call   GetLoadUMB      ; Now check if they offered /L:0
37068                                      ; 01/01/2023
37069                                      ; ds = cs
37070                                      ;mov    al,[UmbLoad]
37071                                      ;;mov   al,[cs:UmbLoad] ; 09/04/2019 - Retro DOS v4.0
37072 00003239 800E[FF23]00                or      byte [UmbLoad],0
37073                                      ;or     al,al           ; --Is the load UMB 0? (-1==unspecified)
37074 0000323E 7503                        jnz     short husX      ; If not, we're done.
37075
37076 00003240 E86802                      call    hl_unlink       ; If so, however, fix UMBs and strategy.
37077                                  husX:
37078 00003243 07                          pop     es
37079                                      ; 01/01/2023
37080                                      ;pop    ds
37081                                      ;pop    cx
37082                                      ;pop    ax
37083 00003244 C3                          retn
37084
37085                                  ; ---------------------------------------------------------------------------
37086                                  ;*** GetLoadUMB - Returns the load UMB number in AL (-1 if not specified)
37087                                  ; ---------------------------------------------------------------------------
37088                                  ; ENTRY:  None
37089                                  ; EXIT:   AL == load UMB
37090                                  ; ERROR:  None
37091                                  ; USES:   Flags, AX
37092                                  ; ---------------------------------------------------------------------------
37093
37094                                  ;GetLoadUMB:
37095                                  ;   ;getdata al, UmbLoad
37096                                  ;   push    ds
37097                                  ;   push    cs
37098                                  ;   pop     ds
37099                                  ;   mov     al,[UmLoad]
37100                                  ;   pop     ds
37101                                  ;   retn
37102
37103                                  ; ---------------------------------------------------------------------------
37104                                  ;*** GetLoadSize - Returns the load UMB minimum size (0 if not specified)
37105                                  ; ---------------------------------------------------------------------------
37106                                  ; ENTRY:  None
37107                                  ; EXIT:   AX == load UMB minimum size
37108                                  ; ERROR:  None
37109                                  ; USES:   Flags, AX
37110                                  ; ---------------------------------------------------------------------------
37111
37112                                  ; 08/09/2023 - Retro DOS v4.2 IO.SYS (Optimization)
37113                                  %if 0
37114                                      ; 01/01/2023 - Retro DOS v4.2
37115                                  GetLoadSize:
37116                                      ; 09/04/2019 - Retro DOS v4.0
37117                                      ;mov    al,[cs:UmbLoad]
37118                                      ; 01/01/2023
37119                                      ; ds = cs
37120                                      mov     al,[UmbLoad]
37121                                      ;jmp    short GetSize
37122
37123                                      ;push   bx
37124                                      ;;push  si
37125                                      ;push   ds
37126                                      ;push   cs
37127                                      ;pop    ds
37128
37129                                      ;mov    al,[UmbLoad]
37130
37131                                      ;xor    ah,ah                   ;    ax==UMB
37132                                      ;mov    bx,UmbSize              ;    bx==array
37133                                      ;shl    al,1                     ;    ax==offset
37134                                      ;;add   ax,bx                   ;    ax==element index
37135                                      ;;mov   si,ax                   ; ds:si==element index
37136
37137                                      ;;lodsw                         ;    hh
37138
37139                                      ;add    bx,ax
37140                                      ;mov    ax,[bx]
37141
37142                                      ;pop    ds
37143                                      ;;pop   si
37144                                      ;pop    bx
37145                                      ;retn
37146                                  %endif
37147
37148                                  ; ---------------------------------------------------------------------------
37149                                  ;*** GetSize - Returns the UMB in AL's minimum size (0 if not specified)
37150                                  ; ---------------------------------------------------------------------------
37151                                  ; ENTRY:  AL == a UMB number
37152                                  ; EXIT:   AX == UMB minimum size, as specified by the user
37153                                  ; ERROR:  None
37154                                  ; USES:   Flags, AX
37155                                  ; ---------------------------------------------------------------------------
37156
37157                                  ; 08/09/2023 - Retro DOS v4.2 IO.SYS (Optimization)
37158                                  GetLoadSize:
37159                                      ; ds = cs
37160                                      ;mov    al,[UmbLoad]
37161                                      ; al = [UmbLoad]
37162                                      ; ....
37163
37164                                      ; 01/01/2023 - Retro DOS v4.2
37165                                  GetSize:
37166                                      ; 09/04/2019 - Retro DOS v4.0
37167
37168                                      ;push   bx ; 01/01/2023
37169                                      ;push   si
37170                                      ;push   ds
37171                                      ;push   cs
```

```
                                    ;pop    ds
                                    xor     ah,ah                   ;    ax==UMB
00003245 30E4                       mov     bx,UmbSize              ;    bx==array
00003247 BB[1024]                   shl     al,1                     ;    ax==offset
0000324A D0E0                       ;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
                                
0000325A 88CB                       mov     bl,cl                   ; Now bl==UMB number, AX==size
0000325C B700                       mov     bh,0                    ;     bx==UMB number, AX==size
0000325E D0E3                       shl     bl,1                    ;     bx==offset into array, AX=size
                                    ;mov    [es:bx+UmbSize],ax      ; Store the size
                                    ; 01/01/2023
00003260 2E8987[1024]               mov     [cs:bx+UmbSize],ax      ; Store the size
                                
                                    ; 01/01/2023
                                    ;pop    es
                                    ;;pop   di ; ?
                                    ;pop    bx
                                
00003265 C3                         retn
                                %endif
                                
                                ; ---------------------------------------------------------------------------
                                ;*** hideUMB - marks as HIDDEN all FREE elements in UMB passed as AL
                                ; ---------------------------------------------------------------------------
                                ; ENTRY:    AL must indicate a valid UMB; 0==conv && is invalid.
                                ; EXIT:     None; free elements in UMB marked as hidden
                                ; ERROR:    None
                                ; USES:     Flags
                                ; ---------------------------------------------------------------------------
                                
                                    ; 01/01/2023 - Retro DOS v4.2
                                hideUMB:
                                    ; 02/01/2023
00003266 52                         push    dx ; (*)
                                    ; 01/01/2023
                                    ;push   ax
00003267 06                         push    es
                                
00003268 E86800                     call    findUMB ; (*)   ; Returns with carry if err, else ES == MCB
0000326B 7224                       jc      short huX
                                
                                ; ------------------------------------------------
                                ; HU10--ES - MCB inside UMB; if it's a system MCB,
                                ;            we're not in the same UMB, so exit.
                                ; ------------------------------------------------
                                
0000326D E84AFF                 hu10:       call    isSysMCB        ; Returns with ZF set if owner is SYSTEM
00003270 741F                       jz      short huX       ; If it is, we've finished the UMB.
                                    ;call   isFreeMCB       ; Returns with ZF set if owner is 0
00003272 26830E010000               or      word [es:ARENA.OWNER],0
00003278 7503                       jnz     short hu20
                                
0000327A E81700                     call    hideMCB
                                hu20:
                                    ;mov    al,[es:ARENA.SIGNATURE]
                                    ;cmp    al,arena_signature_end  ;'Z'
                                    ; 19/07/2023
0000327D 26803E00005A               cmp     byte [es:ARENA.SIGNATURE],'Z'
00003283 740C                       jz      short huX       ; 'Z' means this was the last MCB... that's it.
                                
                                    ;NextMCB es,ax          ; Go on forward.
00003285 8CC0                       mov     ax,es
                                    ;add    ax,[es:3]
00003287 2603060300                 add     ax,[es:ARENA.SIZE]
0000328C 40                         inc     ax
0000328D 8EC0                       mov     es,ax
                                
0000328F EBDC                       jmp     short hu10
                                huX:
00003291 07                         pop     es
                                    ; 01/01/2023
                                    ;pop    ax
                                    ; 02/01/2023
00003292 5A                         pop     dx ; (*)
00003293 C3                         retn
                                
                                ; 02/01/2023
                                %if 0
                                
                                ; ---------------------------------------------------------------------------
                                ;*** isTiny - returns with ZF set if user didn't specify /S
                                ; ---------------------------------------------------------------------------
                                ; ENTRY:    None
                                ; EXIT:     ZF set if user DIDN'T specify /S
                                ; ERROR:    None
                                ; USES:     Flags
                                ; ---------------------------------------------------------------------------
                                
                                    ; 02/01/2023 - Retro DOS v4.2 (Modified MSDOS 6.21 IO.SYS)
                                isTiny:
                                    ; 02/01/2023
                                    ;push   ax
                                
                                    ;getdata al,fUmbTiny
                                    ;
                                    ;push   ds
                                    ;push   cs
                                    ;pop    ds
                                    ;mov    al,[fUmbTiny]
                                    ;pop    ds
                                
                                    ; 09/09/2023
                                    ;mov    al,[cs:fUmbTiny]
                                    ; 02/01/2023
                                    ; ds = cs
                                    mov     al,[fUmbTiny]
                                
                                    or      al,al
                                    ; 02/01/2023
                                    ;pop    ax
                                    retn
                                
                                %endif
                                
                                ; ---------------------------------------------------------------------------
                                ;*** isFreeMCB - returns with ZF set if current MCB (ES:0) is FREE
                                ; ---------------------------------------------------------------------------
                                ; ENTRY:    ES:0 should point to an MCB
                                ; EXIT:     ZF set if MCB is free, else !ZF
                                ; ERROR:    None
                                ; USES:     Flags
                                ; ---------------------------------------------------------------------------
                                
                                ;isFreeMCB:
                                ;   or      word [es:ARENA.OWNER],0
                                ;   retn
                                
                                ; ---------------------------------------------------------------------------
```

```
37420                                   ;*** hideMCB - marks as HIDDEN the MCB at ES:0
37421                                   ; ----------------------------------------------------------------------------
37422                                   ; ENTRY:    ES:0 should point to an MCB
37423                                   ; EXIT:     None; MCB marked as HIDDEN
37424                                   ; ERROR:    None
37425                                   ; USES:     None
37426                                   ; ----------------------------------------------------------------------------
37427
37428                                   hideMCB:
37429 00003294 26C70601000800             mov     word [es:ARENA.OWNER],SystemPSPOwner ; 8
37430 0000329B 26C70608004849             mov     word [es:ARENA.NAME+0], 'HI' ; 4948h
37431 000032A2 26C7060A004444             mov     word [es:ARENA.NAME+2], 'DD' ; 4444h
37432 000032A9 26C7060C00454E             mov     word [es:ARENA.NAME+4], 'EN' ; 4E45h
37433 000032B0 26C7060E002020             mov     word [es:ARENA.NAME+6], '  ' ; 2020h
37434 000032B7 C3                         retn
37435
37436                                   ; ----------------------------------------------------------------------------
37437                                   ;*** unHideMCB - marks as FREE the MCB at ES:0
37438                                   ; ----------------------------------------------------------------------------
37439                                   ; ENTRY:    ES:0 should point to an MCB
37440                                   ; EXIT:     None; MCB marked as FREE
37441                                   ; ERROR:    None
37442                                   ; USES:     None
37443                                   ; ----------------------------------------------------------------------------
37444
37445                                       ; 03/01/2023 - Retro DOS v4.2 (Modified MSDOS 6.21 IO.SYS)
37446
37447                                   unHideMCB:
37448                                       ; 03/01/2023
37449                                       ;push   ax
37450 000032B8 26C70601000000             mov     word [es:ARENA.OWNER],FreePSPOwner ; 0
37451 000032BF B82020                     mov     ax,'  ' ; 2020h
37452 000032C2 26A30800                   mov     [es:ARENA.NAME+0],ax
37453 000032C6 26A30A00                   mov     [es:ARENA.NAME+2],ax
37454 000032CA 26A30C00                   mov     [es:ARENA.NAME+4],ax
37455 000032CE 26A30E00                   mov     [es:ARENA.NAME+6],ax
37456                                     ; 03/01/2023
37457                                     ;pop    ax
37458 000032D2 C3                         retn
37459
37460                                   ; ----------------------------------------------------------------------------
37461                                   ;*** findUMB - makes ES:0 point to the first MCB in UMB given as AL
37462                                   ;           -- returns UmbHEAD pointer (0x9FFF) if passed AL==0
37463                                   ; ----------------------------------------------------------------------------
37464                                   ; ENTRY:    AL should be to a valid UMB number
37465                                   ; EXIT:     ES:0 points to first MCB in UMB (_not_ the 8+SC MCB that heads it)
37466                                   ; ERROR:    Carry set if couldn't reach UMB (too high)
37467                                   ; USES:     Flags, ES
37468                                   ; ----------------------------------------------------------------------------
37469
37470                                       ; 01/01/2023 - Retro DOS v4.2 (Modified MSDOS 6.21 IO.SYS)
37471                                       ; (SYSINIT:3344h)
37472                                   findUMB:
37473                                       ; 01/01/2023
37474                                       ;push   ax
37475                                       ; 02/01/2023
37476 000032D3 51                         push    cx ; *
37477                                     ;push   dx
37478
37479 000032D4 30E4                       xor     ah,ah           ; Zap ah, so al==ax
37480
37481 000032D6 89C2                       mov     dx,ax           ; Store the to-be-found UMB number in DX
37482
37483 000032D8 E8D2FE                     call    UmbHead         ; Returns first UMB segment in AX
37484                                     ; 22/07/2023
37485                                     ;mov    es,ax ; *
37486 000032DB 31C9                       xor     cx,cx           ; Pretend we're on UMB 0 for now...
37487
37488                                     ; 22/07/2023
37489                                   fu10:
37490 000032DD 8EC0                       mov     es,ax ; * ; **
37491                                   ; ------------------------------------------
37492                                   ; FU10--CX - This UMB number; 0 == conventional
37493                                   ;       DX - The UMB number they're looking for
37494                                   ;       ES - The current MCB address
37495                                   ; ------------------------------------------
37496
37497                                   ;fu10:
37498 000032DF 39D1                       cmp     cx,dx           ; If CX==DX, we've found the UMB we're
37499 000032E1 7417                       je      short fuX       ; searching for--so exit.
37500
37501 000032E3 E8D4FE                     call    isSysMCB        ; Returns with ZF set if owner is SYSTEM
37502 000032E6 7501                       jnz     short fu20
37503
37504 000032E8 41                         inc     cx              ; If it _was_ SYSTEM, we're in a new UMB.
37505                                   fu20:
37506                                       ;mov    al,[es:ARENA.SIGNATURE]
37507                                       ;cmp    al,arena_signature_end ; 'Z'
37508                                       ; 19/07/2023
37509 000032E9 26803E00005A               cmp     byte [es:ARENA.SIGNATURE],arena_signature_end
37510 000032EF 7408                       je      short fuE       ; 'Z' means this was the last MCB... that's it.
37511
37512                                     ;NextMCB es,ax          ; Go on forward.
37513                                     ; 22/07/2023
37514                                     ; ax = es
37515                                     ;mov    ax,es ; * ; 22/07/2023
37516                                     ;add    ax,[es:3]
37517 000032F1 2603060300                 add     ax,[es:ARENA.SIZE]
37518 000032F6 40                         inc     ax
37519                                     ; 22/07/2023
37520                                     ;mov    es,ax ; **
37521 000032F7 EBE4                       jmp     short fu10
37522                                   fuE:
37523 000032F9 F9                         stc
37524                                   fuX:
37525                                       ; 01/01/2023
37526                                       ;pop    dx
37527                                       ; 02/01/2023
37528 000032FA 59                         pop     cx ; *
37529                                     ;pop    ax              ; The address is already in ES.
37530 000032FB C3                         retn
37531
37532                                   ; ----------------------------------------------------------------------------
37533                                   ;*** BigFree - makes ES:0 point to the largest free MCB in UMB given as AL
37534                                   ; ----------------------------------------------------------------------------
37535                                   ; ENTRY:    AL should be to a valid UMB number
37536                                   ; EXIT:     ES:0 points to largest free MCB in UMB, AX returns its size
37537                                   ; ERROR:    Carry set if couldn't reach UMB (0 or too high)
37538                                   ; USES:     Flags, ES
37539                                   ; ----------------------------------------------------------------------------
37540
37541                                       ; 01/01/2023 - Retro DOS v4.2
37542                                   BigFree:
37543                                       ; 01/01/2023
```

```
37544                                   ;push   bx
37545 000032FC 51                       push    cx
37546
37547 000032FD E8D3FF                   call    findUMB             ; Returns with CF if err, else ES==MCB
37548 00003300 723A                     jc      short bfX           ; (would be "jc bfE"; it just does stc)
37549
37550 00003302 31DB                     xor     bx,bx               ; Segment address of largest free MCB
37551 00003304 31C9                     xor     cx,cx               ; Size of largest free MCB
37552
37553                                 ; ------------------------------------------
37554                                 ; BF10--ES - Current MCB address
37555                                 ;       BX - Address of largest free MCB so far
37556                                 ;       CX - Size of largest free MCB so far
37557                                 ; ------------------------------------------
37558
37559                                 bf10:
37560 00003306 E8B1FE                   call    isSysMCB            ; If we've left the MCB, we're done.
37561 00003309 7428                     jz      short bf30
37562
37563                                   ;call   isFreeMCB           ; Returns with ZF set if owner is 0
37564 0000330B 26830E010000             or      word [es:ARENA.OWNER],0
37565 00003311 750C                     jnz     short bf20
37566
37567 00003313 26A10300                 mov     ax,[es:ARENA.SIZE]
37568                                   ;cmp    cx,[es:ARENA.SIZE]  ; Compare sizes...
37569 00003317 39C1                     cmp     cx,ax
37570                                   ;jg     short bf20          ; Unless we're bigger,
37571                                   ; 19/07/2023
37572 00003319 7D04                     jge     short bf20
37573
37574 0000331B 8CC3                     mov     bx,es               ; Store this new element's address,
37575                                   ;mov    cx,[es:ARENA.SIZE]  ; and its size.
37576 0000331D 89C1                     mov     cx,ax
37577
37578                                 bf20:
37579                                   ;mov    al,[es:ARENA.SIGNATURE]
37580                                   ;cmp    al,arena_signature_end; 'Z'
37581                                   ; 19/07/2023
37582                                   ;cmp    byte [es:0],'Z'
37583 0000331F 26803E00005A             cmp     byte [es:ARENA.SIGNATURE],arena_signature_end
37584 00003325 740C                     jz      short bf30          ; 'Z' means this was the last MCB.
37585
37586                                   ;NextMCB es,ax              ; Go on forward.
37587 00003327 8CC0                     mov     ax,es
37588                                   ;add    ax,[es:3]
37589 00003329 2603060300               add     ax,[es:ARENA.SIZE]
37590 0000332E 40                       inc     ax
37591 0000332F 8EC0                     mov     es,ax
37592
37593 00003331 EBD3                     jmp     short bf10
37594
37595 00003333 8EC3                   bf30:     mov     es,bx               ; Return the address
37596 00003335 89C8                     mov     ax,cx               ; Return the size
37597 00003337 09DB                     or      bx,bx
37598 00003339 7501                     jnz     short bfX           ; (if size==0, there's nothing free)
37599                                 bfE:
37600 0000333B F9                       stc
37601                                 bfX:
37602 0000333C 59                       pop     cx
37603                                   ; 01/01/2023
37604                                   ;pop    bx
37605 0000333D C3                       retn
37606
37607                                 ; ---------------------------------------------------------------------------
37608                                 ;*** isSpecified - sets ZF if UMB in AL wasn't specified in DH/LH line.
37609                                 ; ---------------------------------------------------------------------------
37610                                 ; ENTRY:    AL should be to a valid UMB number
37611                                 ; EXIT:     ZF set if UMB wasn't specified, ZF clear if it was
37612                                 ; ERROR:    None
37613                                 ; USES:     Flags
37614                                 ; ---------------------------------------------------------------------------
37615
37616                                     ; 02/01/2023 - Retro DOS v4.2
37617
37618                                 isSpecified:
37619                                     ; 02/01/2023
37620                                     ;push   ax
37621
37622 0000333E 30FF                     xor     bh,bh
37623 00003340 88C3                     mov     bl,al
37624
37625                                   ;getdata al,DS:UmbUsed[bx]
37626                                   ;
37627                                   ;push   ds
37628                                   ;push   cs
37629                                   ;pop    ds
37630                                   ;mov    al,[bx+UmbUsed]
37631                                   ;pop    ds
37632
37633                                   ;mov    al,[cs:bx+UmbUsed]
37634                                   ; 02/01/2023
37635                                   ; ds = cs
37636 00003342 8A87[0024]               mov     al,[bx+UmbUsed]
37637
37638 00003346 08C0                     or      al,al               ; Sets ZF if al==0 (ie, if unspecified)
37639
37640                                   ; 09/09/2023
37641                                   ; 02/01/2023
37642                                   ;pop    ax
37643
37644 00003348 C3                       retn
37645
37646                                 ; ---------------------------------------------------------------------------
37647                                 ;*** shrinkMCB - breaks an MCB into two pieces, the lowest one's size==AX
37648                                 ; ---------------------------------------------------------------------------
37649                                 ; ENTRY:    AX == new size, ES:0 == current MCB
37650                                 ; EXIT:     None; MCB broken if carry clear
37651                                 ; ERROR:    Carry set if MCB isn't as large as AX+0x20 (not a useful split)
37652                                 ; USES:     Flags
37653                                 ; ---------------------------------------------------------------------------
37654                                 ; If the size of the to-be-split MCB isn't at least 0x20 bytes greater than
37655                                 ; the specified new size, the split is useless; if it's only 0x10 bytes, that
37656                                 ; 0x10 will be used to make a header that mentions a 0-byte free space, and
37657                                 ; that just sucks up 0x10 bytes for nothing. So we make 0x20 bytes the
37658                                 ; minimum for performing a split.
37659                                 ; ---------------------------------------------------------------------------
37660
37661                                 MIN_SPLIT_SIZE      equ     20h
37662
37663                                     ; 02/01/2023 - Retro DOS v4.2 (Modified MSDOS 6.21 IO.SYS)
37664
37665                                 shrinkMCB:
37666                                     ;pushreg <bx,cx,es>
37667                                     ; 02/01/2023
```

```
                                ;push   bx
00003349 51                     push    cx
0000334A 06                     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,
```

```
000033A0 39D8                       cmp     ax,bx           ; Ah... if (specified > max free)
000033A2 7607                       jbe     short hu_10

000033A4 88C8                       mov     al,cl           ;  Then mark that UMB as unused. Nya nya.
000033A6 E81DFD                     call    unMarkUMB
000033A9 EB10                       jmp     short hu_20
                                hu_10:
                                    ;call   isTiny          ; Returns ZF clear if user specified /S
                                    ;jz     short hu_20
                                    ; 02/01/2023
                                ;isTiny:
                                    ;mov    al,[fUmbTiny] ; ds = cs
                                    ;or     al,al
000033AB 800E[FC23]00               or      byte [fUmbTiny],0
000033B0 7409                       jz      short hu_20

000033B2 E894FF                     call    shrinkMCB       ; They specified /S, so shrink the MCB to AX
000033B5 7204                       jc      short hu_20     ; Ah... if didn't shrink after all, skip this:

000033B7 8CC2                       mov     dx,es
000033B9 EB09                       jmp     short hu_30     ; Skip the spec check.. we wanna hide this one.

000033BB 89C8                   hu_20:      mov     ax,cx
000033BD E87EFF                     call    isSpecified     ; If they specified this UMB, we're done...
000033C0 7510                       jnz     short hu_X      ; so leave.

000033C2 31D2                       xor     dx,dx
                                hu_30:
000033C4 88C8                       mov     al,cl

000033C6 E89DFE                     call    hideUMB         ; Hides everything in UMB #al

000033C9 09D2                       or      dx,dx           ; Did we shrink a UMB? If not, DX==0,
000033CB 7405                       jz      short hu_X      ; So we should leave.

000033CD 8EC2                       mov     es,dx           ; Ah, but if it isn't, DX==the MCB's address;
000033CF E8E6FE                     call    unHideMCB       ; Un-hides the lower portion of that MCB.
                                hu_X:
000033D2 07                         pop     es
                                    ; 01/01/2023
                                    ;pop    dx
                                    ;pop    bx
000033D3 C3                         retn

                                ; ---------------------------------------------------------------------------
                                ;*** UnFreeze - Marks FROZEN elements as FREE
                                ; ---------------------------------------------------------------------------
                                ; Entry:  None
                                ; Exit:   None; all 8+FROZEN elements are marked as FREE, from any UMB.
                                ; Error:  None
                                ; Uses:   Flags
                                ; ---------------------------------------------------------------------------

                                    ; 03/01/2023 - Retro DOS v4.2 (Modified MSDOS 6.21 IO.SYS)
                                UnFreeze:
                                    ; 03/01/2023
                                    ;push   ax
000033D4 06                         push    es

000033D5 E8D5FD                     call    UmbHead         ; Returns with carry if err, else ES == MCB
000033D8 721C                       jc      short ufX

                                    ; 22/07/2023
                                uf10:
000033DA 8EC0                       mov     es,ax ; *

                                ; ------------------------------
                                ; UF10--ES - Current MCB address
                                ; ------------------------------

                                ;uf10:
000033DC E81900                     call    isFrozMCB       ; Returns with ZF set if MCB is FROZEN
000033DF 7505                       jnz     short uf20
000033E1 E8D4FE                     call    unHideMCB
                                    ; 09/09/2023
                                    ; ax <> es
000033E4 8CC0                       mov     ax,es ; *
                                uf20:
                                    ;mov    al,[es:ARENA.SIGNATURE]
                                    ;cmp    al,arena_signature_end ; 'Z'
                                    ; 22/07/2023
000033E6 26803E00005A               cmp     byte [es:ARENA.SIGNATURE],arena_signature_end ; 'Z'
000033EC 7408                       jz      short ufX       ; 'Z' means this was the last MCB.. that's it.

                                    ;NextMCB es,ax          ; Go on forward.
                                    ; 22/07/2023
                                    ; ax = es
                                    ;mov    ax,es ; *
                                    ;add    ax,[es:3]
000033EE 2603060300                 add     ax,[es:ARENA.SIZE]
000033F3 40                         inc     ax
                                    ; 22/07/2023
                                    ;mov    es,ax
000033F4 EBE4                       jmp     short uf10
                                ufX:
000033F6 07                         pop     es
                                    ; 03/01/2023
                                    ;pop    ax
000033F7 C3                         retn

                                ; ---------------------------------------------------------------------------
                                ;*** isFrozMCB - returns with ZF set if current MCB (ES:0) is FROZEN
                                ; ---------------------------------------------------------------------------
                                ; ENTRY:    ES:0 should point to an MCB
                                ; EXIT:     ZF set if MCB is frozen, else !ZF
                                ; ERROR:    None
                                ; USES:     Flags
                                ; ---------------------------------------------------------------------------

                                isFrozMCB:
                                    ;push   ax

                                    ;mov    ax,[es:ARENA.OWNER]    ; Check the owner...
                                    ;cmp    ax,SystemPSPOwner      ; 8 (for US OR Japan) is valid
000033F8 26833E010008               cmp     word [es:ARENA.OWNER],SystemPSPOwner
000033FE 7522                       jne     short ifmX

                                    ;mov    ax,[es:ARENA.NAME+0]
                                    ;cmp    ax,'FR' ; 5246h
00003400 26813E08004652             cmp     word [es:ARENA.NAME+0],'FR'
00003407 7519                       jne     short ifmX
                                    ;mov    ax,[es:ARENA.NAME+2]
```

```
37916                                   ;cmp    ax,'OZ' ; 5A4Fh
37917 00003409 26813E0A004F5A           cmp     word [es:ARENA.NAME+2],'OZ'
37918 00003410 7510                     jne     short ifmX
37919                                   ;mov    ax,[es:ARENA.NAME+4]
37920                                   ;cmp    ax,'EN' ; 4E45h
37921 00003412 26813E0C00454E           cmp     word [es:ARENA.NAME+4],'EN'
37922 00003419 7507                     jne     short ifmX
37923                                   ;mov    ax,[es:ARENA.NAME+6]
37924                                   ;cmp    ax,'  ' ; 2020h
37925 0000341B 26813E0E002020           cmp     word [es:ARENA.NAME+6],'  '
37926                                 ifmX:
37927                                   ;pop    ax
37928 00003422 C3                       retn
37929
37930                                 ; ------------------------------------------------------------------------------
37931                                 ;*** frezMCB - marks as 8+FROZEN the MCB at ES:0
37932                                 ; ------------------------------------------------------------------------------
37933                                 ; ENTRY:    ES:0 should point to an MCB
37934                                 ; EXIT:     None; MCB frozen
37935                                 ; ERROR:    None
37936                                 ; USES:     None
37937                                 ; ------------------------------------------------------------------------------
37938
37939                                 frezMCB:
37940 00003423 26C70601000800           mov     word [es:ARENA.OWNER],SystemPSPOwner ; 8
37941 0000342A 26C70608004652           mov     word [es:ARENA.NAME+0],'FR'
37942 00003431 26C7060A004F5A           mov     word [es:ARENA.NAME+2],'OZ'
37943 00003438 26C7060C00454E           mov     word [es:ARENA.NAME+4],'EN'
37944 0000343F 26C7060E002020           mov     word [es:ARENA.NAME+6],'  '
37945 00003446 C3                       retn
37946
37947                                 ; ------------------------------------------------------------------------------
37948                                 ;*** FreezeUM - Marks FROZEN all UM elements now FREE, save those in load UMB
37949                                 ; ------------------------------------------------------------------------------
37950                                 ; Entry:  None
37951                                 ; Exit:   None; all free elements not in load UMB marked as 8+FROZEN
37952                                 ; Error:  None
37953                                 ; Uses:   Flags
37954                                 ; ------------------------------------------------------------------------------
37955
37956                                     ; 01/01/2023 - Retro DOS v4.2
37957                                 FreezeUM:
37958                                     ; 01/01/2023
37959                                     ;push   ax
37960                                     ;push   cx
37961                                     ;push   dx
37962 00003447 06                       push    es
37963
37964                                     ;;call  GetLoadUMB
37965                                     ; 01/01/2023
37966                                     ; ds = cs
37967                                     ;mov    al,[cs:UmbLoad] ; 19/04/2019 - Retro DOS v4.0
37968 00003448 A0[FF23]                 mov     al,[UmbLoad]
37969
37970 0000344B 30E4                     xor     ah,ah           ; Zap ah, so al==ax
37971 0000344D 89C2                     mov     dx,ax           ; Store the load UMB in DX, so we can skip it
37972
37973 0000344F E85BFD                   call    UmbHead         ; Returns first UMB segment in AX
37974                                   ; 22/07/2023
37975                                   ;mov    es,ax ; *
37976 00003452 31C9                     xor     cx,cx           ; Pretend we're on UMB 0 for now...
37977
37978                                   ; 22/07/2023
37979                                 fum10:
37980 00003454 8EC0                     mov     es,ax ; *
37981
37982                                 ; ----------------------------------------
37983                                 ; FUM10--ES - Current MCB address
37984                                 ;        CX - Current UMB number
37985                                 ;        DX - UMB number to skip (load UMB)
37986                                 ; ----------------------------------------
37987
37988                                 ;fum10:
37989 00003456 E861FD                   call    isSysMCB        ; Returns with ZF set if owner is SYSTEM
37990 00003459 7501                     jnz     short fum20
37991
37992 0000345B 41                       inc     cx              ; If it _was_ SYSTEM, we're in a new UMB.
37993                                 fum20:
37994 0000345C 39D1                     cmp     cx,dx           ; If this is the load UMB, we don't want to
37995 0000345E 740B                     je      short fum30     ; freeze anything... so skip that section.
37996
37997                                   ;call   isFreeMCB       ; Oh. If it's not free, we can't freeze it
37998 00003460 26830E010000             or      word [es:ARENA.OWNER],0
37999 00003466 7503                     jnz     short fum30     ; either.
38000
38001 00003468 E8B8FF                   call    frezMCB
38002                                 fum30:
38003                                   ;mov    al,[es:ARENA.SIGNATURE]
38004                                   ;cmp    al,arena_signature_end ; 'Z'
38005                                   ; 22/07/2023
38006 0000346B 26803E00005A             cmp     byte [es:ARENA.SIGNATURE],arena_signature_end ; 'Z'
38007 00003471 7408                     je      short fumX      ; 'Z' means this was the last MCB.. that's it.
38008
38009                                   ;NextMCB es, ax               ; Go on forward.
38010                                   ; 22/07/2023
38011                                   ; ax = es
38012                                   ;mov    ax,es
38013                                   ;add    ax,[es:3]
38014 00003473 2603060300               add     ax,[es:ARENA.SIZE]
38015 00003478 40                       inc     ax
38016                                   ; 22/07/2023
38017                                   ;mov    es,ax ; *
38018 00003479 EBD9                     jmp     short fum10
38019
38020 0000347B 07                     fumX:     pop     es
38021                                   ; 01/01/2023
38022                                   ;pop    dx
38023                                   ;pop    cx
38024                                   ;pop    ax
38025 0000347C C3                       retn
38026
38027                                 ; ------------------------------------------------------------------------------
38028                                 ;*** UmbTest - returns with carry set if UMBs are not available, else CF==false
38029                                 ; ------------------------------------------------------------------------------
38030                                 ; ENTRY:    None
38031                                 ; EXIT:     Carry is clear if UMBs are available, or set if they are not
38032                                 ; ERROR:    None
38033                                 ; USES:     CF (AX,BX,DS,ES pushed 'cause they're used by others)
38034                                 ; ------------------------------------------------------------------------------
38035
38036                                     ; 01/01/2023 - Retro DOS v4.2 (Modified MSDOS 6.21 IO.SYS)
38037                                 UmbTest:
38038                                     ; 01/01/2023
38039                                     ;push   ax
```

```
0000347D 53                             push    bx ; *
                                        ;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 ?

00003491 B452                           mov     ah,GET_IN_VARS          ; Call int 21h, function 52h...
00003493 CD21                           int     21h

00003495 268B47FE                       mov     ax,[es:bx-2]
                                        ; 22/07/2023
                                um10:
00003499 8EC0                           mov     es,ax ; * ; **

                                ; ------------------------------
                                ; UM10: ES = Current MCB pointer
                                ; ------------------------------

                                ;um10:
                                    ;mov    al,[es:ARENA.SIGNATURE]
                                    ;cmp    al,arena_signature_end ; 'Z'
                                    ; 22/07/2023
0000349B 26803E00005A                   cmp     byte [es:ARENA.SIGNATURE],arena_signature_end ; 'Z'
000034A1 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 ; *
000034A3 2603060300                     add     ax,[es:ARENA.SIZE]
000034A8 40                             inc     ax
                                        ;mov    es,ax ; **

000034A9 EBEE                           jmp     short um10              ; And restart the loop.
                                um20:
                                    ; 22/07/2023
                                    ; ax = es
                                    ;mov    ax,es

000034AB 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 ; ?

000034AE 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:
000034AF 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]
000034B1 8A1E[3024]                     mov     bl,[fm_umb]

000034B5 B80358                         mov     ax,DOS_SET_UMBLINK ; 5803h
000034B8 CD21                           int     21h
000034BA 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:
                                ;
                                ;     DATE    WHO      DESCRIPTION
                                ;   --------  -------  -------------------------------------------------------
                                ;   05/14/92  t-richj  Original
                                ;   06/21/92  t-richj  Final revisions before check-in
                                
                                UMB_HeadIdx equ    8Ch     ; Offset from ES (after func52h) to get UMBHead
                                
                                ; --------------------------------------------------------------------------
                                ;*** UnHideUMBs - Marks HIDDEN elements as FREE
                                ; --------------------------------------------------------------------------
                                ; ENTRY:  None; perhaps, earlier, HideUMBs was called... if not, we have
                                ;              very little to do, as no elelments will be marked as HIDDEN.
                                ; EXIT:   Sets InHigh to zero; carry clear if HideUMBs was called earlier.
                                ; ERROR:  None
                                ; USES:   fInHigh (from highvar.inc), carry flag
                                ; --------------------------------------------------------------------------
                                
                                    ; 31/12/2022 - Retro DOS v4.2 (Modified MSDOS 6.21 IO.SYS)
                                    ; (SYSINIT:357Bh)
                                
                                UnHideUMBs:
000034B7 50                         push    ax              ; Save ax for what we're about to do
                                
                                ; --------------------------------------------------------------------------
                                ; BUGBUG t-richj 11-8-92: The following six lines were commented out for a good
                                ;    length of time. Those six constitute a check of whether or not we should
                                ;    indeed clean up the upper-memory chain; without such a check, COMMAND.COM
                                ;    will destroy the current link-state and memory-allocation strategy after
                                ;    every command execution.
                                ; --------------------------------------------------------------------------
                                
                                    ;getdata al,fInHigh   ; Get InHigh from data segment
                                    ;
                                    ;push  ds
                                    ;push  cs
                                    ;pop   ds
                                    ;mov   al,[fInHigh]
                                    ;pop   ds
                                
                                    ;mov   al,[cs:fInHigh]
                                    ; 31/12/2022
                                    ; ds = cs
000034B8 A0[FB23]                   mov     al,[fInHigh]
                                
000034BB 08C0                       or      al,al
000034BD 7503                       jnz     short uhu10     ; If didn't call loadhigh/devicehigh earlier,
                                
000034BF 58                         pop     ax              ; then there's nothing to do here... so
000034C0 F9                         stc                     ; restore everything and return. Just like
000034C1 C3                         retn                    ; that.
                                uhu10:
000034C2 E88E00                     call    linkumb         ; Make sure UMBs are linked in.
000034C5 E81200                     call    FreeUMBs
                                
                                    ;putdata fInHigh,0    ; We're leaving, so update fInHigh.
                                    ;
                                    ;push  es
                                    ;push  cs
                                    ;pop   es
                                    ;mov   byte [es:fInHigh],0
                                    ;pop   ds
                                
                                    ; 31/12/2022
                                    ; ds = cs
                                    ;mov   byte [cs:fInHigh],0
000034C8 C606[FB23]00               mov     byte [fInHigh],0
                                
                                    ;call  he_unlink      ; Unlink UMBs
                                    ; 31/12/2022
                                ;;he_unlink:
000034CD 30FF                       xor     bh,bh
                                
                                    ;getdata bl,fm_umb    ; Restore original link-state
                                    ;mov   bl,[cs:fm_umb]
000034CF 8A1E[3024]                 mov     bl,[fm_umb]
                                
000034D3 B80358                     mov     ax,DOS_SET_UMBLINK ; 5803h
000034D6 CD21                       int     21h
                                    ;;retn
                                
000034D8 58                         pop     ax
                                    ; 12/12/2022
                                    ;clc    ; 12/12/2022 (this clc may not be necessary!?)
000034D9 C3                         retn
                                
                                ; 31/12/2022
                                ;%if 0
                                ;
                                ;; --------------------------------------------------------------------------
                                ;;*** he_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
                                ;; --------------------------------------------------------------------------
                                ;
                                ;he_unlink:
                                ;   xor     bh, bh
                                ;
                                ;   ;getdata bl, fm_umb   ; Restore original link-state
                                ;   mov     bl,[cs:fm_umb]
                                ;
                                ;   mov     ax,DOS_SET_UMBLINK ; 5803h
                                ;   int     21h
                                ;   retn
                                ;
                                ;%endif
                                
                                ; --------------------------------------------------------------------------
                                ;*** freeUMBs - frees all HIDDEN memory elements in upper-memory.
                                ; --------------------------------------------------------------------------
                                ; ENTRY:    None
                                ; EXIT:     None; HIDDEN memory elements returned to FREE
                                ; ERROR:    None (ignore CF)
                                ; USES:     Flags
                                ; --------------------------------------------------------------------------
                                
                                FreeUMBs:
000034DA 50                         push    ax
000034DB 06                         push    es
```

```
000034DC E86700                     call    HeadUmb         ; Returns with carry if err, else ES == MCB
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
000035A0 C606[5124]00               mov     byte [DeviceHi],0
                                initforlo_x:
                                    ; 11/12/2022
                                    ; ds = cs
                                                                  ;  into UMB indicate that
                                                                  ;  we are loading low
                                    ;mov    ax,[cs:memhi]         ; AX = start of Low memory
                                    ;mov    dx,[cs:ALLOCLIM]      ; DX = End of Low memory
000035A5 A1[6403]                   mov     ax,[memhi]
000035A8 8B16[A502]                 mov     dx,[ALLOCLIM]
                                idl1:
000035AC E86600                     call    DevSetMark            ; setup a sub-arena for DD
                                    ; 11/12/2022
                                    ; ds = cs
                                    ;mov    [cs:DevLoadAddr],ax   ; init the Device load address
                                    ;mov    [cs:DevLoadEnd],dx    ; init the limit of the block
                                    ;mov    word [cs:DevEntry],0  ; init Entry point to DD
                                    ;mov    [cs:DevEntry+2],ax
000035AF A3[3524]                   mov     [DevLoadAddr],ax
000035B2 8916[3724]                 mov     [DevLoadEnd],dx
000035B6 C706[3924]0000             mov     word [DevEntry],0
000035BC A3[3B24]                   mov     [DevEntry+2],ax
                                    ; 01/01/2023
                                    ;pop    es ; *
000035BF C3                         retn

                                ;--------------------------------------------------------------------------
                                ;
                                ; procedure : SpaceInUMB?
                                ;
                                ;   Input : DevUMBAddr, DevUMBSize, DevUMBFree & DevSize
                                ;   Output : Carry set if no space in UMB
                                ;            Carry clear if Space is available for the device in
                                ;              current UMB
                                ;
                                ;--------------------------------------------------------------------------

                                SpaceInUMB:
                                    ; 11/12/2022
                                    ; ds = cs
                                    ;mov    ax,[cs:DevUMBSize]
                                    ;add    ax,[cs:DevUMBAddr]    ; End of UMB
                                    ;sub    ax,[cs:DevUMBFree]    ; - Free = Remaining space
000035C0 A1[4524]                   mov     ax,[DevUMBSize]
000035C3 0306[4324]                 add     ax,[DevUMBAddr]             ; End of UMB
000035C7 2B06[4724]                 sub     ax,[DevUMBFree]             ; - Free = Remaining space
                                    ; 11/12/2022
                                    ;or     ax,ax                 ; Nospace ?
                                    ;jnz    short spcinumb1
                                    ;stc
                                    ;retn
                                    ; 11/12/2022
000035CB 83F801                     cmp     ax,1
000035CE 7205                       jb      short spcinumb2       ; cf=1
                                spcinumb1:
000035D0 48                         dec     ax                    ; space for sub-arena
                                    ; 11/12/2022
                                    ; ds = cs
000035D1 3B06[3324]                 cmp     ax,[DevSize]
                                    ;cmp    ax,[cs:DevSize]             ; do we have space ?
                                spcinumb2:
000035D5 C3                         retn

                                ;--------------------------------------------------------------------------
                                ;
                                ; procedure : PrepareMark
                                ;
                                ;   Input : AX==Address of MCB (not addr of free space), BX==Size
                                ;   Output : None; MCB marked appropriately and DevUMB* set as needed.
                                ;
                                ;--------------------------------------------------------------------------

                                ;   ; 01/11/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
                                ;
                                ;PrepareMark:
                                ;   push    ds
                                ;   mov     ds,ax
                                ;   mov     word [ARENA.OWNER],8
                                ;   mov     word [ARENA.NAME],'SD' ; 4453h
                                ;   pop     ds
                                ;
                                ;   inc     ax
                                ;   mov     [cs:DevUMBAddr],ax
                                ;   mov     [cs:DevUMBFree],ax
                                ;   mov     [cs:DevUMBSize],bx    ; update the UMB Variables
                                ;   retn

                                ;--------------------------------------------------------------------------
                                ;
                                ; procedure : GetUMBForDev
                                ;
                                ;   Input : DevSize
                                ;   Output : Carry set if couldn't allocate a UMB to fit the
                                ;            the device.
                                ;            If success carry clear
                                ;
                                ;   Allocates the biggest UMB for loading devices and updates
                                ;   DevUMBSize, DevUMBAddr & DevUMBFree if it succeeded in allocating
                                ;   UMB.
                                ;
                                ;   This routine relies on the fact that all of the low memory
                                ;   is allocated, and any DOS alloc calls should return memory
                                ;   from the UMB pool.
                                ;
                                ;--------------------------------------------------------------------------
                                    ; 01/11/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 IO.SYS)
                                    ; (SYSINIT:2BC6h)

                                GetUMBForDev:
                                    ; 11/12/2022
                                    ; ds = cs
000035D6 BBFFFF                     mov     bx,0FFFFh
000035D9 B80048                     mov     ax,4800h
000035DC CD21                       int     21h
                                            ; DOS - 2+ - ALLOCATE MEMORY
                                            ; BX = number of 16-byte paragraphs desired

000035DE 09DB                       or      bx,bx
                                    ;jz     short gufd_err
                                    ; 09/09/2023
000035E0 742E                       jz      short gufd_error ; bx = 0
```

```
000035E2 4B                             dec     bx
                                        ; 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

```

```
38784 00003644 BF0800                   mov     di,devmark.filename ; 8
38785 00003647 B90800                   mov     cx,8                ; maximum 8 characters
38786                                dsm_next_char:
38787 0000364A AC                       lodsb
38788 0000364B 08C0                     or      al, al
38789 0000364D 7407                     jz      short blankout
38790 0000364F 3C2E                     cmp     al, '.'
38791 00003651 7403                     je      short blankout
38792 00003653 AA                       stosb
38793 00003654 E2F4                     loop    dsm_next_char
38794                                blankout:
38795 00003656 E304                     jcxz    dsm_exit
38796 00003658 B020                     mov     al, ' '
38797 0000365A F3AA                     rep     stosb               ; blank out the rest
38798                                dsm_exit:
38799 0000365C 58                       pop     ax                  ; restore load addr
38800 0000365D 5E                       pop     si
38801 0000365E 1F                       pop     ds
38802                                   ; 03/01/2023
38803                                   ;pop    di
38804 0000365F 07                       pop     es
38805 00003660 C3                       retn
38806
38807                                ;--------------------------------------------------------------------------
38808                                ;
38809                                ; procedure : SizeDevice
38810                                ;
38811                                ;   Input : ES:SI - points to device file to be sized
38812                                ;
38813                                ;   Output : Carry set if file cannot be opened or if it is an OS2EXE file
38814                                ;
38815                                ;   Calculates the size of the device file in paras and stores it
38816                                ;   in DevSize
38817                                ;
38818                                ;--------------------------------------------------------------------------
38819
38820                                   ; 01/11/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 IO.SYS)
38821                                SizeDevice:
38822                                   ; 01/11/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
38823                                   ; 11/12/2022 ; *
38824 00003661 1E                      push    ds ; *
38825 00003662 06                      push    es
38826 00003663 1F                      pop     ds
38827 00003664 89F2                    mov     dx,si               ; ds:dx -> file name
38828 00003666 B8003D                  mov     ax,3D00h            ; open
38829 00003669 CD21                    int     21h
38830 0000366B 7237                    jc      short sd_err        ; open failed
38831
38832 0000366D 89C3                    mov     bx,ax               ; BX - file handle
38833 0000366F B80242                  mov     ax,4202h            ; seek
38834 00003672 31C9                    xor     cx,cx
38835 00003674 89CA                    mov     dx,cx               ; to end of file
38836 00003676 CD21                    int     21h
38837 00003678 7223                    jc      short sd_close      ; did seek fail (impossible)
38838 0000367A 83C00F                  add     ax,15               ; para convert
38839 0000367D 83D200                  adc     dx,0
38840 00003680 F7C2F0FF                test    dx,0FFF0h           ; size > 0ffffh paras ?
38841                                  ;jz     short szdev1        ; no
38842                                  ; 22/07/2023
38843 00003684 7409                    jz      short sd_ctp
38844 00003686 2EC706[3324]FFFF        mov     word [cs:DevSize],0FFFFh ; invalid device size
38845                                                              ; assuming that we fail later
38846 0000368D EB0E                    jmp     short sd_close
38847                                sd_ctp:
38848                                   ; 22/07/2023
38849                                ;szdev1:
38850 0000368F B104                    mov     cl,4                ; convert it to paras
38851 00003691 D3E8                    shr     ax,cl
38852 00003693 B10C                    mov     cl,12
38853 00003695 D3E2                    shl     dx,cl
38854 00003697 09D0                    or      ax,dx ; * ; cf=0
38855                                  ;
38856                                  ; 22/07/2023 - Retro DOS v4.2 IO.SYS (optimized)
38857                                  ; MSDOS 6.21 IO.SYS - SYSINIT:37A6h
38858                                  ; 01/11/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
38859                                  ;cmp    ax,[cs:DevSizeOption]
38860                                  ;ja     short szdev2
38861                                  ;mov    ax,[cs:DevSizeOption]
38862                                  ; 12/12/2022
38863                                  ;clc
38864                                ;szdev2:
38865 00003699 2EA3[3324]              mov     [cs:DevSize],ax             ; save file size (in paragraps)
38866                                  ; 22/07/2023
38867                                  ;clc ; cf=0 ; *       ; CLC is not needed here
38868                                                        ; (OR instruction clears CF) - E.TAN 22/07/2023
38869
38870                                  ; 01/11/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
38871                                  ; 12/12/2022
38872                                  ; cf=0
38873                                  ;clc
38874                                sd_close:
38875 0000369D 9C                      pushf                       ; let close not spoil our
38876                                                              ;  carry flag
38877 0000369E B8003E                  mov     ax,3E00h            ; close
38878 000036A1 CD21                    int     21h                 ; we are not checking for err
38879 000036A3 9D                      popf
38880                                sd_err:
38881                                   ; 01/11/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
38882                                   ; 11/12/2022 ; *
38883 000036A4 1F                      pop     ds ; *
38884 000036A5 C3                      retn
38885
38886                                ;--------------------------------------------------------------------------
38887                                ;
38888                                ; procedure : ExecDev
38889                                ;
38890                                ;   Input : ds:dx -> device to be executed
38891                                ;           DevLoadAddr - contains where device has to be loaded
38892                                ;
38893                                ;   Output : Carry if error
38894                                ;            Carry clear if no error
38895                                ;
38896                                ;   Loads a device driver using the 4b03h function call
38897                                ;
38898                                ;--------------------------------------------------------------------------
38899
38900                                   ; 01/11/2022
38901                                ExecDev:
38902 000036A6 2E8B1E[3524]            mov     bx,[cs:DevLoadAddr]
38903 000036AB 2E891E[4D24]            mov     [cs:DevExecAddr],bx ; Load the parameter block
38904                                                              ;  block for exec with
38905                                                              ;  load address
38906 000036B0 2E891E[4F24]            mov     [cs:DevExecReloc],bx
38907 000036B5 8CCB                    mov     bx,cs
```

```
38908 000036B7 8EC3                     mov     es,bx
38909 000036B9 BB[4D24]                 mov     bx,DevExecAddr          ; es:bx points to parameters
38910                                   ;mov    al,3    ; (load program only)
38911                                   ;mov    ah,EXEC ; 4Bh
38912                                   ; 04/07/2023
38913 000036BC B8034B                   mov     ax,(EXEC<<8)|03h
38914 000036BF CD21                     int     21h                     ; load in the device driver
38915                                           ; DOS - 2+ - LOAD OR EXECUTE (EXEC)
38916                                           ; DS:DX -> ASCIZ filename
38917                                           ; ES:BX -> parameter block
38918                                           ; AL = subfunction
38919 000036C1 C3                       retn
38920
38921                                ;--------------------------------------------------------------------------
38922                                ;
38923                                ; procedure : RetFromUM
38924                                ;
38925                                ;   Input : None
38926                                ;   Output : ConvLoad set if didn't previously call HideUMBs
38927                                ;            ConvLoad clear if did.
38928                                ;
38929                                ;   Prepares memory for more devices after returning from loading one
38930                                ;   using the DOS 6 options (/L:... etc).
38931                                ;
38932                                ;--------------------------------------------------------------------------
38933
38934                                ; 31/12/2022 - Retro DOS v4.2 (Modified MSDOS 6.21 IO.SYS)
38935                                ;  (SYSINIT:37D1h)
38936
38937                                ; 01/11/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
38938                                ;%if 0
38939                                RetFromUM:
38940                                    ; 31/12/2022
38941                                    ; ds = cs
38942 000036C2 9C                       pushf
38943                                    ;mov    byte [cs:ConvLoad],1
38944 000036C3 C606[4124]01             mov     byte [ConvLoad],1
38945 000036C8 E8ECFD                   call    UnHideUMBs
38946 000036CB 7204                     jc      short rfUM1             ; Skip this if didn't HideUMBs
38947                                    ; 31/12/2022
38948                                    ; ds = cs
38949                                    ;;mov   byte [cs:ConvLoad],0
38950                                    ;mov    byte [ConvLoad],0
38951                                    ; 09/09/2023
38952 000036CD FE0E[4124]               dec     byte [ConvLoad] ; -> 0
38953                                rfUM1:
38954 000036D1 9D                       popf
38955 000036D2 C3                       retn
38956
38957                                ;%endif ; 01/11/2022
38958
38959                                ;--------------------------------------------------------------------------
38960                                ;
38961                                ; procedure : RemoveNull
38962                                ;
38963                                ;   Input : ES:SI points to a null terminated string
38964                                ;
38965                                ;   Output : none
38966                                ;
38967                                ;   Replaces the null at the end of a string with blank
38968                                ;
38969                                ;--------------------------------------------------------------------------
38970
38971                                    ; 01/11/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 IO.SYS)
38972                                    ; (SYSINIT:2CCEh)
38973                                RemoveNull:
38974                                    ; 11/12/2022
38975                                    ; ds = cs
38976                                rn_next:
38977 000036D3 268A1C                   mov     bl,[es:si]
38978 000036D6 08DB                     or      bl,bl                   ; null ?
38979 000036D8 7403                     jz      short rn_gotnull
38980 000036DA 46                       inc     si                      ; advance the pointer
38981 000036DB EBF6                     jmp     short rn_next
38982                                rn_gotnull:
38983                                    ; 11/12/2022
38984 000036DD 8A1E[6624]               mov     bl,[DevSavedDelim]
38985                                    ;mov    bl,[cs:DevSavedDelim]
38986 000036E1 26881C                   mov     [es:si],bl              ; replace null with blank
38987                                    ; 02/11/2022
38988                                ; 11/12/2022
38989                                rba_ok:             ; 10/04/2019
38990 000036E4 C3                       retn
38991
38992                                ;--------------------------------------------------------------------------
38993                                ;
38994                                ; procedure : RoundBreakAddr
38995                                ;
38996                                ;   Input : DevBrkAddr
38997                                ;   Output : DevBrkAddr
38998                                ;
38999                                ;   Rounds DevBrkAddr to a para address so that it is of the form xxxx:0
39000                                ;
39001                                ;--------------------------------------------------------------------------
39002
39003                                RoundBreakAddr:
39004 000036E5 2EA1[3D24]               mov     ax,[cs:DevBrkAddr]
39005 000036E9 E829DC                   call    ParaRound
39006 000036EC 2E0106[3F24]             add     [cs:DevBrkAddr+2],ax
39007 000036F1 2EC706[3D24]0000         mov     word [cs:DevBrkAddr],0
39008 000036F8 2EA1[3724]               mov     ax,[cs:DevLoadEnd]
39009 000036FC 2E3906[3F24]             cmp     [cs:DevBrkAddr+2],ax
39010 00003701 76E1                     jbe     short rba_ok
39011 00003703 E92911                   jmp     mem_err
39012                                    ; 02/11/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
39013                                    ; 11/12/2022
39014                                ;rba_ok:
39015                                ;   retn
39016
39017                                ;--------------------------------------------------------------------------
39018                                ;
39019                                ; procedure : DevSetBreak
39020                                ;
39021                                ;   Input : DevBrkAddr
39022                                ;   Output : Carry set if Device returned Init failed
39023                                ;            Else carry clear
39024                                ;
39025                                ;--------------------------------------------------------------------------
39026
39027                                DevSetBreak:
39028 00003706 50                       push    ax
39029
39030 00003707 2EA1[3F24]                mov     ax,[cs:DevBrkAddr+2]   ;remove the init code
39031 0000370B 2E803E[6619]00           cmp     byte [cs:multdeviceflag],0
```

```
39032 00003711 750F                     jne     short set_break_continue ;do not check it.
39033 00003713 2E3B06[3524]             cmp     ax,[cs:DevLoadAddr]
39034 00003718 7508                     jne     short set_break_continue ;if not same, then o.k.
39035
39036                                   ;cmp    word [cs:DevBrkAddr],0
39037                                   ;je     short break_failed    ;[DevBrkAddr+2]=[memhi] & [DevBrkAddr]=0
39038                                   ; 12/12/2022
39039 0000371A 2E833E[3D24]01           cmp     word [cs:DevBrkAddr],1
39040 00003720 7204                     jb      short break_failed
39041
39042                                 set_break_continue:
39043 00003722 E8C0FF                   call    RoundBreakAddr
39044                                   ; 12/12/2022
39045 00003725 F8                       clc
39046                                 break_failed:
39047 00003726 58                       pop     ax
39048                                   ;clc
39049 00003727 C3                       retn
39050
39051                                   ; 12/12/2022
39052                                 ;break_failed:
39053                                   ;pop    ax
39054                                   ;stc
39055                                   ;retn
39056
39057                                 ;--------------------------------------------------------------------------
39058                                 ;
39059                                 ; procedure : DevBreak
39060                                 ;
39061                                 ;   Input : DevLoadAddr & DevBrkAddr
39062                                 ;   Output : none
39063                                 ;
39064                                 ;   Marks a succesful install of a device driver
39065                                 ;   Sets device size field in sub-arena &
39066                                 ;   Updates Free ptr in UMB or adjusts memhi
39067                                 ;
39068                                 ;--------------------------------------------------------------------------
39069
39070                                   ; 11/12/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 IO.SYS)
39071                                 DevBreak:
39072                                   ;push  ds ; 11/12/2022
39073
39074                                   ; 11/12/2022
39075 00003728 0E                       push    cs
39076 00003729 1F                       pop     ds
39077                                   ;mov    ax,[cs:DevLoadAddr]
39078                                   ;mov    bx,[cs:DevBrkAddr+2]
39079 0000372A A1[3524]                 mov     ax,[DevLoadAddr]
39080 0000372D 8B1E[3F24]               mov     bx,[DevBrkAddr+2]
39081                                   ; 11/12/2022
39082 00003731 1E                       push    ds
39083
39084 00003732 48                       dec     ax                      ; seg of sub-arena
39085 00003733 8ED8                     mov     ds,ax
39086 00003735 40                       inc     ax                      ; Back to Device segment
39087 00003736 29D8                     sub     ax,bx
39088 00003738 F7D8                     neg     ax                      ; size of device in paras
39089 0000373A A33000                   mov     [devmark.size],ax       ; store it in sub-arena
39090
39091                                   ; 11/12/2022
39092 0000373D 1F                       pop     ds
39093                                   ; ds = cs
39094
39095 0000373E 803E[5124]00             cmp     byte [DeviceHi],0
39096                                   ;cmp    byte [cs:DeviceHi],0
39097 00003743 7405                     je      short db_lo
39098                                   ;mov    [cs:DevUMBFree],bx      ; update Free ptr in UMB
39099                                   ;jmp    short db_exit
39100                                   ; 11/12/2022
39101 00003745 891E[4724]               mov     [DevUMBFree],bx
39102 00003749 C3                       retn
39103                                 db_lo:
39104                                   ; 11/12/2022
39105                                   ; ds = cs
39106                                   ;mov    [cs:memhi],bx
39107                                   ;mov    word [cs:memlo],0
39108 0000374A 891E[6403]               mov     [memhi],bx
39109 0000374E C706[6203]0000           mov     word [memlo],0 ; 18/12/2022
39110                                 db_exit:
39111                                   ;pop    ds ; 11/12/2022
39112                                 sd_ret:             ; 09/09/2023
39113 00003754 C3                       retn
39114
39115                                 ; 10/04/2019 - Retro DOS v4.0
39116
39117                                 ;--------------------------------------------------------------------------
39118                                 ;
39119                                 ; procedure : ParseSize
39120                                 ;
39121                                 ;   Parses the command line for SIZE= command
39122                                 ;
39123                                 ;   ES:SI = command line to parsed
39124                                 ;
39125                                 ;   returns ptr to command line after SIZE= option in ES:SI
39126                                 ;   updates the DevSizeOption variable with value supplied
39127                                 ;   in SIZE=option
39128                                 ;   Returns carry if the SIZE option was invalid
39129                                 ;
39130                                 ;--------------------------------------------------------------------------
39131
39132                                   ; 02/11/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 IO.SYS)
39133                                   ; (SYSINIT:2D5Ah)
39134
39135                                   ; 09/09/2023 - Retro DOS v4.2 IO.SYS (Optimization) ((&BugFix))
39136                                   ; (MSDOS 6.21 IO.SYS - SYSINIT:3871h) - Retro DOS v4.2 -
39137                                   ; (PCDOS 7.1 IO.SYS - SYSINIT:3D6Eh)  - Retro DOS v5.0 -
39138                                 ParseSize:
39139                                   ;push   bx
39140                                   ;mov    bx,si
39141
39142                                   ; 09/09/2023
39143 00003755 56                       push    si ; * ; mov bx,si
39144
39145                                   ; 11/12/2022
39146                                   ; ds = cs
39147                                   ;mov    word [cs:DevSizeOption],0 ; init the value
39148                                   ;mov    [cs:DevCmdLine],si
39149                                   ;mov    [cs:DevCmdLine+2],es
39150 00003756 C706[5224]0000           mov     word [DevSizeOption],0 ; init the value
39151 0000375C 8936[6224]               mov     [DevCmdLine],si
39152 00003760 8C06[6424]               mov     [DevCmdLine+2],es
39153 00003764 E82400                   call    SkipDelim
39154 00003767 26813C5349               cmp     word [es:si],'SI' ; 4953h
39155 0000376C 7528                     jne     short ps_no_size
```

```
39156 0000376E 26817C025A45               cmp     word [es:si+2],'ZE' ; 455Ah
39157 00003774 7520                       jne     short ps_no_size
39158 00003776 268A4404                   mov     al,[es:si+4]
39159 0000377A E80D10                     call    delim
39160                                     ;jne    short ps_no_size
39161                                     ; 22/07/2023
39162 0000377D 7518                       jne     short ps_no_size_2 ; cf=0 here
39163 0000377F 83C605                     add     si,5
39164 00003782 E81400                     call    GetHexNum
39165 00003785 7210                       jc      short ps_err
39166                                     ; 11/12/2022
39167                                     ; ds = cs
39168                                     ;mov    [cs:DevSizeOption],ax
39169 00003787 A3[5224]                   mov     [DevSizeOption],ax
39170
39171                                     ; 09/09/2023
39172 0000378A 58                         pop     ax  ; * (discard previous si value on top of stack)
39173
39174                                   ;   call    SkipDelim ; **
39175                                   ;
39176                                   ;   ; 22/07/2023
39177                                   ;;ps_no_size_2:
39178                                   ;   ; cf = 0
39179                                   ;   retn
39180
39181                                       ; 09/09/2023
39182                                       ;jmp    short SkipDelim
39183
39184                                   ; 09/09/2023 - Retro DOS v4.2 IO.SYS (Optimization)
39185                                   %if 1
39186                                       ; 01/11/2022
39187                                   SkipDelim:
39188                                   sd_next_char:
39189 0000378B 268A04                     mov     al,[es:si]
39190 0000378E E8F90F                     call    delim
39191 00003791 75C1                       jnz     short sd_ret ; cf=0 ; 09/09/2023
39192 00003793 46                         inc     si
39193 00003794 EBF5                       jmp     short sd_next_char ; 01/11/2022
39194                                     ; 11/12/2022
39195                                     ; 01/11/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
39196                                   ;sd_ret:
39197                                       ;retn
39198                                   %endif
39199
39200                                       ;;;call SkipDelim ; **
39201                                       ; 02/11/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
39202                                       ;mov    bx,si
39203                                   ps_no_size:
39204                                       ;mov    si,bx
39205                                       ;pop    bx
39206 00003796 F8                         clc     ; cf=0
39207                                     ;retn
39208                                     ; 11/12/2022
39209                                   ps_err:             ; cf=1
39210                                   ps_no_size_2:       ; 09/09/2023 (cf=0)
39211                                       ; 09/09/2023
39212 00003797 5E                         pop     si ; * ; mov si,bx
39213                                   ;sd_ret:    ; cf=?
39214 00003798 C3                         retn
39215
39216                                   ;ps_err:
39217                                       ; 02/11/2022
39218                                       ;pop    bx
39219                                       ;stc
39220                                       ; 01/11/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
39221                                       ; 11/12/2022
39222                                       ; cf=1
39223                                       ;stc
39224                                       ; 11/12/2022
39225                                   ;sd_ret:
39226                                       ; 22/07/2023
39227                                       ; 12/04/2019
39228                                       ;retn
39229
39230                                   ; 12/04/2019 - Retro DOS v4.0
39231
39232                                   ;--------------------------------------------------------------------------
39233                                   ;
39234                                   ; procedure : SkipDelim
39235                                   ;
39236                                   ;   Skips delimiters in the string pointed to by ES:SI
39237                                   ;   Returns ptr to first non-delimiter character in ES:SI
39238                                   ;
39239                                   ;--------------------------------------------------------------------------
39240
39241                                   ; 09/09/2023 - Retro DOS v4.2 IO.SYS (Optimization)
39242                                   %if 0
39243                                       ; 01/11/2022
39244                                   SkipDelim:
39245                                   sd_next_char:
39246                                       mov     al,[es:si]
39247                                       call    delim
39248                                       jnz     short sd_ret
39249                                       inc     si
39250                                       jmp     short sd_next_char ; 01/11/2022
39251                                       ; 11/12/2022
39252                                       ; 01/11/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
39253                                   ;sd_ret:
39254                                       ;retn
39255                                   %endif
39256
39257                                   ;--------------------------------------------------------------------------
39258                                   ;
39259                                   ; procedure : GetHexNum
39260                                   ;
39261                                   ;   Converts an ascii string terminated by a delimiter into binary.
39262                                   ;   Assumes that the ES:SI points to a Hexadecimal string
39263                                   ;
39264                                   ;   Returns in AX the number number of paras equivalent to the
39265                                   ;   hex number of bytes specified by the hexadecimal string.
39266                                   ;
39267                                   ;   Returns carry in case it encountered a non-hex character or
39268                                   ;   if it encountered crlf
39269                                   ;
39270                                   ;--------------------------------------------------------------------------
39271
39272                                   ; 13/05/2019
39273
39274                                       ; 03/01/2023 - Retro DOS v4.2 (Modified MSDOS 6.21 IO.SYS)
39275                                       ; (SYSINIT:38C5h)
39276
39277                                       ; 01/11/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 IO.SYS)
39278                                       ; (SYSINIT:2DA5h)
39279                                   GetHexNum:
```

```
00003799 31C0                   xor     ax,ax
0000379B 31D2                   xor     dx,dx
                        ghn_next:
0000379D 268A1C                 mov     bl,[es:si]
000037A0 80FB0D                 cmp     bl,cr  ; 0Dh
000037A3 7436                   je      short ghn_err
000037A5 80FB0A                 cmp     bl,lf  ; 0Ah
000037A8 7431                   je      short ghn_err
000037AA 50                     push    ax
000037AB 88D8                   mov     al,bl
000037AD E8DA0F                 call    delim
000037B0 58                     pop     ax
                                ; 03/01/2023
000037B1 B90400                 mov     cx,4
000037B4 7410                   jz      short ghn_into_paras
000037B6 E82400                 call    GetNibble
                                ;jc     short ghn_err
                                ; 11/12/2022
000037B9 7221                   jc      short ghn_ret ; cf=1
                                ; 03/01/2023
                                ;mov    cx,4
                        ghn_shift1:
000037BB D1E0                   shl     ax,1
000037BD D1D2                   rcl     dx,1
000037BF E2FA                   loop    ghn_shift1
000037C1 08D8                   or      al,bl
000037C3 46                     inc     si
000037C4 EBD7                   jmp     short ghn_next
                        ghn_into_paras:
000037C6 83C00F                 add     ax,15
000037C9 83D200                 adc     dx,0
000037CC F7C2F0FF               test    dx,0FFF0h
000037D0 7509                   jnz     short ghn_err
                                ; 03/01/2023
                                ;mov    cx,4
                        ghn_shift2:
000037D2 F8                     clc
000037D3 D1DA                   rcr     dx,1
000037D5 D1D8                   rcr     ax,1
000037D7 E2F9                   loop    ghn_shift2
000037D9 F8                     clc
000037DA C3                     retn
                                ; 11/12/2022
                        ghn_err:
                        gnib_err:
000037DB F9                     stc
                        ghn_ret:
                        gnib_ret:
000037DC C3                     retn

                        ;--------------------------------------------------------------------------
                        ;
                        ; procedure : GetNibble
                        ;
                        ;   Convert one nibble (hex digit) in BL into binary
                        ;
                        ;   Returns binary value in BL
                        ;
                        ;   Returns carry if BL contains non-hex digit
                        ;
                        ;--------------------------------------------------------------------------

                        GetNibble:
000037DD 80FB30                 cmp     bl,'0'
                                ;jb     short gnib_err
                                ; 11/12/2022
000037E0 72FA                   jb      short gnib_ret ; cf=1
000037E2 80FB39                 cmp     bl,'9'
000037E5 7704                   ja      short is_it_hex
000037E7 80EB30                 sub     bl,'0'          ; clc
000037EA C3                     retn
                        is_it_hex:
000037EB 80FB41                 cmp     bl,'A'
                                ;jb     short gnib_err
                                ; 11/12/2022
000037EE 72EC                   jb      short gnib_ret ; cf=1
000037F0 80FB46                 cmp     bl,'F'
000037F3 77E6                   ja      short gnib_err ; 11/12/2022
000037F5 80EB37                 sub     bl,'A'- 10     ; clc
000037F8 C3                     retn

                                ; 11/12/2022
                        ;gnib_err:
                        ;       stc
                        ;gnib_ret:
                        ;       retn

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

                        ; 12/04/2019 - Retro DOS v4.0

                        ; umb.inc (MSDOS 6.0, 1991)
                        DOS_ARENA   equ 24h         ; offset of arena_head var in DOS data segm.
                        UMB_ARENA   equ 8Ch         ; offset of umb_head in DOS data

                        XMM_REQUEST_UMB    equ 10h
                        XMM_RELEASE_UMB    equ 11h

                        ; 09/09/2023 - Retro DOS v4.2 IO.SYS (Optimization)

                        ;--------------------------------------------------------------------------
                        ;
                        ; Procedure Name   : umb_insert
                        ;
                        ; Inputs           : DOSDATA:UMB_HEAD = start of umb chain
                        ;                  : BX = seg address of UMB to be linked in
                        ;                  : DX = size of UMB to be linked in paras
                        ;                  ; DS = data
                        ;
                        ; Outputs          : links the UMB into the arena chain
                        ;
                        ; Uses                    : AX, CX, ES, DX, BX
                        ;
                        ;--------------------------------------------------------------------------

                        umb_insert:
000037F9 1E                     push    ds

                                ; 31/12/2022
                                ; ds = cs

                                ;mov    ds,[cs:DevDOSData]
000037FA 8E1E[6024]             mov     ds,[DevDOSData] ; 31/12/2022
                                ;mov    ds,[8Ch]
```

```
39404 000037FE 8E1E8C00                 mov     ds,[UMB_ARENA]          ; es = UMB_HEAD
39405 00003802 8CD8                     mov     ax,ds
39406 00003804 8EC0                     mov     es,ax
39407                                   ui_next:
39408 00003806 39D8                     cmp     ax,bx                   ; Q: is current block above
39409                                                                   ;    new block
39410 00003808 770F                     ja      short ui_insert                 ; Y: insert it
39411                                                                   ; Q: is current block the
39412                                                                   ;    last
39413 0000380A 26803E00005A             cmp     byte [es:ARENA.SIGNATURE],arena_signature_end ; 'Z'
39414 00003810 745C                     je      short ui_append                 ; Y: append new block to chain
39415                                                                   ; N: get next block
39416 00003812 8ED8                     mov     ds,ax                   ; M005
39417                                   ;call   get_next                ; ax = es = next block
39418 00003814 E83B01                   call    _get_next_ ; 13/04/2019 - Retro DOS v4.0
39419 00003817 EBED                     jmp     short ui_next
39420
39421                                   ui_insert:
39422 00003819 8CD9                     mov     cx,ds                   ; ds = previous arena
39423 0000381B 41                       inc     cx                      ; top of previous block
39424
39425 0000381C 29D9                     sub     cx,bx
39426 0000381E F7D9                     neg     cx                      ; cx = size of used block
39427                                   ;mov    byte [0],'M'
39428 00003820 C60600004D               mov     byte [ARENA.SIGNATURE],arena_signature_normal ; 'M'
39429                                   ;mov    word [1],8
39430 00003825 C70601000800             mov     word [ARENA.OWNER],8  ; mark as system owned
39431                                   ;mov    [3],cx
39432 0000382B 890E0300                 mov     [ARENA.SIZE],cx
39433                                   ;mov    word [8],4353h ; 'SC'
39434 0000382F C70608005343             mov     word [ARENA.NAME],'SC' ; 4353h
39435
39436                                   ; prepare the arena at start of new block
39437
39438 00003835 8EC3                     mov     es,bx
39439 00003837 26C60600004D             mov     byte [es:ARENA.SIGNATURE],arena_signature_normal ; 'M'
39440 0000383D 26C70601000000           mov     word [es:ARENA.OWNER],arena_owner_system ; 0
39441                                                                   ; mark as free
39442 00003844 83EA02                   sub     dx,2                    ; make room for arena at
39443                                                                   ; start & end of new block
39444 00003847 2689160300               mov     [es:ARENA.SIZE],dx
39445
39446                                   ; prepare arena at end of new block
39447
39448 0000384C 01D3                     add     bx,dx
39449 0000384E 43                       inc     bx
39450 0000384F 8EC3                     mov     es,bx                   ; es=arena at top of new block
39451 00003851 43                       inc     bx                      ; bx=top of new block
39452
39453                                                                   ; ax contains arena just above
39454                                                                   ; this block
39455 00003852 29D8                     sub     ax,bx                   ; ax = size of used block
39456
39457 00003854 26C60600004D             mov     byte [es:ARENA.SIGNATURE],arena_signature_normal
39458 0000385A 26C70601000800           mov     word [es:ARENA.OWNER],8       ; mark as system owned
39459 00003861 26A30300                 mov     [es:ARENA.SIZE],ax
39460 00003865 26C70608005343           mov     word [es:ARENA.NAME],'SC' ; 4353h
39461
39462 0000386C EB47                     jmp     short ui_done
39463
39464                                   ui_append:
39465                                                                   ; es = arena of last block
39466 0000386E 2603060300               add     ax,[es:ARENA.SIZE]    ; ax=top of last block-1 para
39467 00003873 26832E030001             sub     word [es:ARENA.SIZE],1; reflect the space we are
39468                                                                   ; going to rsrv on top of this
39469                                                                   ; block for the next arena.
39470                                   ; 13/05/2019
39471 00003879 26C60600004D             mov     byte [es:ARENA.SIGNATURE],arena_signature_normal
39472
39473 0000387F 89C1                     mov     cx,ax                   ; cx=top of prev block-1
39474 00003881 40                       inc     ax
39475 00003882 29D8                     sub     ax,bx                   ; ax=top of prev block -
39476                                                                   ;    seg. address of new block
39477 00003884 F7D8                     neg     ax
39478
39479 00003886 8EC1                     mov     es,cx                   ; ds = arena of unused block
39480
39481 00003888 26C60600004D             mov     byte [es:ARENA.SIGNATURE],arena_signature_normal
39482 0000388E 26C70601000800           mov     word [es:ARENA.OWNER],8       ; mark as system owned
39483 00003895 26A30300                 mov     [es:ARENA.SIZE],ax
39484 00003899 26C70608005343           mov     word [es:ARENA.NAME],'SC'
39485
39486                                   ; prepare the arena at start of new block
39487 000038A0 8EC3                     mov     es,bx
39488 000038A2 26C60600005A             mov     byte [es:ARENA.SIGNATURE],arena_signature_end
39489 000038A8 26C70601000000           mov     word [es:ARENA.OWNER],arena_owner_system
39490                                                                   ; mark as free
39491 000038AF 4A                       dec     dx                      ; make room for arena
39492 000038B0 2689160300               mov     [es:ARENA.SIZE],dx
39493                                   ui_done:
39494                                   uc_done: ; 31/12/2022 ; *!
39495 000038B5 1F                       pop     ds
39496                                   ; ds = cs ; 31/12/2022
39497                                   ;uc_done:   ; 18/12/2022
39498                                   au_exit:    ; 09/09/2023
39499 000038B6 C3                       retn
39500
39501                                   ;--------------------------------------------------------------------------
39502                                   ;
39503                                   ; procedure : AllocUMB
39504                                   ;
39505                                   ;   Allocate all UMBs and link it to DOS arena chain
39506                                   ;
39507                                   ;--------------------------------------------------------------------------
39508
39509                                   AllocUMB:
39510                                       ; 31/12/2022
39511                                       ; ds = cs
39512 000038B7 E84700                   call    InitAllocUMB            ; link in the first UMB
39513 000038BA 72FA                     jc      short au_exit           ; quit on error
39514                                   au_next:
39515 000038BC E87000                   call    umb_allocate            ; allocate
39516 000038BF 7205                     jc      short au_coalesce
39517 000038C1 E835FF                   call    umb_insert              ; & insert till no UMBs
39518 000038C4 EBF6                     jmp     short au_next
39519                                   au_coalesce:
39520                                       ; 09/09/2023
39521                                   ;   call    umb_coalesce          ; coalesce all UMBs
39522                                   ;au_exit:
39523                                   ;   ; 31/12/2022
39524                                   ;   ; ds = cs
39525                                   ;   retn
39526
39527                                       ; 09/09/2023
```

```
                                    ;jmp    short umb_coalesce

                                ; 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
```

```
39776
39777                                   ;mov    es,[cs:DevDOSData]
39778                                   ; 31/12/2022
39779 0000399F 8E06[6024]               mov     es,[DevDOSData] ; ds = cs
39780 000039A3 BF8C00                   mov     di,UMB_ARENA ; 8Ch
39781 000039A6 26890D                   mov     [es:di],cx              ; initialize umb_head in DOS
39782                                                                   ;  data segment with the arena
39783                                                                   ;  just below Top of Mem
39784
39785                                 ; we must now scan the arena chain and update the size of the last arena
39786
39787 000039A9 BF2400                   mov     di,DOS_ARENA ; 24h
39788 000039AC 268E05                   mov     es,[es:di]              ; es = start arena
39789 000039AF 31FF                     xor     di,di
39790                                 ;scan_next
39791                                 ; 09/12/2022
39792                                 scannext:
39793 000039B1 26803D5A                 cmp     byte [es:di],arena_signature_end  ; 'Z'
39794 000039B5 740C                     je      short got_last
39795
39796 000039B7 8CC0                     mov     ax,es
39797 000039B9 2603060300               add     ax,[es:ARENA.SIZE]
39798 000039BE 40                       inc     ax
39799 000039BF 8EC0                     mov     es,ax
39800                                   ;jmp    short scan_next
39801                                   ; 09/12/2022
39802 000039C1 EBEE                     jmp     short scannext
39803                                 got_last:
39804                                   ;sub    word [es:ARENA.SIZE],1
39805                                   ; 09/09/2023
39806 000039C3 26FF0E0300               dec     word [es:ARENA.SIZE]
39807
39808 000039C8 26C60600004D             mov     byte [es:ARENA.SIGNATURE],arena_signature_normal ; 'M'
39809                                   ;clc
39810                                   ; 01/11/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
39811                                   ; 11/12/2022
39812                                   ; cf=0
39813                                   ;clc
39814 000039CE C3                       retn
39815
39816                                 ; 11/12/2022
39817                                 ;;lfu_err:
39818                                 ;   ;stc
39819                                 ;   ; 01/11/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
39820                                 ;   ; 11/12/2022
39821                                 ;   ; cf=1
39822                                 ;   ;stc
39823                                 ;   retn
39824
39825                                 ;--------------------------------------------------------------------------
39826                                 ;
39827                                 ; procedure : ShrinkUMB
39828                                 ;
39829                                 ;   Shrinks the current UMB in use, so that the unused portions
39830                                 ;   of the UMB is given back to the DOS free mem pool
39831                                 ;
39832                                 ;--------------------------------------------------------------------------
39833
39834                                 ShrinkUMB:
39835                                     ; 12/12/2022
39836                                     ; ds = cs
39837 000039CF 833E[4324]00             cmp     word [DevUMBAddr],0
39838                                   ;cmp    word [cs:DevUMBAddr],0
39839 000039D4 741F                     je      short su_exit
39840 000039D6 06                       push    es
39841                                   ; 01/01/2023
39842                                   ;push   bx
39843                                   ; 12/12/2022
39844                                   ;mov    bx,[cs:DevUMBFree]
39845                                   ;sub    bx,[cs:DevUMBAddr]
39846                                   ;mov    es,[cs:DevUMBAddr]
39847 000039D7 8B1E[4724]               mov     bx,[DevUMBFree]
39848 000039DB 2B1E[4324]               sub     bx,[DevUMBAddr]
39849 000039DF 8E06[4324]               mov     es,[DevUMBAddr]
39850
39851 000039E3 B8004A                   mov     ax,4A00h
39852 000039E6 CD21                     int     21h
39853                                           ; DOS - 2+ - ADJUST MEMORY BLOCK SIZE (SETBLOCK)
39854                                           ; ES = segment address of block to change
39855                                           ; BX = new size in paragraphs
39856 000039E8 8CC0                     mov     ax,es
39857 000039EA 48                       dec     ax
39858 000039EB 8EC0                     mov     es,ax
39859 000039ED 26C70601000800           mov     word [es:ARENA.OWNER],8
39860                                   ; 01/01/2023
39861                                   ;pop    bx
39862 000039F4 07                       pop     es
39863                                 su_exit:
39864 000039F5 C3                       retn
39865
39866                                 ;--------------------------------------------------------------------------
39867                                 ;
39868                                 ; procedure : UnlinkUMB
39869                                 ;
39870                                 ;   Unlinks the UMBs from the DOS arena chain
39871                                 ;
39872                                 ;--------------------------------------------------------------------------
39873
39874                                 UnlinkUMB:
39875                                     ; 12/12/2022
39876                                     ; ds = cs
39877 000039F6 1E                       push    ds
39878 000039F7 06                       push    es
39879                                   ; 12/12/2022
39880 000039F8 803E[5F24]00             cmp     byte [FirstUMBLinked],0
39881                                   ;cmp    byte [cs:FirstUMBLinked],0
39882 000039FD 7420                     je      short ulu_x             ; nothing to unlink
39883                                   ; 12/12/2022
39884 000039FF 8E06[6024]               mov     es,[DevDOSData]
39885                                   ;mov    es,[cs:DevDOSData]      ; get DOS data seg
39886 00003A03 268E1E2400               mov     ds,[es:DOS_ARENA]
39887 00003A08 268B3E8C00               mov     di,[es:UMB_ARENA]
39888                                 ulu_next:
39889 00003A0D E83BFF                   call    get_next
39890 00003A10 720D                     jc      short ulu_x
39891 00003A12 39C7                     cmp     di,ax                   ; is the next one UMB ?
39892 00003A14 7404                     je      short ulu_found
39893 00003A16 8ED8                     mov     ds,ax
39894 00003A18 EBF3                     jmp     short ulu_next
39895                                 ulu_found:
39896                                   ;mov    byte [0],'Z'
39897 00003A1A C60600005A               mov     byte [ARENA.SIGNATURE],arena_signature_end ; 'Z'
39898                                 ulu_x:
39899 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)
                                ;   ;;mov   ax,Bios_Data          ; get Bios_Data segment
                                ;   ;mov    ax,KERNEL_SEGMENT ; 70h
                                ;   ; 21/10/2022
                                ;   ;mov    ax,DOSBIODATASEG ; 0070h
                                ;   ;mov    ds,ax                ; set Bios_Data segment
                                ;
                                ;   ; 27/07/2023
                                ;   ;;test word [cs:switches],flagec35 ; 4
                                ;   ;test  byte [cs:switches],flagec35
                                ;   ;jz     short not_ec35
                                ;
                                ;   ; 27/07/2023
                                ;   ; 01/11/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
                                ;   ;;test word [switches],flagec35 ; 4
                                ;   ; 12/12/2022
                                ;   ;test  byte [switches],flagec35 ; 4
                                ;   ;jz     short eot_ok
                                ;
                                    ;mov    cl,[cs:drive]         ; which drive was this for?
                                    ; 02/11/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
                                    ;mov    cl,[drive]
                                    ; 27/07/2023
                                    ;mov    ax,DOSBIODATASEG ; 0070h
                                    ;mov    ds,ax
00003A47 BA7000                     mov     dx,DOSBIODATASEG
00003A4A 8EDA                       mov     ds,dx
00003A4C F6C404                     test    ah,flagec35    ; test byte [cs:switches],flagec35
00003A4F 7408                       jz      short not_ec35
                                    ;mov    al,1                  ; assume drive 0
                                    ;shl    al,cl                 ; set proper bit depending on drive
                                    ;;or    [531h],al ; (MSDOS 6.21 IO.SYS Offset SYINIT:3EACh)
                                    ;or     [ec35_flag],al        ; set the bit in the permanent flags
                                    ; 27/07/2023
00003A51 B401                       mov     ah,1
00003A53 D2E4                       shl     ah,cl
00003A55 0826[A204]                 or      [ec35_flag],ah

                                ; 07/07/2023 - Retro DOS v4.2 IO.SYS (optimization)
                                ;   MSDOS 6.21 IO.SYS - SYINIT:3EB0h
                                ; 01/11/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
                                not_ec35:
                                ;   Now adjust the BIOS's EOT variable if our new drive has more
                                ;   sectors per track than any old ones.

                                    ; 27/07/2023
                                    ;;mov   al,[cs:deviceparameters+20]
                                    ;mov    al,[cs:deviceparameters+A_DEVICEPARAMETERS.DP_BPB+A_BPB.SECTORSPERTRACK]

                                    ;cmp    al,[12Ch] ; (MSDOS 6.21 IO.SYS Offset SYINIT:3EB4h)
00003A59 3A06[2C01]                 cmp     al,[eot]
00003A5D 7603                       jbe     short eot_ok
00003A5F A2[2C01]                   mov     [eot],al
                                eot_ok:
00003A62 5A                         pop     dx                    ; fix up all the registers
00003A63 59                         pop     cx
00003A64 5B                         pop     bx
00003A65 58                         pop     ax
00003A66 1F                         pop     ds ; 13/05/2019
00003A67 C3                         retn

                                ;--------------------------------------------------------------------------
                                ;
                                ; procedure : diddleback
                                ;
                                ; replace default values for further drivparm commands
                                ;
                                ;--------------------------------------------------------------------------

                                diddleback:
00003A68 1E                         push    ds
00003A69 0E                         push    cs
00003A6A 1F                         pop     ds
                                    ;mov    word [deviceparameters+4],80
00003A6B C706[C24D]5000             mov     word [deviceparameters+A_DEVICEPARAMETERS.DP_CYLINDERS],80
                                    ;mov    byte [deviceparameters+1],2
00003A71 C606[BF4D]02               mov     byte [deviceparameters+A_DEVICEPARAMETERS.DP_DEVICETYPE],DEV_3INCH720KB ; 2
                                    ;mov    word [deviceparameters+2],0
00003A76 C706[C04D]0000             mov     word [deviceparameters+A_DEVICEPARAMETERS.DP_DEVICEATTRIBUTES],0
00003A7C C706[1D4F]0000             mov     word [switches],0         ; zero all switches
00003A82 1F                         pop     ds
00003A83 C3                         retn

                                ; 03/01/2023
                                %if 0

                                ; 15/04/2019 - Retro DOS v4.0

                                ;--------------------------------------------------------------------------
                                ;
                                ; procedure : parseline
                                ;
                                ; entry point is parseline. al contains the first character in command line.
                                ;
                                ;--------------------------------------------------------------------------

                                    ; 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:
                                    cmp     al,cr                 ; carriage return?
                                    je      short done_line
                                    cmp     al,lf                 ; linefeed?
                                    je      short put_back        ; put it back and done

                                ; anything less or equal to a space is ignored.

                                    cmp     al,' '                  ; space?
                                    jbe     short getnext         ; skip over space
```

```
                            cmp     al,'/'
                            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
```

```
                        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
```

```
40520
40521 00003B47 06                       push    es
40522
40523 00003B48 0E                       push    cs
40524 00003B49 07                       pop     es
40525
40526 00003B4A 31DB                     xor     bx,bx
40527 00003B4C 8A1E[BF4D]               mov     bl,[deviceparameters+A_DEVICEPARAMETERS.DP_DEVICETYPE]
40528 00003B50 80FB00                   cmp     bl,DEV_5INCH ; 0
40529 00003B53 7506                     jne     short got_80
40530
40531 00003B55 C706[C24D]2800           mov     word [deviceparameters+A_DEVICEPARAMETERS.DP_CYLINDERS],40
40532                                                                   ; 48 tpi=40 cyl
40533                                   got_80:
40534 00003B5B D1E3                     shl     bx,1                    ; get index into bpb table
40535 00003B5D 8BB7[2E50]               mov     si,[bpbtable+bx]        ; get address of bpb
40536
40537                                   ;mov    di,deviceparameters+7
40538                                   ; 02/11/2022
40539 00003B61 BF[C54D]                 mov     di,deviceparameters+A_DEVICEPARAMETERS.DP_BPB ; es:di -> bpb
40540 00003B64 B93B00                   mov     cx,A_BPB.size ; 31
40541                                   ; 09/09/2023
40542                                   ;mov    cx,59 ; PCDOS 7.1 IBMBIO.COM A_BPB.size
40543 00003B67 FC                       cld
40544                                   ;repe   movsb
40545                                   ; 02/11/2022
40546 00003B68 F3A4                     rep     movsb
40547
40548 00003B6A 07                       pop     es
40549
40550                                   ; 12/12/2022
40551 00003B6B F606[1D4F]20             test    byte [switches],flagseclim ; 20h
40552                                   ;test   word [switches],flagseclim ; 20h
40553 00003B70 7406                     jz      short see_heads
40554
40555 00003B72 A1[1A4F]                 mov     ax,[slim]
40556                                   ;mov    [deviceparameters+20],ax
40557 00003B75 A3[D24D]                 mov     [deviceparameters+A_DEVICEPARAMETERS.DP_BPB+A_BPB.SECTORSPERTRACK],ax
40558
40559                                   see_heads:
40560                                       ; 12/12/2022
40561 00003B78 F606[1D4F]40                 test    byte [switches],flagheads ; 40h
40562                                       ;test   word [switches],flagheads ; 40h
40563 00003B7D 7406                         jz      short heads_not_altered
40564
40565 00003B7F A1[184F]                     mov     ax,[hlim]
40566                                       ;mov    [deviceparameters+22],ax
40567 00003B82 A3[D44D]                     mov     [deviceparameters+A_DEVICEPARAMETERS.DP_BPB+A_BPB.HEADS],ax
40568
40569                                   heads_not_altered:
40570
40571                                   ; set up correct media descriptor byte and sectors/cluster
40572                                   ;   sectors/cluster is always 2 except for any one sided disk or 1.44M
40573
40574                                       ;mov    byte [deviceparameters+9],2
40575                                       ; 02/11/2022
40576                                       ;mov    byte [deviceparameters+A_DEVICEPARAMETERS.DP_BPB+A_BPB.SECTORSPERCLUSTER],2
40577                                       ; 03/01/2023
40578 00003B85 B80200                       mov     ax,2
40579 00003B88 A2[C74D]                     mov     [deviceparameters+A_DEVICEPARAMETERS.DP_BPB+A_BPB.SECTORSPERCLUSTER],al ; 2
40580
40581 00003B8B B3F0                         mov     bl,0F0h                 ; get default mediabyte
40582
40583                                   ;   preload the mediadescriptor from the bpb into bh for convenient access
40584
40585                                       ;mov    bh,[deviceparameters+17]
40586                                       ; 02/11/2022
40587 00003B8D 8A3E[CF4D]                   mov     bh,[deviceparameters+A_DEVICEPARAMETERS.DP_BPB+A_BPB.MEDIADESCRIPTOR]
40588
40589                                       ; 03/01/2023
40590                                       ; ax = 2
40591 00003B91 3906[D44D]                   cmp     [deviceparameters+A_DEVICEPARAMETERS.DP_BPB+A_BPB.HEADS],ax ; >2 heads?
40592                                       ;cmp    word [deviceparameters+A_DEVICEPARAMETERS.DP_BPB+A_BPB.HEADS],2 ; >2 heads?
40593 00003B95 773C                         ja      short got_correct_mediad ; just use default if heads>2
40594
40595 00003B97 7524                         jne     short only_one_head   ; one head, do one head stuff
40596
40597                                   ;   two head drives will use the mediadescriptor from the bpb
40598
40599 00003B99 88FB                         mov     bl,bh                   ; get mediadescriptor from bpb
40600
40601                                   ;   two sided drives have two special cases to look for. One is
40602                                   ;      a 320K diskette (40 tracks, 8 secs per track). It uses
40603                                   ;      a mediaid of 0fch. The other is 1.44M, which uses only
40604                                   ;      one sector/cluster.
40605
40606                                   ;   any drive with 18secs/trk, 2 heads, 80 tracks, will be assumed
40607                                   ;      to be a 1.44M and use only 1 sector per cluster. Any other
40608                                   ;      type of 2 headed drive is all set.
40609
40610 00003B9B 833E[D24D]12                 cmp     word [deviceparameters+A_DEVICEPARAMETERS.DP_BPB+A_BPB.SECTORSPERTRACK],18
40611 00003BA0 7509                         jne     short not_144m
40612 00003BA2 833E[C24D]50                 cmp     word [deviceparameters+A_DEVICEPARAMETERS.DP_CYLINDERS],80
40613 00003BA7 7502                         jne     short not_144m
40614
40615                                   ;   We've got cyl=80, heads=2, secpertrack=18. Set cluster size to 1.
40616
40617 00003BA9 EB24                         jmp     short got_one_secperclus_drive
40618
40619                                   ;   check for 320K
40620
40621                                   not_144m:
40622 00003BAB 833E[C24D]28                 cmp     word [deviceparameters+A_DEVICEPARAMETERS.DP_CYLINDERS],40
40623 00003BB0 7521                         jne     short got_correct_mediad
40624 00003BB2 833E[D24D]08                 cmp     word [deviceparameters+A_DEVICEPARAMETERS.DP_BPB+A_BPB.SECTORSPERTRACK],8
40625 00003BB7 751A                         jne     short got_correct_mediad
40626
40627 00003BB9 B3FC                         mov     bl,0FCh
40628 00003BBB EB16                         jmp     short got_correct_mediad
40629
40630                                   only_one_head:
40631
40632                                   ;   if we don't have a 360K drive, then just go use 0f0h as media descr.
40633
40634 00003BBD 803E[BF4D]00                 cmp     byte [deviceparameters+A_DEVICEPARAMETERS.DP_DEVICETYPE],DEV_5INCH ; 0
40635 00003BC2 740B                         je      short got_one_secperclus_drive
40636
40637                                   ;   single sided 360K drive uses either 0fch or 0feh, depending on
40638                                   ;     whether sectorspertrack is 8 or 9. For our purposes, anything
40639                                   ;     besides 8 will be considered 0fch
40640
40641 00003BC4 B3FC                         mov     bl,0FCh                 ; single sided 9 sector media id
40642 00003BC6 833E[D24D]08                 cmp     word [deviceparameters+A_DEVICEPARAMETERS.DP_BPB+A_BPB.SECTORSPERTRACK],8
40643                                       ; 12/12/2022
```

```
                                    jne     short got_one_secperclus_drive ; okay if anything besides 8
                                    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
                                    dec     ax  ; ax = 1
                                    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:
                                    mov     [deviceparameters+A_DEVICEPARAMETERS.DP_BPB+A_BPB.MEDIADESCRIPTOR],bl

                                ;    Calculate the correct number of Total Sectors on medium

                                    mov     ax,[deviceparameters+A_DEVICEPARAMETERS.DP_CYLINDERS]
                                    mul     word [deviceparameters+A_DEVICEPARAMETERS.DP_BPB+A_BPB.HEADS]
                                    mul     word [deviceparameters+A_DEVICEPARAMETERS.DP_BPB+A_BPB.SECTORSPERTRACK]
                                    mov     word [deviceparameters+A_DEVICEPARAMETERS.DP_BPB+A_BPB.TOTALSECTORS],ax
                                    clc                            ; we currently return no errors

                                    retn

                                ;   M047 -- end rewritten routine

                                ;--------------------------------------------------------------------------
                                ;
                                ; procedure : organize
                                ;
                                ;--------------------------------------------------------------------------

                                ; 09/09/2023 - Retrodos v4.2 IO.SYS (Optimization)
                                %if 1
                                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
                                    ; 09/09/2023
                                    jmp     short org0
                                nochar1:
                                    stc
                                    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
                                    mov     cx,[count]
                                    ;mov    cx,[cs:count]
                                    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

                                    xor     si,si
                                    mov     di,si
                                    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:
                                    mov     byte [com_level],0 ; 09/09/2023
                                org1:
                                    call    skip_comment
                                    jz      short end_commd_line  ; found a comment string and skipped.
                                    call    get2                  ; not a comment string. then get a char.
                                    cmp     al,lf ; 0Ah
                                    je      short end_commd_line  ; starts with a blank line.
                                    cmp     al,' ' ; 20h
                                    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:
                                    push    cx
                                    push    si
                                    push    di
                                    mov     bp,si
                                    dec     bp
                                        mov     si,comtab          ; prepare to search command table
                                    mov     ch,0
```

```
                                findcom:
00003C18 89EF                       mov     di,bp
00003C1A 8A0C                       mov     cl,[si]
00003C1C 46                         inc     si
00003C1D E345                       jcxz    nocom

                                    ; 02/11/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)

                                ; 03/01/2023 - Retrodos v4.2 (Modified MSDOS 6.21 IO.SYS)

                                ;ifdef      MULTI_CONFIG

                                ;   Simplify future parsing by collapsing ";" onto "REM", and at the same
                                ;   time skip the upcoming delimiter test (since ";" need not be followed by
                                ;   anything in particular)

00003C1F 26803D3B                   cmp     byte [es:di],CONFIG_SEMICOLON  ; ';'
00003C23 7430                       je      short semicolon
                                loopcom:
                                    ;mov    al,[es:di]
                                    ;inc    di
                                    ;and    al,~20h ; 0DFh        ; force upper case
                                    ;inc    si                      ; compare to byte @es:di
                                    ;cmp    al,[si-1]
                                    ; 28/07/2023 - Retro DOS v4.2 IO.SYS (optimization)
00003C25 268A25                     mov     ah,[es:di]
00003C28 47                         inc     di
00003C29 80E4DF                     and     ah,~20h ; 0DFh
00003C2C AC                         lodsb                   ; mov al,[si]
                                                            ; inc si
                                    ;cmp    al,ah
                                    ;loope  loopcom
                                    ; 28/07/2023
00003C2D 30C4                       xor     ah,al           ; result: ah = 0 (*) if ah = al
00003C2F E1F4                       loopz   loopcom
                                ;else
                                ;   repe    cmpsb
                                ;endif
                                    ; 02/11/2022
                                    ; 03/01/2023 - Retro DOS v4.2
                                    ;repe   cmpsb

                                    ; 28/07/2023
                                    ;lahf
00003C31 01CE                           add     si,cx                   ; bump to next position without affecting flags
                                    ;sahf
00003C33 AC                             lodsb                           ; get indicator letter
                                    ;jnz    short findcom
                                        ; 28/07/2023
00003C34 08E4                       or      ah,ah                   ; (*)
00003C36 75E0                       jnz     short findcom

00003C38 26803D0D                   cmp     byte [es:di],cr             ; the next char might be cr,lf
00003C3C 7421                       je      short gotcom0           ; such as in "rem",cr,lf case.
00003C3E 26803D0A                   cmp     byte [es:di],lf
00003C42 741B                       je      short gotcom0

                                    ; 02/11/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)

                                ; 03/01/2023 - Retrodos v4.2 (Modified MSDOS 6.21 IO.SYS)

                                ;ifdef      MULTI_CONFIG

                                ; Skip the delimiter test for the BEGIN identifier (it doesn't have one).

00003C44 3C5B                       cmp     al,CONFIG_BEGIN  ; '['
00003C46 7417                       je      short gotcom0
                                ;endif
00003C48 50                         push    ax
00003C49 268A05                     mov     al,[es:di]              ; now the next char. should be a delim.

                                ;ifdef      MULTI_CONFIG

                                ;   If keyword is *immediately* followed by a question mark (?), then
                                ;   set the high bit of the ASCII command code (CONFIG_OPTION_QUERY) that is
                                ;   stored in the CONFIG.SYS memory image.

00003C4C 3C3F                       cmp     al,'?'                  ; explicit interactive command?
00003C4E 7509                       jne     short no_query          ; no
00003C50 58                         pop     ax                      ; yes, so retrieve the original code
                                    ;or     al,80h ; 03/01/2023
00003C51 0C80                       or      al,CONFIG_OPTION_QUERY  ; and set the QUERY bit
00003C53 EB0A                       jmp     short gotcom0           ;
                                semicolon:
00003C55 B030                       mov     al,CONFIG_REM ; '0'
00003C57 EB06                       jmp     short gotcom0
                                no_query:
                                ;endif  ;MULTI_CONFIG

                                    ; 02/11/2022
                                    ; 03/01/2023 - Retro DOS v4.2
                                    ;push   ax
                                    ;mov    al,[es:di]              ; now the next char. should be a delim.

00003C59 E82E0B                     call    delim
                                no_delim:
00003C5C 58                         pop     ax
00003C5D 75B9                       jnz     short findcom
                                gotcom0:
00003C5F 5F                         pop     di
00003C60 5E                         pop     si
00003C61 59                         pop     cx
00003C62 EB10                       jmp     short gotcom
                                nocom:
00003C64 5F                         pop     di
00003C65 5E                         pop     si
00003C66 59                         pop     cx
00003C67 B05A                           mov     al,CONFIG_UNKNOWN  ; 'Z'
00003C69 AA                         stosb                           ; save indicator char.
                                _skipline:
00003C6A E86801                     call    get2
00003C6D 3C0A                       cmp     al,lf ; 0Ah             ; skip this bad command line
00003C6F 75F9                           jne     short _skipline
                                    ;jmp    short end_commd_line  ; handle next command line
                                    ; 09/09/2023
00003C71 E973FF                     jmp     end_commd_line
                                gotcom:
00003C74 AA                             stosb                               ; save indicator char in buffer

                                    ; 02/11/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)

                                ; 03/01/2023 - Retrodos v4.2 (Modified MSDOS 6.21 IO.SYS)

                                ;ifdef      MULTI_CONFIG
```

```
40892
40893                                   ;   Don't pollute "cmd_indicator" with the CONFIG_OPTION_QUERY bit though;
40894                                   ;   it screws up the direct comparisons below.
40895
40896 00003C75 247F                        and     al,~CONFIG_OPTION_QUERY ; 7Fh
40897                                   ;endif
40898                                       ;mov    [cs:cmd_indicator],al ; save it for the future use.
40899                                       ; 03/01/2023
40900                                       ; ds = cs
40901 00003C77 A2[5403]                   mov     [cmd_indicator],al    ; save it for the future use.
40902
40903                                   ;ifdef      MULTI_CONFIG
40904
40905                                   ;   There is no whitespace/delimiter between the "begin block" character
40906                                   ;   ([) and the name of block (eg, [menu]), therefore skip this delimiter
40907                                   ;   skipping code
40908
40909 00003C7A 3C5B                        cmp     al,CONFIG_BEGIN
40910 00003C7C 7455                        je      short org31
40911 00003C7E 3C4F                        cmp     al,CONFIG_SUBMENU ; 'O'
40912 00003C80 740F                        je      short no_mapcase
40913 00003C82 3C45                        cmp     al,CONFIG_MENUITEM ; 'E'
40914 00003C84 740B                        je      short no_mapcase
40915 00003C86 3C41                        cmp     al,CONFIG_MENUDEFAULT ; 'A'
40916 00003C88 7407                        je      short no_mapcase
40917 00003C8A 3C4A                        cmp     al,CONFIG_INCLUDE ; 'J'
40918 00003C8C 7403                        je      short no_mapcase
40919 00003C8E E8350B                      call    mapcase               ; map case of rest of line to UPPER
40920                                   no_mapcase:
40921                                   ;endif
40922                                       ;; 02/11/2022
40923                                       ;;mov   [cs:cmd_indicator],al ; save it for the future use.
40924                                       ;; 03/01/2023
40925                                       ;; ds = cs
40926                                       ;mov    [cmd_indicator],al
40927                                   org2:
40928 00003C91 E84101                      call    get2                    ; skip the command name until delimiter
40929 00003C94 3C0A                           cmp     al,lf         ; 0Ah
40930 00003C96 740F                        je      short org21
40931 00003C98 3C0D                        cmp     al,cr       ; 0Dh
40932 00003C9A 740B                        je      short org21
40933                                      ; 02/11/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
40934                                      ; 03/01/2023 - Retro DOS v4.2
40935 00003C9C 3C2F                        cmp     al, '/'               ; T-RICHJ: Added to allow DEVHIGH/L:...
40936 00003C9E 7407                        je      short org21           ; T-RICHJ: to be parsed properly.
40937
40938 00003CA0 E8E70A                      call    delim
40939 00003CA3 75EC                           jnz short org2
40940 00003CA5 EB02                        jmp     short org3
40941                                   org21:                                  ;if cr or lf then
40942 00003CA7 4E                          dec     si                    ; undo si, cx register
40943 00003CA8 41                          inc     cx                    ;  and continue
40944                                   org3:
40945                                       ;cmp    byte [cs:cmd_indicator],CONFIG_COMMENT ; 'Y'
40946                                       ;je     short get_cmt_token
40947                                       ; 02/11/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
40948                                       ; 03/01/2023 - Retro DOS v4.2
40949                                       ;cmp    byte [cs:cmd_indicator],CONFIG_DEVICE ; 'D'
40950                                       ;je     short org_file
40951                                       ;cmp    byte [cs:cmd_indicator],CONFIG_INSTALL ; 'I'
40952                                       ;je     short org_file
40953                                       ;cmp    byte [cs:cmd_indicator],CONFIG_INSTALLHIGH ; 'W'
40954                                       ;je     short org_file
40955                                       ; 02/11/2022
40956                                       ; 03/01/2023 - Retro DOS v4.2
40957                                       ;;cmp   byte [cs:cmd_indicator],CONFIG_DEVICE ; 'D'
40958                                       ;;je    short org_file
40959                                       ;cmp    byte [cs:cmd_indicator],CONFIG_SHELL ; 'S'
40960                                       ;je     short org_file
40961                                          ;cmp       byte [cs:cmd_indicator],CONFIG_SWITCHES ; '1'
40962                                       ;je     short org_switch
40963
40964                                       ; 03/01/2023
40965                                       ; ds = cs
40966
40967 00003CA9 803E[5403]59                cmp     byte [cmd_indicator],CONFIG_COMMENT ; 'Y'
40968 00003CAE 745D                        je      short get_cmt_token
40969                                      ; 02/11/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
40970                                      ; 03/01/2023 - Retro DOS v4.2
40971 00003CB0 803E[5403]44                cmp     byte [cmd_indicator],CONFIG_DEVICE ; 'D'
40972 00003CB5 7430                        je      short org_file
40973 00003CB7 803E[5403]49                cmp     byte [cmd_indicator],CONFIG_INSTALL ; 'I'
40974 00003CBC 7429                        je      short org_file
40975 00003CBE 803E[5403]57                cmp     byte [cmd_indicator],CONFIG_INSTALLHIGH ; 'W'
40976 00003CC3 7422                        je      short org_file
40977                                      ; 02/11/2022
40978                                      ; 03/01/2023 - Retro DOS v4.2
40979                                      ;cmp    byte [cmd_indicator],CONFIG_DEVICE ; 'D'
40980                                      ;je     short org_file
40981 00003CC5 803E[5403]53                   cmp byte [cmd_indicator],CONFIG_SHELL ; 'S'
40982 00003CCA 741B                        je      short org_file
40983 00003CCC 803E[5403]31                   cmp byte [cmd_indicator],CONFIG_SWITCHES ; '1'
40984 00003CD1 7403                        je      short org_switch
40985
40986                                   org31:
40987 00003CD3 E99500                      jmp     org4
40988
40989                                   org_switch:
40990 00003CD6 E81601                      call    skip_comment
40991 00003CD9 7472                        jz      short end_commd_line_brdg
40992
40993 00003CDB E8F700                      call    get2
40994 00003CDE E8B10A                      call    org_delim
40995 00003CE1 74F3                        jz      short org_switch
40996
40997 00003CE3 AA                          stosb
40998 00003CE4 E99300                      jmp     org5
40999
41000                                   org_file:                 ; get the filename and put 0 at end
41001 00003CE7 E80501                      call    skip_comment
41002 00003CEA 7464                        jz      short org_put_zero
41003
41004 00003CEC E8E600                      call    get2           ; not a comment
41005 00003CEF E8980A                      call    delim
41006 00003CF2 74F3                        jz      short org_file ; skip the possible delimiters
41007
41008 00003CF4 AA                          stosb                  ; copy the first non delim char found in buffer
41009
41010                                   org_copy_file:
41011 00003CF5 E8F700                      call    skip_comment   ; comment char in the filename?
41012 00003CF8 7456                        jz      short org_put_zero ; then stop copying filename at that point
41013
41014 00003CFA E8D800                      call    get2
41015 00003CFD 3C2F                        cmp     al,'/'         ; a switch char? (device=filename/xxx)
```

```
41016 00003CFF 7457                     je      short end_file_slash ; this will be the special case.
41017
41018 00003D01 AA                       stosb                   ; save the char. in buffer
41019 00003D02 E8850A                   call    delim
41020 00003D05 7459                     jz      short end_copy_file
41021
41022 00003D07 3C20                     cmp     al, ' '
41023 00003D09 77EA                     ja      short org_copy_file ; keep copying
41024 00003D0B EB53                     jmp     short end_copy_file ; otherwise, assume end of the filename.
41025
41026                                 get_cmt_token:                  ; get the token. just max. 2 char.
41027 00003D0D E8C500                   call    get2
41028 00003D10 3C20                     cmp     al,' '          ; skip white spaces or "=" char.
41029 00003D12 74F9                     je      short get_cmt_token ; (we are allowing the other special
41030 00003D14 3C09                     cmp     al,tab ; 9      ;  characters can used for comment id.
41031 00003D16 74F5                     je      short get_cmt_token ;  character.)
41032 00003D18 3C3D                     cmp     al,'='          ; = is special in this case.
41033 00003D1A 74F1                     je      short get_cmt_token
41034 00003D1C 3C0D                     cmp     al,cr
41035 00003D1E 7426                     je      short get_cmt_end ; cannot accept the carriage return
41036 00003D20 3C0A                     cmp     al,lf
41037 00003D22 7422                     je      short get_cmt_end
41038
41039                                   ; 03/01/2023
41040                                   ; ds = cs
41041                                   ;mov    [cs:cmmt1],al  ; store it
41042                                   ;mov    byte [cs:cmmt],1 ; 1 char. so far.
41043 00003D24 A2[5203]                 mov     [cmmt1],al      ; store it
41044 00003D27 C606[5103]01             mov     byte [cmmt],1  ; 1 char. so far.
41045 00003D2C E8A600                   call    get2
41046 00003D2F 3C20                     cmp     al,' ' ; 20h
41047 00003D31 7413                     je      short get_cmt_end
41048 00003D33 3C09                     cmp     al,tab ; 9
41049 00003D35 740F                     je      short get_cmt_end
41050 00003D37 3C0D                     cmp     al,cr  ; 0Dh
41051 00003D39 740B                     je      short get_cmt_end
41052 00003D3B 3C0A                     cmp     al,lf  ; 0Ah
41053 00003D3D 740E                     je      short end_commd_line_brdg
41054
41055                                   ;mov    [cs:cmmt2],al
41056                                   ;inc    byte [cs:cmmt]
41057                                   ; 03/01/2023
41058 00003D3F A2[5303]                 mov     [cmmt2],al
41059 00003D42 FE06[5103]               inc     byte [cmmt]
41060
41061                                 get_cmt_end:
41062 00003D46 E88C00                   call    get2
41063 00003D49 3C0A                     cmp     al,lf
41064 00003D4B 75F9                     jne     short get_cmt_end     ; skip it.
41065                                 end_commd_line_brdg:
41066 00003D4D E997FE                   jmp     end_commd_line        ; else jmp to end_commd_line
41067
41068                                 org_put_zero:                           ; make the filename in front of
41069 00003D50 26C60500                 mov     byte [es:di],0        ;  the comment string to be an asciiz.
41070 00003D54 47                       inc     di
41071 00003D55 E98FFE                   jmp     end_commd_line        ;  (maybe null if device=/*)
41072
41073                                 end_file_slash:                 ; al = "/" option char.
41074 00003D58 26C60500                 mov     byte [es:di],0        ; make a filename an asciiz
41075 00003D5C 47                       inc     di                    ; and
41076 00003D5D AA                       stosb                         ; store "/" after that.
41077 00003D5E EB1A                     jmp     short org5            ; continue with the rest of the line
41078
41079                                 end_copy_file:
41080 00003D60 26C645FF00               mov     byte [es:di-1],0      ; make it an asciiz and handle the next char.
41081 00003D65 3C0A                     cmp     al,lf
41082 00003D67 74E4                     je      short end_commd_line_brdg
41083 00003D69 EB0F                     jmp     short org5
41084
41085                                 org4:                                   ; org4 skips all delimiters after the command name except for '/'
41086 00003D6B E88100                   call    skip_comment
41087 00003D6E 74DD                     jz      short end_commd_line_brdg
41088
41089 00003D70 E86200                   call    get2
41090 00003D73 E81C0A                   call    org_delim             ; skip delimiters except '/' (mrw 4/88)
41091 00003D76 74F3                     jz      short org4
41092 00003D78 EB08                     jmp     short org51
41093
41094                                 org5:                                   ; rest of the line
41095 00003D7A E87200                   call    skip_comment          ; comment?
41096 00003D7D 74CE                     jz      short end_commd_line_brdg
41097 00003D7F E85300                   call    get2                  ; not a comment.
41098
41099                                 org51:
41100 00003D82 AA                       stosb                         ; copy the character
41101 00003D83 3C22                     cmp     al,'"'  ; 22h         ; a quote ?
41102 00003D85 743A                     je      short at_quote
41103 00003D87 3C20                     cmp     al,' '  ; 20h
41104 00003D89 77EF                     ja      short org5
41105
41106                                   ; 09/09/2023
41107                                   ; (Note: PCDOS 7.1 IBMBIO.COM does not contain M051 modification)
41108
41109                                                                   ; M051 - Start
41110                                   ; 03/01/2023
41111                                   ; ds = cs
41112 00003D8B 803E[5403]55                 cmp byte [cmd_indicator],CONFIG_DEVICEHIGH
41113                                   ;cmp    byte [cs:cmd_indicator],CONFIG_DEVICEHIGH ; Q: is this devicehigh
41114 00003D90 7514                     jne     short not_dh          ; N:
41115 00003D92 3C0A                     cmp     al,lf                 ; Q: is this line feed
41116 00003D94 7416                     je      short org_dhlf        ; Y: stuff a blank before the lf
41117 00003D96 3C0D                     cmp     al,cr                 ; Q: is this a cr
41118 00003D98 75E0                     jne     short org5            ; N:
41119 00003D9A 26C645FF20               mov     byte [es:di-1],' '    ; overwrite cr with blank
41120 00003D9F AA                       stosb                         ; put cr after blank
41121 00003DA0 FE06[223A]               inc     byte [insert_blank]
41122                                   ;inc    byte [cs:insert_blank]; indicate that blank has been
41123                                                                 ; inserted
41124 00003DA4 EBD4                     jmp     short org5
41125                                 not_dh:                                 ; M051 - End
41126
41127 00003DA6 3C0A                     cmp     al,lf                 ; line feed?
41128 00003DA8 740F                     je      short org1_brdg               ; handles the next command line.
41129 00003DAA EBCE                     jmp     short org5            ; handles next char in this line.
41130
41131                                 org_dhlf:                       ; M051 - Start
41132                                   ; 03/01/2023
41133                                   ; ds = cs
41134 00003DAC 803E[223A]01             cmp     byte [insert_blank],1
41135                                   ;cmp    byte [cs:insert_blank],1 ; Q:has a blank already been inserted
41136 00003DB1 7406                     je      short org1_brdg               ; Y:
41137 00003DB3 26C645FF20               mov     byte [es:di-1],' '    ; overwrite lf with blank
41138 00003DB8 AA                       stosb                         ; put lf after blank
41139                                                                 ; M051 - End
```

```
                                org1_brdg:
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.
```

```
41264 00003E1B 268A04                      mov     al,[es:si]
41265 00003E1E 46                          inc     si
41266 00003E1F 49                          dec     cx
41267 00003E20 3C0A                        cmp     al,lf           ; line feed?
41268 00003E22 75F5                        jne     short skip_cmmt
41269                                  no_commt:
41270 00003E24 C3                          retn
41271
41272                                  ; 01/01/2023 - Retro DOS v4.2 (Modified MSDOS 6.21 IO.SYS)
41273                                  ; (SYSINIT:42C8h)
41274
41275                                  ; 02/11/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
41276                                  ;%if 0
41277
41278                                  ;ifdef      MULTI_CONFIG
41279
41280                                  ;--------------------------------------------------------------------------
41281                                  ;
41282                                  ;   kbd_read: wait for keystroke
41283                                  ;
41284                                  ;   INPUT
41285                                  ;       DS == CS == sysinitseg
41286                                  ;
41287                                  ;   OUTPUT
41288                                  ;       Carry SET to clean boot, CLEAR otherwise
41289                                  ;
41290                                  ;   OTHER REGS USED
41291                                  ;       All
41292                                  ;
41293                                  ;   HISTORY
41294                                  ;       Created 16-Nov-1992 by JeffPar
41295                                  ;
41296                                  ;--------------------------------------------------------------------------
41297
41298                                  kbd_read:
41299 00003E25 F606[814C]02                      test    byte [bDisableUI],2
41300 00003E2A 7520                              jnz     short kbd_nodelay
41301
41302 00003E2C 1E                                push    ds              ; the bios timer tick count is incremented
41303 00003E2D 29C0                              sub     ax,ax           ; 18.2 times per second;
41304 00003E2F 8ED8                              mov     ds,ax           ; watch the timer tick count for 37 transitions
41305                                        ;mov    dx,[046Ch]      ; get initial value
41306                                  kbd_loop:
41307 00003E31 B401                              mov     ah,1            ;
41308 00003E33 CD16                              int     16h             ; peek the keyboard
41309 00003E35 7514                              jnz short kbd_loopdone ; something's there, get out
41310 00003E37 B402                              mov     ah,2            ; peek the shift states
41311 00003E39 CD16                              int     16h             ;
41312 00003E3B A803                              test    al,03h          ; either right or left shift key bits set?
41313 00003E3D 750C                              jnz short kbd_loopdone ; yes
41314 00003E3F A16C04                            mov     ax,[046Ch] ;
41315                                        ;sub    ax,dx           ; get difference
41316                                          ; 15/04/2019 - Retro DOS v4.0
41317 00003E42 2E2B06[8C03]                    sub     ax,[cs:_timer_lw_] ; MSDOS 6.21 IO.SYS - SYSINIT:42E5h
41318
41319 00003E47 3C25                            cmp     al,37           ; reached limit?  ; (2 seconds)
41320 00003E49 72E6                              jb  short kbd_loop ; not yet
41321                                  kbd_loopdone:
41322 00003E4B 1F                                pop     ds              ; delay complete!
41323                                  kbd_nodelay:
41324 00003E4C 29DB                              sub     bx,bx           ; assume clean boot
41325 00003E4E B402                              mov     ah,2            ; peek the shift states
41326 00003E50 CD16                              int     16h             ;
41327 00003E52 A803                              test    al,03h          ; either right or left shift key bits set?
41328 00003E54 7407                              jz      short kbd_notshift ; no
41329 00003E56 43                                inc     bx              ; yes
41330 00003E57 43                                inc     bx
41331                                          ; MSDOS 6.21 IO.SYS - SYSINIT:4301h
41332 00003E58 800E[854C]04                    or      byte [bQueryOpt],4
41333                                  kbd_notshift:                   ;
41334 00003E5D B401                              mov     ah,1            ; peek the keyboard
41335 00003E5F CD16                              int     16h             ;
41336 00003E61 743E                              jz  short kbd_test ; no key present
41337 00003E63 08C0                              or      al,al           ; is it a function key?
41338 00003E65 753A                              jnz short kbd_test ; no
41339
41340                                          ; MSDOS 6.21 IO.SYS - SYSINIT:430Bh
41341 00003E67 80FC62                            cmp     ah,62h          ; CTRL F5
41342 00003E6A 7405                              je  short kbd_cfg_bypass
41343
41344 00003E6C 80FC3F                            cmp     ah,3Fh          ; F5 function key?
41345 00003E6F 750D                              jne short kbd_notf5       ; no
41346                                  kbd_cfg_bypass:
41347 00003E71 BA[FC51]                          mov     dx,_$CleanMsg
41348 00003E74 E8DD0B                            call    print
41349                                          ; MSDOS 6.21 IO.SYS - SYSINIT:431Bh
41350 00003E77 800E[854C]04                    or      byte [bQueryOpt],4
41351 00003E7C EB16                              jmp     short kbd_eat   ; yes, clean boot selected
41352                                  kbd_notf5:
41353                                      ; MSDOS 6.21 IO.SYS - SYSINIT:4322h
41354 00003E7E 80FC65                            cmp     ah,65h          ; CTRL F8
41355 00003E81 7405                              je  short kbd_cfg_confirm
41356
41357 00003E83 80FC42                            cmp     ah,42h          ; F8 function key?
41358 00003E86 7523                              jne short kbd_exit ; no
41359                                  kbd_cfg_confirm:
41360 00003E88 BA[3A52]                          mov     dx,_$InterMsg
41361 00003E8B E8C60B                            call    print           ;
41362 00003E8E B301                              mov     bl,1            ; yes, interactive-boot option enabled
41363 00003E90 881E[854C]                        mov     [bQueryOpt],bl  ; change default setting
41364                                  kbd_eat:                        ;
41365 00003E94 B400                              mov     ah,0            ;
41366 00003E96 CD16                              int     16h             ; eat the key we assumed was a signal
41367 00003E98 C606[8B4C]FF                      mov byte [secElapsed],-1
41368 00003E9D 09DB                              or      bx,bx           ;
41369 00003E9F 7405                              jz  short kbd_clean      ;
41370                                  kbd_test:                       ;
41371 00003EA1 80FB02                            cmp     bl,2            ;
41372 00003EA4 7205                              jb  short kbd_exit ;
41373                                  kbd_clean:                      ;
41374 00003EA6 E86E08                            call    disable_autoexec; yes, tell COMMAND to skip autoexec.bat
41375 00003EA9 F9                                stc                     ; set carry to indicate abort
41376 00003EAA C3                                retn                     ;
41377                                  kbd_exit:                       ;
41378 00003EAB F8                                clc                     ; clear carry to indicate success
41379 00003EAC C3                                retn                     ;
41380
41381                                  ;--------------------------------------------------------------------------
41382                                  ;
41383                                  ;   set_numlock: set numlock LED
41384                                  ;
41385                                  ;   INPUT
41386                                  ;       ES:SI -> numlock setting (ie, "ON" or "OFF")
41387                                  ;
```

```
                                ;   OUTPUT
                                ;       None
                                ;
                                ;   OTHER REGS USED
                                ;       None
                                ;
                                ;   HISTORY
                                ;       Created 16-Nov-1992 by JeffPar
                                ;
                                ;-----------------------------------------------------------------------------

                                    ; 04/01/2023 - Retro DOS v4.2

                                set_numlock:
                                        ; 04/01/2023
                                    ;push   ax
00003EAD 1E                             push    ds
00003EAE 29C0                           sub     ax,ax
00003EB0 8ED8                           mov     ds,ax
00003EB2 268B04                         mov     ax,[es:si]      ; get 1st 2 bytes of value (ON or OF)
00003EB5 2E3B06[C451]                   cmp     ax,[cs:OnOff+2]   ; should we turn it off?
00003EBA 7507                           jne short not_off  ; no
00003EBC 80261704DF                     and     byte [0417h],~20h ; 0DFh
00003EC1 EB0D                           jmp     short set_done
                                not_off:
00003EC3 2E3B06[C251]                   cmp     ax,[cs:OnOff]     ; should we turn it on?
00003EC8 F9                             stc
00003EC9 7505                           jne short set_done ; no
00003ECB 800E170420                     or      byte [0417h],20h
                                set_done:
00003ED0 1F                             pop     ds
                                    ; 04/01/2023
                                    ;pop    ax
00003ED1 C3                             retn

                                ; 16/04/2019 - Retro DOS v4.0

                                ;-----------------------------------------------------------------------------
                                ;
                                ;   menu_check:  check for presence of menu (and other) configuration blocks
                                ;
                                ;   INPUT
                                ;       CX == "organized" config.sys memory image length
                                ;    ES:SI -> "organized" config.sys memory image
                                ;       DS == CS == sysinitseg
                                ;
                                ;   OUTPUT
                                ;       Same as above; the idea is that menu_check simply transforms
                                ;       a block-structured config.sys image into a conventional image,
                                ;       based on the user's block selection and any other boot-time options
                                ;       the user may have employed...
                                ;
                                ;   OTHER REGS USED
                                ;       All
                                ;
                                ;   NOTES
                                ;       [count] and [org_count] are set to the new config.sys image length
                                ;
                                ;   HISTORY
                                ;       Created 16-Mar-1992 by JeffPar
                                ;
                                ;-----------------------------------------------------------------------------

                                    ; 04/01/2023 - Retrodos v4.2 (Modified MSDOS 6.21 IO.SYS)
                                    ; (SYSINIT:4378h)

                                menu_check:

                                ;   Search for SWITCHES, determine if /N or /F are present; if so, then
                                ;   disable clean/interactive boot options

00003ED2 51                             push    cx              ;
00003ED3 56                             push    si              ;
00003ED4 29DB                           sub     bx,bx           ; remains ZERO until first block
                                swchk_loop:                     ;
00003ED6 E83507                         call    get_char        ; get first char of current line
00003ED9 724C                           jc  short swchk_end       ; hit eof
00003EDB 3C5B                           cmp     al,CONFIG_BEGIN ; '['
00003EDD 7503                           jne short swchk_next1 ;
00003EDF 43                             inc     bx              ; remember that we've seen a block
00003EE0 EB40                           jmp     short swchk_nextline
                                swchk_next1:                    ;
00003EE2 3C4E                           cmp     al,CONFIG_NUMLOCK
00003EE4 750E                           jne short swchk_next2 ;
00003EE6 09DB                           or      bx,bx           ; only do NUMLOCK commands that exist
00003EE8 7538                           jnz short swchk_nextline ; before the first block
00003EEA E8C0FF                         call    set_numlock     ; REM it out so we don't act on it later, too
00003EED 26C644FF30                     mov     byte [es:si-1],CONFIG_REM
00003EF2 EB2E                           jmp     short swchk_nextline
                                swchk_next2:                    ;
00003EF4 3C31                           cmp     al,CONFIG_SWITCHES
00003EF6 752A                           jne short swchk_nextline ; this line ain't it
                                swchk_scan:                     ;
00003EF8 E81307                         call    get_char        ; look for /N or /F
                                swchk_scan1:                    ;
00003EFB 3C0A                           cmp     al,LF           ; end of line?
00003EFD 7423                           je  short swchk_nextline ; yes
00003EFF 3C2F                           cmp     al,'/'          ; switch-char?
00003F01 75F5                           jne short swchk_scan ; no
00003F03 E80807                         call    get_char        ;
00003F06 24DF                           and     al,~20h ; 0DFh    ; convert to upper case
00003F08 3A06[6323]                     cmp     al,[swit_n+1]     ; 'N'
00003F0C 7507                           jne short swchk_scan2 ; no
00003F0E 800E[814C]01                   or      byte [bDisableUI],1
00003F13 EBE3                           jmp short swchk_scan ; continue looking for switches of interest
                                swchk_scan2:                    ;
00003F15 3A06[6F23]                     cmp     al,[swit_f+1]     ; 'F'
00003F19 75E0                           jne short swchk_scan1 ; no
00003F1B 800E[814C]02                   or      byte [bDisableUI],2
00003F20 EBD6                           jmp     short swchk_scan ; continue looking for switches of interest
                                swchk_nextline:                 ;
00003F22 E8C306                         call    skip_opt_line   ;
00003F25 EBAF                           jmp     short swchk_loop ;
                                swchk_end:                      ;
00003F27 5E                             pop     si              ;
00003F28 59                             pop     cx              ;

                                ;   Do the keyboard tests for clean/interactive boot now, but only if
                                ;   the DisableUI flag is still clear

00003F29 F606[814C]01                   test    byte [bDisableUI],1
00003F2E 7508                           jnz short menu_search
                                ;
                                ;   Wait for 2 seconds first, UNLESS the /F bit was set in bDisableUI, or
```

```
41512                                   ;   there is anything at all in the keyboard buffer
41513                                   ;
41514 00003F30 E8F2FE                          call    kbd_read
41515 00003F33 7303                            jnc short menu_search
41516 00003F35 E9EE01                          jmp menu_abort
41517
41518                                   ;   Search for MENU block; it is allowed to be anywhere in config.sys
41519
41520                                   menu_search:
41521 00003F38 29DB                            sub     bx,bx           ; if no MENU, default to zero for no_selection
41522 00003F3A BF[C64C]                        mov     di,szMenu ;
41523 00003F3D E80304                          call    find_block      ; find the MENU block
41524 00003F40 7337                            jnc short menu_found ;
41525 00003F42 C606[BE4C]00                    mov     byte [szBoot],0
41526 00003F47 E90C02                          jmp no_selection ; not found
41527
41528                                   ;   Process the requested menu color(s)
41529
41530                                   menu_color:
41531 00003F4A 51                          push    cx              ;
41532 00003F4B 52                          push    dx              ;
41533                                      ;;mov   dx,0007h        ; default color setting
41534                                      ; 10/09/2023
41535                                      ;mov    dl,7 ; !*!
41536 00003F4C E89E06                      call    get_number      ; get first number
41537 00003F4F 80E30F                      and     bl,0Fh  ; !**! ; first # is foreground color (for low nibble)
41538 00003F52 88DD                        mov     ch,bl           ; save it in CH
41539                                      ; 01/08/2023 - Retro DOS v4.2 IO.SYS (optimization) by Erdogan Tan
41540                                      ; (high nibble of dl is 0)
41541                                      ;and    dl,0F0h ; !*!   ; (low nibble of dl would be zero)
41542                                      ;or     dl,bl           ; (low nibble of dl is 7) ! 14/08/2023
41543 00003F54 88DA                        mov     dl,bl   ; 14/08/2023
41544 00003F56 E83108                      call    delim           ; did we hit a delimiter
41545 00003F59 750E                        jne     short check_color ; no, all done
41546 00003F5B E88F06                      call    get_number      ; get next number
41547 00003F5E 80E30F                      and     bl,0Fh          ; second # is background color (for high nibble)
41548 00003F61 88DE                        mov     dh,bl           ; save it in DH
41549                                      ; 10/09/2023
41550                                      ;and    dl,0Fh  ; !**!  ;
41551 00003F63 B104                        mov     cl,4            ;
41552 00003F65 D2E3                        shl     bl,cl           ;
41553 00003F67 08DA                        or      dl,bl           ;
41554                                   check_color:
41555 00003F69 38F5                        cmp     ch,dh           ; are foreground/background the same?
41556 00003F6B 7503                        jne     short set_color       ; no
41557 00003F6D 80F208                      xor     dl,08h          ; yes, so modify the fgnd intensity
41558                                   set_color:
41559 00003F70 8816[7C4C]                  mov     [bMenuColor],dl ;
41560 00003F74 5A                          pop     dx              ;
41561 00003F75 59                          pop     cx              ;
41562 00003F76 E9A900                      jmp     menu_nextitem
41563
41564                                   ;   Back to our regularly scheduled program (the COLOR and other goop
41565                                   ;   above is there simply to alleviate short jump problems)
41566
41567                                   menu_found:
41568 00003F79 C606[864C]01                    mov     byte [bDefBlock],1
41569                                          ;mov       word [offDefBlock],0
41570 00003F7E C606[8A4C]FF                    mov     byte [secTimeOut],-1
41571 00003F83 8026[854C]FD                    and     byte [bQueryOpt],~2 ; 0FDh
41572                                      ; 10/09/2023
41573 00003F88 29D2                        sub     dx,dx
41574 00003F8A 8916[884C]                  mov     [offDefBlock],dx ; 0
41575
41576 00003F8E E85706                          call    skip_opt_line   ; skip to next line
41577                                      ; 10/09/2023
41578                                      ;sub    dx,dx           ; initialize total block count (0 => none yet)
41579
41580                                   ;   Process the menu block now
41581
41582                                   menu_process:
41583 00003F91 E87A06                          call    get_char        ; get first char of current line
41584 00003F94 722E                            jc  short to_menu_getdefault ; could happen if menu block at end (rare)
41585 00003F96 247F                            and     al,~CONFIG_OPTION_QUERY ; 7Fh
41586 00003F98 3C5B                            cmp     al,CONFIG_BEGIN ; BEGIN implies END
41587 00003F9A 7428                            je  short to_menu_getdefault
41588 00003F9C 3C4F                            cmp     al,CONFIG_SUBMENU
41589 00003F9E 744D                            je  short menu_item       ; go process sub-menu
41590 00003FA0 3C45                            cmp     al,CONFIG_MENUITEM
41591 00003FA2 7449                            je  short menu_item       ; go process menu item
41592 00003FA4 3C41                            cmp     al,CONFIG_MENUDEFAULT
41593 00003FA6 741E                            je  short menu_default ; go process menu default
41594 00003FA8 3C52                            cmp     al,CONFIG_MENUCOLOR
41595 00003FAA 749E                            je  short menu_color ; go process menu color
41596 00003FAC 3C4E                            cmp     al,CONFIG_NUMLOCK
41597 00003FAE 740F                            je  short menu_numlock ;
41598 00003FB0 3C30                            cmp     al,CONFIG_REM   ; allow remarks in menu block
41599 00003FB2 746E                            je  short menu_nextitem ;
41600 00003FB4 E8C307                          call    any_delim       ; allow blank lines and such
41601 00003FB7 7469                            je  short menu_nextitem ;
41602 00003FB9 F9                              stc                     ;
41603 00003FBA E82607                          call    print_error     ; non-MENU command!
41604 00003FBD EB63                            jmp     short menu_nextitem
41605                                   menu_numlock:
41606 00003FBF E8EBFE                          call    set_numlock
41607 00003FC2 EB5E                            jmp     short menu_nextitem
41608                                   to_menu_getdefault:
41609 00003FC4 EB62                            jmp     short menu_getdefault
41610
41611                                   ;   Save the offset of the default block name, we'll need it later
41612
41613                                   menu_default:
41614 00003FC6 8936[884C]                      mov     [offDefBlock],si; save address of default block name
41615 00003FCA 803E[8B4C]00                    cmp     byte [secElapsed],0
41616 00003FCF 751A                            jne short timeout_skip ; secElapsed is only zero for the FIRST menu,
41617 00003FD1 E8EA05                          call    skip_token      ; and for subsequent menus IF nothing was typed;
41618 00003FD4 724C                            jc  short menu_nextitem ; secElapsed becomes -1 forever as soon as
41619 00003FD6 E8FB05                          call    skip_delim      ; something is typed
41620 00003FD9 7247                            jc  short menu_nextitem ;
41621 00003FDB 89DE                            mov     si,bx           ;
41622 00003FDD E80D06                          call    get_number      ; get number (of seconds for timeout)
41623 00003FE0 80FB5A                          cmp     bl,90           ; limit it to a reasonable number
41624                                      ;jb     short timeout_ok ; (besides, 99 is the largest # my simple
41625 00003FE3 7602                            jna short timeout_ok ; 01/08/2023
41626 00003FE5 B35A                        mov     bl,90           ;  display function can handle)
41627                                   timeout_ok:                     ;
41628 00003FE7 881E[8A4C]                      mov     [secTimeOut],bl ;
41629                                   timeout_skip:
41630 00003FEB EB35                            jmp     short menu_nextitem
41631
41632                                   ;   Verify that this is a valid menu item by searching for the named block
41633
41634                                   menu_item:
41635                                       ;cmp    dl,9    ; 04/01/2023
```

```
41636 00003FED 80FA09                       cmp     dl,MAX_MULTI_CONFIG ; have we reached the max # of items yet?
41637 00003FF0 7330                         jae short menu_nextitem ;
41638 00003FF2 89F7                         mov     di,si           ; DS:DI -> block name to search for
41639 00003FF4 E83303                       call    srch_block      ;
41640 00003FF7 7406                         je  short menu_itemfound ;
41641 00003FF9 F9                           stc                     ;
41642 00003FFA E8E606                       call    print_error     ; print error and pause
41643 00003FFD EB23                         jmp     short menu_nextitem ; if not found, ignore this menu item
41644
41645                                   ;   srch_block, having succeeded, returns DI -> past the token that it
41646                                   ;   just matched, which in this case should be a descriptive string; ES:SI
41647                                   ;   and CX are unmodified
41648
41649                                   menu_itemfound:
41650 00003FFF 42                           inc     dx              ; otherwise, increment total block count
41651 00004000 89D3                         mov     bx,dx           ; and use it to index the arrays of offsets
41652 00004002 8887[8C4C]                   mov [abBlockType+bx],al
41653 00004006 01DB                         add     bx,bx           ; of recorded block names and descriptions
41654
41655                                   ;   There should be a description immediately following the block name on
41656                                   ;   MENUITEM line; failing that, we'll just use the block name as the
41657                                   ;   description...
41658
41659 00004008 89B7[964C]                   mov     [aoffBlockName+bx],si
41660 0000400C 89B7[AA4C]                   mov     [aoffBlockDesc+bx],si
41661 00004010 89DF                         mov     di,bx           ; skip_delim modifies BX, so stash it in DI
41662 00004012 E8A905                       call    skip_token      ;
41663 00004015 720B                         jc  short menu_nextitem ; hit eol/eof
41664 00004017 E8BA05                       call    skip_delim      ;
41665 0000401A 7206                         jc  short menu_nextitem ; hit eol/eof
41666 0000401C 87FB                         xchg    bx,di           ;
41667 0000401E 89BF[AA4C]                   mov     [aoffBlockDesc+bx],di
41668
41669                                   menu_nextitem:
41670 00004022 E8C305                       call    skip_opt_line   ;
41671 00004025 E969FF                       jmp     menu_process    ; go back for more lines
41672
41673                                   ;   Display menu items now, after determining which one is default
41674
41675                                   menu_getdefault:
41676 00004028 08D2                         or      dl,dl           ; where there any valid blocks at all?
41677 0000402A 7505                         jnz short menu_valid ; yes
41678 0000402C 29DB                         sub     bx,bx           ; no, so force autoselect of 0
41679 0000402E E9ED00                       jmp     menu_autoselect ; (meaning: process common blocks only)
41680                                   menu_valid:
41681 00004031 29DB                         sub     bx,bx           ;
41682 00004033 8816[874C]                   mov     [bMaxBlock],dl  ; first, record how many blocks we found
41683 00004037 8B3E[884C]                   mov     di,[offDefBlock];
41684 0000403B 09FF                         or      di,di           ; does a default block exist?
41685 0000403D 741C                         jz  short menu_nodefault ; no
41686 0000403F 43                           inc     bx              ; yes, walk name table, looking for default
41687                                   menu_chkdefault:
41688 00004040 53                           push    bx              ;
41689 00004041 01DB                         add     bx,bx           ;
41690 00004043 8BB7[964C]                   mov     si,[aoffBlockName+bx]
41691 00004047 B98000                       mov     cx,128          ; arbitrary maximum length of a name
41692 0000404A 1E                           push    ds              ;
41693 0000404B 06                           push    es              ;
41694 0000404C 1F                           pop     ds              ;
41695 0000404D E81A03                       call    comp_names      ; is this block the same as the default?
41696 00004050 1F                           pop     ds              ;
41697 00004051 5B                           pop     bx              ;
41698 00004052 7409                         je  short menu_setdefault ; yes
41699 00004054 43                           inc     bx              ;
41700 00004055 3A1E[874C]                   cmp     bl,[bMaxBlock]  ; all done searching?
41701 00004059 76E5                         jbe short menu_chkdefault ; not yet
41702                                   menu_nodefault:
41703 0000405B B301                         mov     bl,1            ; if no default, force default to #1
41704                                   menu_setdefault:
41705 0000405D 881E[864C]                   mov     [bDefBlock],bl  ; yes, this will be the initial current block
41706
41707                                   ;   If the timeout was explicitly set to 0 (or technically, anything that
41708                                   ;   failed to resolve to a number, like "NONE" or "EAT POTATOES"), then we're
41709                                   ;   supposed to skip menu display and run with the specified default block;
41710                                   ;   however, if the user hit Enter prior to boot, thereby requesting fully
41711                                   ;   INTERACTIVE boot, then we shall display the menu block anyway (though still
41712                                   ;   with no timeout)
41713
41714 00004061 803E[8A4C]00                    cmp     byte [secTimeOut],0 ; is timeout zero? (ie, assume default)
41715 00004066 750A                         jne short menu_display ; no
41716 00004068 F606[854C]01                 test    byte [bQueryOpt],1 ; yes, but was INTERACTIVE requested?
41717 0000406D 7503                         jnz short menu_display ; yes, so *don't* assume default after all
41718 0000406F E9C700                       jmp     not_topmenu       ;
41719
41720                                   ;   Reset the mode, so that we know screen is clean and cursor is home
41721
41722                                   menu_display:
41723 00004072 B40F                         mov     ah,0Fh          ; get current video mode
41724 00004074 CD10                         int     10h             ;
41725 00004076 B400                         mov     ah,00h          ; just re-select that mode
41726 00004078 CD10                         int     10h             ;
41727 0000407A 06                           push    es              ;
41728 0000407B B84000                       mov     ax,40h          ; reach down into the ROM BIOS data area
41729 0000407E 8EC0                         mov     es,ax           ; and save the current (default) video page
41730 00004080 26A14E00                     mov     ax,[es:004Eh]   ; start address and page #, in case the
41731 00004084 A3[834C]                     mov     [wCRTStart],ax  ; undocumented QUIET option was enabled
41732 00004087 26A06200                     mov     al,[es:0062h]   ;
41733 0000408B A2[824C]                     mov     [bCRTPage],al   ;
41734 0000408E A1[7D4C]                     mov     ax,[bMenuPage]    ; select new page for menu
41735 00004091 CD10                         int     10h             ;
41736 00004093 B80006                       mov     ax,0600h        ; clear entire screen
41737 00004096 8A3E[7C4C]                   mov     bh,[bMenuColor] ; using this color
41738 0000409A 29C9                         sub     cx,cx           ; upper left row/col
41739                                       ;mov       dl,[es:CRT_Cols]
41740 0000409C 268A164A00                   mov dl,[es:4Ah]
41741 000040A1 FECA                     dec     dl                  ;
41742                                       ;mov       dh,[es:CRT_Rows];
41743 000040A3 268A368400                   mov dh,[es:84h]
41744 000040A8 08F6                     or      dh,dh           ; # of rows valid?
41745 000040AA 7504                         jnz short menu_clear ; hopefully
41746 000040AC 8A36[804C]                   mov     dh,[bLastRow]   ; no, use a default
41747                                   menu_clear:
41748 000040B0 CD10                         int     10h             ; clear the screen using the req. attribute
41749 000040B2 07                           pop     es              ;
41750 000040B3 8836[804C]                   mov     [bLastRow],dh   ; save DH
41751 000040B7 BA[7752]                     mov     dx,_$MenuHeader
41752 000040BA E89709                       call    print           ; cursor now on row 3 (numbered from 0)
41753
41754 000040BD F606[814C]01                 test    byte [bDisableUI],1
41755 000040C2 751F                         jnz     short menu_nostatus
41756 000040C4 8A3E[7D4C]                   mov     bh,[bMenuPage]  ;
41757 000040C8 8A36[804C]                   mov     dh,[bLastRow]   ; restore DH
41758 000040CC B200                         mov     dl,0            ; print the status line on row DH, col 0,
41759 000040CE B402                         mov     ah,02h          ; now that we can trash the cursor position
```

```
41760 000040D0 CD10                             int     10h             ;
41761 000040D2 BA[C352]                         mov     dx,_$StatusLine
41762 000040D5 E87C09                           call    print           ;
41763 000040D8 B403                             mov     ah,3            ; get cursor position
41764 000040DA CD10                             int     10h             ;
41765 000040DC 80EA02                           sub     dl,2            ;
41766 000040DF 8816[7F4C]                       mov     [bLastCol],dl   ; save column where status char will go
41767
41768                                   menu_nostatus:
41769 000040E3 BB0100                           mov     bx,1            ; now prepare to display all the menu items
41770                                   menu_disploop:
41771 000040E6 E8B002                           call    print_item ; print item #BL
41772 000040E9 43                               inc     bx              ; why "inc bx"? because it's a 1-byte opcode
41773 000040EA 3A1E[874C]                       cmp     bl,[bMaxBlock]  ; all done?
41774 000040EE 76F6                             jbe short menu_disploop ; not yet
41775
41776                                   ;   Set cursor position to just below the menu items
41777
41778 000040F0 B200                              mov     dl,0            ; select column
41779 000040F2 88DE                             mov     dh,bl           ;
41780 000040F4 80C604                           add     dh,4            ; select row below menu
41781 000040F7 8A3E[7D4C]                       mov     bh,[bMenuPage]  ;
41782 000040FB B402                             mov     ah,02h          ; set cursor position beneath the block list
41783 000040FD CD10                             int     10h             ;
41784
41785 000040FF BA[B052]                         mov     dx,_$MenuPrmpt
41786 00004102 E84F09                           call    print           ;
41787 00004105 E82903                           call    select_item     ; make a selection, return # in BX
41788 00004108 BA[7050]                         mov     dx,crlfm
41789 0000410B E84609                           call    print           ;
41790 0000410E FF36[814C]                       push    word [bDisableUI]
41791 00004112 800E[814C]01                     or      byte [bDisableUI],1
41792 00004117 E86704                           call    show_status     ; clear the status line now
41793 0000411A 8F06[814C]                       pop     word [bDisableUI]
41794
41795                                   ;   Now begins the "re-organization" process...
41796
41797                                   menu_autoselect:
41798 0000411E 83FBFF                           cmp     bx,-1 ; 0FFFFh    ; clean boot requested?
41799 00004121 7508                             jne short normal_boot ; no
41800 00004123 E8F105                           call    disable_autoexec; basically, add a /D to the command.com line
41801                                   menu_abort:
41802 00004126 29C9                             sub     cx,cx           ; then immediately exit with 0 config.sys image
41803 00004128 E9E400                           jmp menu_exit      ;
41804
41805                                   normal_boot:
41806 0000412B 83FBFE                           cmp     bx,-2 ; 0FFFEh    ; back to top-level menu?
41807 0000412E 7509                             jne short not_topmenu ; no
41808 00004130 8B0E[5603]                       mov     cx,[count]      ; yes, start all over
41809 00004134 29F6                             sub     si,si           ;
41810 00004136 E9FFFD                           jmp     menu_search
41811
41812                                   not_topmenu:
41813 00004139 80BF[8C4C]4F                     cmp     byte [abBlockType+bx],CONFIG_SUBMENU
41814 0000413E 7510                             jne short not_submenu
41815 00004140 01DB                             add     bx,bx           ;
41816 00004142 8BBF[964C]                       mov     di,[aoffBlockName+bx]
41817 00004146 E8E101                           call    srch_block      ; THIS CANNOT FAIL!
41818 00004149 89FE                             mov     si,di           ;
41819 0000414B 89D9                             mov     cx,bx           ; ES:SI and CX are ready for another round
41820 0000414D E929FE                           jmp     menu_found
41821
41822                                   not_submenu:
41823 00004150 01DB                             add     bx,bx           ; get BX -> name of selected block
41824 00004152 8B9F[964C]                       mov     bx,[aoffBlockName+bx]
41825
41826                                   ;   BX should now either be ZERO (meaning no block has been selected) or
41827                                   ;   the offset relative to ES of the block name to be processed (along with
41828                                   ;   all the "common" lines of course)
41829
41830                                   no_selection:
41831 00004156 891E[884C]                       mov     [offDefBlock],bx; save selection
41832 0000415A 8B0E[5603]                       mov     cx,[count]      ; reset ES:SI and CX for reprocessing
41833 0000415E 29F6                             sub     si,si           ;
41834 00004160 1E                               push    ds              ;
41835 00004161 8E1E[6219]                       mov     ds,[config_wrkseg]; this is where we'll store new config.sys image
41836 00004165 29FF                             sub     di,di           ;
41837
41838                                   ;   ES:SI-> config.sys, DS:DI -> new config.sys workspace
41839                                   ;
41840                                   ;   Work our way through the config.sys image again, this time copying
41841                                   ;   all lines that are (A) "common" lines outside any block or (B) lines
41842                                   ;   within the requested block. Lines inside INCLUDEd blocks are transparently
41843                                   ;   copied by copy_block in a recursive fashion; the amount of recursion is
41844                                   ;   limited by the fact INCLUDE statements are REMed by copy_block as they are
41845                                   ;   processed and by the number of unique INCLUDE stmts in config.sys...
41846                                   ;
41847                                   ;   BUGBUG 20-Mar-1992 JeffPar: If we can figure out the lower bound of the
41848                                   ;   stack we're running on, then we should check it inside copy_block
41849
41850                                   copyblock_loop:
41851 00004167 53                                 push    bx              ; save selected block name
41852 00004168 E82F01                           call    copy_block      ; process (named or common) block
41853 0000416B 5B                               pop     bx              ;
41854 0000416C 7232                             jc  short move_config ; hit eof
41855
41856                                   ;   copy_block can only return for two reasons: it hit eof or a new block
41857
41858                                   copyblock_begin:
41859
41860                                   ; 10/09/2023
41861                                   %if 0
41862                                           push    ax              ;
41863                                           push    cx              ;
41864                                           push    si              ;
41865                                           push    di              ; always do "common" blocks
41866                                           mov     di,szCommon
41867                                           push    ds              ;
41868                                           push    cs              ;
41869                                           pop     ds              ;
41870                                           call    comp_names      ;
41871                                           pop     ds              ;
41872                                           pop     di              ;
41873                                           pop     si              ;
41874                                           pop     cx              ;
41875                                           pop     ax              ;
41876                                           je  short copyblock_check
41877                                   %endif
41878                                       ; 10/09/2023
41879 0000416E 57                             push    di
41880 0000416F BF[CB4C]                       mov     di,szCommon     ; always do "common" blocks
41881 00004172 E81602                         call    comp_names_x    ; (comp_names_safe)
41882 00004175 5F                             pop     di
41883 00004176 740F                           je      short copyblock_check
```

```
41884
41885 00004178 09DB                    or      bx,bx           ; is there a block name to check?
41886 0000417A 7414                    jz  short copyblock_skip ; no
41887 0000417C 57                      push    di              ;
41888 0000417D 89DF                    mov     di,bx           ; check block against given block name
41889 0000417F 1E                      push    ds              ;
41890 00004180 06                      push    es              ;
41891 00004181 1F                      pop     ds              ;
41892 00004182 E8E501                  call    comp_names      ; is this the block we really want to do?
41893 00004185 1F                      pop     ds              ;
41894 00004186 5F                      pop     di              ;
41895                                copyblock_check:
41896 00004187 7217                    jc  short move_config ; hit eof
41897 00004189 7505                    jne short copyblock_skip ;
41898 0000418B E85A04                  call    skip_opt_line   ;
41899 0000418E EBD7                    jmp short copyblock_loop
41900
41901                                copyblock_skip:                 ;
41902 00004190 E85504                  call    skip_opt_line   ; this ain't the block we wanted, so skip it
41903 00004193 E87804                  call    get_char        ;
41904 00004196 7208                    jc  short move_config ; hit eof
41905 00004198 247F                    and     al,~CONFIG_OPTION_QUERY ; 7Fh
41906 0000419A 3C5B                    cmp     al,CONFIG_BEGIN ;
41907 0000419C 74D0                    je  short copyblock_begin
41908 0000419E EBF0                    jmp     short copyblock_skip ; anything else is just skipped
41909                                ;
41910                                ;   To create as little risk to the rest of SysInit as little as possible,
41911                                ;   and to free the workspace at "config_wrkseg" for creating an environment,
41912                                ;   copy the new config.sys image to "confbot"
41913                                ;
41914                                move_config:
41915 000041A0 89F9                    mov     cx,di           ; now copy workspace at DS:DI to "confbot"
41916 000041A2 51                      push    cx              ;
41917                                ;
41918                                ;   But first, copy the CONFIG=<configuration><0> string to the workspace,
41919                                ;   since the configuration name only currently exists in the "confbot" area
41920                                ;
41921                                    ;mov    cx,7
41922 000041A3 B90700                  mov     cx,szMenu-szBoot-1
41923 000041A6 BE[BE4C]                    mov     si,szBoot ; first copy the CONFIG= part
41924 000041A9 47                          inc     di              ; skip a byte, in case absolutely nothing
41925                                                              ; was copied to the workspace, because we always
41926                                                              ; zero the first byte of the workspace (below)
41927                                copy_boot:
41928                                    ;lods    byte ptr cs:[si];
41929 000041AA 2E                          cs
41930 000041AB AC                      lodsb
41931 000041AC 8805                    mov     [di],al         ;
41932 000041AE 47                          inc     di              ;
41933 000041AF E2F9                        loop    copy_boot       ;
41934
41935 000041B1 06                          push    es              ; then copy the configuration name
41936                                      ;mov       cx,128-7       ; put an upper limit on the name, to be safe
41937                                  ; 04/01/2023
41938 000041B2 B179                    mov    cl,128-7
41939 000041B4 2E8B36[884C]            mov     si,[cs:offDefBlock]; ES:SI -> default block name
41940 000041B9 09F6                        or      si,si           ; valid?
41941 000041BB 7505                        jnz short l1       ; yes
41942 000041BD 0E                          push    cs              ;
41943 000041BE 07                          pop     es              ;
41944 000041BF BE[CB4C]                    mov     si,szCommon
41945 000041C2 268A04                  l1:     mov     al,[es:si]      ;
41946 000041C5 E8B205                      call    any_delim       ;
41947 000041C8 7406                        je  short l2        ;
41948 000041CA 8805                        mov     [di],al         ;
41949 000041CC 46                          inc     si              ;
41950 000041CD 47                          inc     di              ;
41951 000041CE E2F2                        loop    l1              ;
41952 000041D0 C6050A                  l2:     mov     byte [di],lf     ; terminate the configuration string
41953 000041D3 07                          pop     es              ;
41954
41955                                ;   Now we can copy "config_wrkseg" (DS) to "confbot" (ES)
41956
41957 000041D4 29FF                        sub     di,di           ;
41958 000041D6 2E893E[6019]                mov     [cs:config_envlen],di
41959 000041DB 29F6                        sub     si,si           ;
41960 000041DD 59                          pop     cx              ; recover the size of "config_wrkseg"
41961
41962 000041DE 51                          push    cx              ;
41963 000041DF F3A4                        rep     movsb           ; moved!
41964 000041E1 59                          pop     cx              ;
41965 000041E2 8CD8                        mov     ax,ds           ;
41966 000041E4 1F                          pop     ds              ;
41967
41968                                ;   Now that the config_wrkseg is available once again, we shall
41969                                ;   use it to create an environment. The first thing to go in will be
41970                                ;   the "CONFIG=configuration" thing. It is also important to zero
41971                                ;   the first byte of the workspace, so that copy_envvar knows the buffer
41972                                ;   is empty.
41973
41974 000041E5 06                          push    es              ;
41975 000041E6 8EC0                        mov     es,ax           ;
41976 000041E8 46                          inc     si              ; ES:SI -> "CONFIG=configuration"
41977 000041E9 26C606000000                mov     byte [es:0],0     ;empty the environment block
41978 000041EF E82600                      call    copy_envvar     ; copy envvar at ES:SI to "config_wrkseg"
41979 000041F2 07                          pop     es
41980
41981                                ;   Before returning, restore the default video page setting but do NOT
41982                                ;   do it using INT 10h's Set Active Page function, because if the menu was
41983                                ;   displayed on a different page, then it's because we don't want to see
41984                                ;   all the device driver/TSR goop (which goes to the default page)
41985
41986                                menu_done:
41987 000041F3 803E[7D4C]00                cmp     byte [bMenuPage],0
41988 000041F8 7415                        je  short menu_exit       ;
41989 000041FA 06                          push    es              ;
41990 000041FB B84000                      mov     ax,40h          ;
41991 000041FE 8EC0                        mov     es,ax           ;
41992 00004200 A1[834C]                    mov     ax,[wCRTStart]  ;
41993 00004203 26A34E00                    mov     [es:004Eh],ax   ;
41994 00004207 A0[824C]                    mov     al,[bCRTPage]   ;
41995 0000420A 26A26200                    mov     [es:0062h],al   ;
41996 0000420E 07                          pop     es              ;
41997                                menu_exit:
41998 0000420F 890E[5603]                  mov     [count],cx      ; set new counts
41999 00004213 890E[5803]                  mov     [org_count],cx  ;
42000                                      ; 10/09/2023 (*) - Erdogan Tan
42001                                  ; MSDOS 6.21 IO.SYS - SYSINIT:46D3h
42002                                  ; PCDOS 7.1 IBMBIO.COM - SYSINIT:491Ah
42003                                  ;sub    si,si           ; always return ES:SI pointing to config.sys
42004 00004217 C3                      retn
42005
42006                                ; (*) NOTE: MSDOS 6.0 source code (SYSINIT2.ASM) contains 'sub si,si' at this
42007                                ;   position (then 'retn' just after it)
```

```
                        ;   but MSDOS 6.21 and PCDOS 7.1 SYSINITs contain only 'retn' here.
                        ;--------------------------------------------------------------------------
                        ;
                        ;   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                   ;
                                ;   jmp     short copy_envloop
                                
                                copy_envdone:                   ;
00004288 28C0                           sub     al,al           ; do SUB to clear carry as well
0000428A AA                             stosb                   ; always null-terminate these puppies
                                copy_envdel:                    ;
0000428B 268805                         mov     [es:di],al      ; and stick another null to terminate the env.
0000428E 2E893E[6019]                   mov     [cs:config_envlen],di
                                    ; 10/09/2023 (x) - Erdogan Tan
                                    stc ; in order to clear carry flag via cmc (compact code trick!)
                                copy_envexit:                   ;
                                    cmc ; (x) ; reverse carry flag status (je -> cf=1)
00004293 F9                             pop     es              ;
00004294 F5                             pop     ds              ;
00004295 07                             pop     si              ;
00004296 1F                             pop     cx              ;
00004297 5E                             pop     dx              ;
00004298 59                             pop     bx              ;
                                
                                copy_done:  ; 18/12/2022
00004299 C3                             retn
                                
                                ;---------------------------------------------------------------------------
                                ;
                                ;   copy_block:  copy the current block to the new config.sys workspace
                                ;
                                ;   INPUT
                                ;       CX == remaining bytes in "organized" config.sys memory image
                                ;    ES:SI -> remaining bytes in "organized" config.sys memory image
                                ;    DS:DI -> new config.sys workspace (equal in size to the original
                                ;             config.sys image) where the current block is to be copied
                                ;
                                ;   OUTPUT
                                ;       Same as above
                                ;       AL also equals the last character read from the organized image
                                ;
                                ;   OTHER REGS USED
                                ;       All
                                ;
                                ;   NOTES
                                ;       None
                                ;
                                ;   HISTORY
                                ;       Created 16-Mar-1992 by JeffPar
                                ;
                                ;---------------------------------------------------------------------------
                                
                                    ; 04/01/2023 - Retrodos v4.2 (Modified MSDOS 6.21 IO.SYS)
                                    ; (SYSINIT:4759h)
                                
                                copy_block:
0000429A E87103                         call    get_char        ; check for include
0000429D 72FA                           jc  short copy_done     ;
0000429F 247F                       and     al,~CONFIG_OPTION_QUERY ; 7Fh
000042A1 3C5B                       cmp     al,CONFIG_BEGIN ; another BEGIN implies END as well
000042A3 74F4                           je  short copy_done ;
                                
000042A5 3C4A                           cmp     al,CONFIG_INCLUDE ; 'J'
000042A7 88E0                           mov     al,ah           ; AL == the original line code
000042A9 753A                           jne short copy_line     ; not an "include" line
                                
                                ;   We have hit an "INCLUDE" line; first, REM out the line so that we
                                ;   never try to include the block again (no infinite include loops please),
                                ;   then search for the named block and call copy_block again.
                                
000042AB 26C644FF30                     mov     byte [es:si-1],CONFIG_REM ; '0'
000042B0 57                             push    di              ;
                                
000042B1 BF[C64C]                       mov     di,szMenu
000042B4 E8D400                         call    comp_names_safe ; don't allow INCLUDE MENU
000042B7 7426                           je  short copy_skip     ;
                                
000042B9 BF[CB4C]                       mov     di,szCommon
000042BC E8CC00                         call    comp_names_safe ; don't allow INCLUDE COMMON
000042BF 741E                           je  short copy_skip     ;
                                
000042C1 89F7                           mov     di,si           ; try to find the block
000042C3 E86400                         call    srch_block      ;
000042C6 89FA                           mov     dx,di           ;
                                    ; 10/09/2023
                                    ;pop    di              ;
000042C8 7514                           jne short copy_error ; no such block
000042CA 5F                         pop     di  ; 10/09/2023
000042CB 51                             push    cx              ;
000042CC 89D9                           mov     cx,bx           ;
000042CE 56                             push    si              ;
000042CF 4A                             dec     dx              ;
000042D0 89D6                           mov     si,dx           ;
000042D2 E80E03                         call    skip_line       ; skip the rest of the "block name" line
000042D5 E8C2FF                         call    copy_block      ; and copy in the rest of that block
000042D8 5E                             pop     si              ;
000042D9 59                             pop     cx              ;
000042DA 28C0                           sub     al,al           ; force skip_opt_line to skip...
000042DC EB2B                           jmp     short copy_nextline
                                
                                copy_error:
                                    ; 10/09/2023
                                    clc
000042DE F8                     copy_skip:
000042DF 5F                             pop     di
                                ;copy_error:
                                    ; 10/09/2023 (cf=0)
                                    ;clc                    ;
000042E0 E80004                         call    print_error     ; note that carry is clear, no pause
000042E3 EB24                           jmp     short copy_nextline
                                
                                ;   Copy the line at ES:SI to the current location at DS:DI
                                
                                copy_line:
000042E5 8805                           mov     [di],al         ;
000042E7 47                             inc     di              ;
000042E8 3C20                           cmp     al,' '          ; is this is a "real" line with a "real" code?
000042EA 721D                           jb  short copy_nextline ; no
000042EC 2E803E[6519]00                 cmp     byte [cs:config_multi],0
000042F2 7409                           je  short copy_loop     ; not a multi-config config.sys, don't embed #s
000042F4 E81700                         call    get_linenum     ; BX == line # of line @ES:SI
000042F7 891D                           mov     [di],bx         ; stash it immediately following the line code
000042F9 47                             inc     di              ;
000042FA 47                             inc     di              ;
000042FB EB08                           jmp     short copy_next ;
                                copy_loop:                      ;
000042FD E80E03                         call    get_char        ;
00004300 7297                           jc  short copy_done ; end of file
00004302 8805                           mov     [di],al         ;
```

```
42256 00004304 47                       inc     di              ;
42257                                 copy_next:
42258 00004305 3C0A                     cmp     al,lf ; 0Ah       ; done with line?
42259 00004307 75F4                     jne short copy_loop       ; nope
42260
42261                                 copy_nextline:
42262 00004309 E8DC02                   call    skip_opt_line   ;
42263 0000430C EB8C                     jmp     short copy_block
42264
42265                                     ; 18/12/2022
42266                                 ;copy_done:
42267                                     ;retn
42268
42269                                 ;--------------------------------------------------------------------------
42270                                 ;
42271                                 ;   get_linenum:  return line # (in BX) of current line (@ES:SI)
42272                                 ;
42273                                 ;   INPUT
42274                                 ;    ES:SI -> some line in the config.sys memory image
42275                                 ;
42276                                 ;   OUTPUT
42277                                 ;       BX == line # (relative to 1)
42278                                 ;
42279                                 ;   OTHER REGS USED
42280                                 ;       DX
42281                                 ;
42282                                 ;   NOTES
42283                                 ;       None
42284                                 ;
42285                                 ;   HISTORY
42286                                 ;       Created 16-Mar-1992 by JeffPar
42287                                 ;
42288                                 ;--------------------------------------------------------------------------
42289
42290                                 get_linenum:
42291 0000430E 50                       push    ax              ;
42292 0000430F 29DB                     sub     bx,bx           ; BX == line # (to be returned)
42293 00004311 51                       push    cx              ;
42294 00004312 89F2                     mov     dx,si           ; DX == the offset we're looking for
42295 00004314 56                       push    si              ;
42296 00004315 2E8B0E[5603]             mov     cx,[cs:count]   ;
42297 0000431A 29F6                     sub     si,si           ; prepare to scan entire file
42298                                 get_linenum_loop:         ;
42299 0000431C E8C402                   call    skip_line       ;
42300 0000431F 7205                     jc  short get_linenum_done
42301 00004321 43                       inc     bx              ;
42302 00004322 39D6                     cmp     si,dx           ; have we exceeded the desired offset yet?
42303 00004324 72F6                     jb  short get_linenum_loop ; no
42304                                 get_linenum_done:         ;
42305 00004326 5E                       pop     si              ;
42306 00004327 59                       pop     cx              ;
42307 00004328 58                       pop     ax              ;
42308 00004329 C3                       retn
42309
42310                                 ;--------------------------------------------------------------------------
42311                                 ;
42312                                 ;   srch_block:  searches entire config.sys for block name @ES:DI
42313                                 ;
42314                                 ;   INPUT
42315                                 ;       ES -> config.sys image
42316                                 ;    ES:DI -> block name to find
42317                                 ;
42318                                 ;   OUTPUT
42319                                 ;       ZF flag set, if found
42320                                 ;    ES:DI -> just past the name in the block heading, if found
42321                                 ;       BX == # bytes remaining from that point, if found
42322                                 ;
42323                                 ;   OTHER REGS USED
42324                                 ;       None
42325                                 ;
42326                                 ;   NOTES
42327                                 ;       This differs from "find_block" in that it searches the ENTIRE
42328                                 ;       config.sys image, not merely the remaining portion, and that it
42329                                 ;       takes a pointer to block name that is *elsewhere* in the image
42330                                 ;       (ie, ES) as opposed to some string constant in our own segment (DS).
42331                                 ;
42332                                 ;   HISTORY
42333                                 ;       Created 16-Mar-1992 by JeffPar
42334                                 ;
42335                                 ;--------------------------------------------------------------------------
42336
42337                                 srch_block:           ; returns BX -> named block in CONFIG.SYS
42338 0000432A 50                       push    ax              ;
42339 0000432B 51                       push    cx              ;
42340 0000432C 2E8B0E[5603]             mov     cx,[cs:count]   ;
42341 00004331 56                       push    si              ;
42342 00004332 29F6                     sub     si,si           ;
42343 00004334 1E                       push    ds              ;
42344 00004335 06                       push    es              ;
42345 00004336 1F                       pop     ds              ;
42346 00004337 E80900                   call    find_block      ;
42347 0000433A 89F7                     mov     di,si           ;
42348 0000433C 89CB                     mov     bx,cx           ;
42349 0000433E 1F                       pop     ds              ;
42350 0000433F 5E                       pop     si              ;
42351 00004340 59                       pop     cx              ;
42352 00004341 58                       pop     ax              ;
42353                                 find_exit: ; 16/04/2019
42354 00004342 C3                       retn                      ;
42355
42356                                 ;--------------------------------------------------------------------------
42357                                 ;
42358                                 ;   find_block:  searches rest of config.sys for block name @DS:DI
42359                                 ;
42360                                 ;   INPUT
42361                                 ;    DS:DI -> block name to find
42362                                 ;    ES:SI -> remainder of config.sys image
42363                                 ;       CX == remaining size of config.sys image
42364                                 ;
42365                                 ;   OUTPUT
42366                                 ;       ZF flag set, if found (also, CF set if EOF)
42367                                 ;    ES:SI -> where the search stopped (at end of block name or EOF)
42368                                 ;       CX == # bytes remaining from that point
42369                                 ;
42370                                 ;   OTHER REGS USED
42371                                 ;       AX
42372                                 ;
42373                                 ;   NOTES
42374                                 ;       This differs from "srch_block" in that it searches only the
42375                                 ;       remaining portion of the config.sys image and leaves SI and CX
42376                                 ;       pointing to where the search left off, and that it takes a pointer
42377                                 ;       to search string in our own segment (DS:DI instead of ES:DI).
42378                                 ;
42379                                 ;   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 86E0                           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 86E0                           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
```

```
                        int     10h             ; DH/DL = row/column
                        push    dx              ; save it
                        mov     ah,02h          ; set cursor position
                        mov     dh,bl           ;
                        add     dh,3            ;
                        mov     dl,5            ;
                        int     10h             ; set cursor position for correct row/col
                        mov     al,bl           ;
                        add     al,'0'          ; convert menu item # to ASCII digit
                        mov     ah,[bMenuColor] ; normal attribute
                        cmp     bl,[bDefBlock]  ; are we printing the current block?
                        jne short print_other ; no
                        or      ah,70h          ; yes, set bgnd color to white
                        mov     ch,ah           ;
                        mov     cl,4            ;
                        rol     ch,cl           ;
                        cmp     ch,ah           ; are fgnd/bgnd the same?
                        jne short print_other ; no
                        xor     ah,08h          ; yes, so modify the fgnd intensity
                print_other:                    ;
                        mov     bh,0            ;
                        add     bx,bx           ;
                        mov     di,[aoffBlockDesc+bx]
                        mov     bl,ah           ; put the attribute in the correct register now
                        mov     bh,[bMenuPage]  ; get correct video page #
                        mov     ah,09h          ; write char/attr
                        mov     cx,1            ;
                        int     10h             ;
                        inc     dl              ; increment column
                        mov     ah,02h          ;
                        int     10h             ;
                        ;mov       ax,0900h+'.'    ;
                        mov ax,092Eh
                    int     10h             ; display '.'
                        inc     dl              ; increment column
                        mov     ah,02h          ;
                        int     10h             ;
                        ;mov       ax,0900h+' '    ;
                        mov ax,0920h
                    int     10h             ; display ' '
                        inc     dl              ; increment column
                        mov     ah,02h          ;
                        int     10h             ;
                        push    es              ;
                print_loop:                     ;
                        mov     al,[es:di]; get a character of the description
                        inc     di              ;
                        cmp     al,TAB ; 9; substitute spaces for tabs
                        jne short print_nontab ;
                        mov     al,' '          ;
                print_nontab:                   ;
                        cmp     al,' '          ;
                        jb  short print_done ; stop at the 1st character < space
                        cmp     al,'$'          ;
                        je  short print_done ; also stop on $
                        mov     ah,09h          ; display function #
                        int     10h             ;
                        inc     dl              ; increment column
                        cmp     dl,78           ; far enough?
                        jae short print_done ; yes
                        mov     ah,02h          ;
                        int     10h             ;
                        jmp short  print_loop
                print_done:                     ;
                        pop     es              ;
                        pop     dx              ;
                        mov     ah,02h          ;
                        int     10h             ; restore previous row/col
                        pop     si              ;
                        pop     dx              ;
                        pop     cx              ;
                        pop     bx              ;
                        pop     ax              ;
                        retn                     ;

                ;---------------------------------------------------------------------------
                ;
                ;   select_item:  wait for user to select menu item, with time-out
                ;
                ;   INPUT
                ;       None
                ;
                ;   OUTPUT
                ;       BX == menu item # (1-N), or -1 for clean boot
                ;       Selected menu item highlighted
                ;       Cursor positioned beneath menu, ready for tty-style output now
                ;
                ;   OTHER REGS USED
                ;       None
                ;
                ;   NOTES
                ;       None
                ;
                ;   HISTORY
                ;       Created 16-Mar-1992 by JeffPar
                ;
                ;---------------------------------------------------------------------------

                select_item:            ; returns digit value in BX (trashes AX/CX/DX)
                        mov     bl,[bDefBlock]  ; BL will be the default block #
                        mov     al,bl           ;
                        call    disp_num        ;
                        call    show_status     ; display current interactive status
                        cmp     byte [secTimeOut],-1
                        je  short input_key       ; no time-out, just go to input
                        mov     ah,GET_TIME ; 2Ch
                        int     21h             ;
                        mov     bh,dh           ; BH = initial # of seconds
                check_time:
                        mov     al,[secTimeOut] ;
                        sub     al,[secElapsed] ;
                        jae short show_time       ;
                        or      byte [bQueryOpt],2  ; disable all further prompting
                        mov     byte [secElapsed],0
                        jmp select_done    ; time's up!
                show_time:
                        push    bx              ;
                        mov     bl,al           ; save # in BL
                        mov     bh,[bMenuPage]  ;
                        mov     ah,03h          ; get cursor position
                        int     10h             ;
                        push    dx              ;
                    add     dl,8            ; move cursor to the right
                        mov     ah,02h          ; set cursor position
```

```
42628 00004471 CD10                     int     10h             ;
42629 00004473 BA[2753]                 mov     dx,_$TimeOut
42630 00004476 E8DB05                   call    print           ; print the "Time remaining: " prompt
42631 00004479 88D8                     mov     al,bl           ; recover # from BL
42632 0000447B 98                       cbw                     ; this works because AL is always <= 90
42633 0000447C B10A                     mov     cl,10           ;
42634 0000447E F6F1                     div     cl              ; AL = tens digit, AH = ones digit
42635 00004480 88E1                     mov     cl,ah           ;
42636 00004482 0430                     add     al,'0'          ;
42637 00004484 B40E                     mov     ah,0Eh          ;
42638 00004486 CD10                     int     10h             ; write TTY tens digit
42639 00004488 88C8                     mov     al,cl           ;
42640 0000448A 0430                     add     al,'0'          ;
42641 0000448C B40E                     mov     ah,0Eh          ;
42642 0000448E CD10                     int     10h             ; write TTY ones digit
42643 00004490 5A                       pop     dx
42644 00004491 B402                     mov     ah,02h          ; set cursor position back to where it was
42645 00004493 CD10                     int     10h             ;
42646 00004495 5B                       pop     bx              ;
42647                           input_key:
42648 00004496 B406                     mov     ah,RAW_CON_IO ; 6
42649 00004498 B2FF                     mov     dl,0FFh         ; input request
42650 0000449A CD21                     int     21h             ;
42651 0000449C 751F                     jnz short got_key  ;
42652 0000449E 803E[8A4C]FF             cmp     byte [secTimeOut],-1; is there a time-out?
42653 000044A3 74F1                     je  short input_key       ; no, just go back to input
42654 000044A5 B42C                     mov     ah,GET_TIME     ;
42655 000044A7 CD21                     int     21h             ; DH = seconds
42656 000044A9 88F4                     mov     ah,dh           ;
42657 000044AB 28FE                     sub     dh,bh           ; should generally be zero or one
42658 000044AD 88E7                     mov     bh,ah           ;
42659 000044AF 7302                     jnc short got_time ;
42660 000044B1 B601                     mov     dh,1            ; it wrapped back to zero, so assume one
42661                           got_time:
42662 000044B3 08F6                     or      dh,dh           ; any change?
42663 000044B5 74DF                     jz  short input_key       ; no
42664 000044B7 0036[8B4C]               add     [secElapsed],dh ;
42665 000044BB EB8D                     jmp short check_time ;
42666                           got_key:
42667 000044BD 50                       push    ax              ;
42668 000044BE B8FFFF                   mov     ax,-1           ; zap both secTimeOut and secElapsed
42669 000044C1 8706[8A4C]               xchg    [secTimeOut],ax
42670 000044C5 3CFF                     cmp     al,-1           ; was time-out already disabled?
42671 000044C7 740E                     je  short timeout_disabled ; yes
42672 000044C9 53                       push    bx              ; let's disable # seconds display
42673 000044CA B8200A                   mov     ax,0A20h        ; write multiple spaces
42674 000044CD 8B1E[7C4C]               mov     bx,[bMenuColor]
42675 000044D1 B95000                   mov     cx,80           ; 80 of them, to be safe
42676 000044D4 CD10                     int     10h             ; to completely obliterate # seconds display
42677 000044D6 5B                       pop     bx               ;
42678
42679                           timeout_disabled:
42680 000044D7 58                       pop     ax              ;
42681 000044D8 08C0                     or      al,al           ; extended key pressed?
42682 000044DA 755A                     jnz short normal_key ; no
42683 000044DC CD21                     int     21h             ; get the next part of the key then
42684 000044DE 74B6                     jz  short input_key       ; hmmm, what happened to the second part?
42685
42686 000044E0 3C48                     cmp     al,48h          ; up arrow?
42687 000044E2 7510                     jne short not_up   ; no
42688 000044E4 80FB01                   cmp     bl,1            ; are we as up as up can get?
42689 000044E7 76AD                     jbe short input_key       ; yes, ignore it
42690 000044E9 FE0E[864C]               dec     byte [bDefBlock] ;
42691 000044ED E8A9FE                   call    print_item      ; re-print the current item
42692 000044F0 FECB                     dec     bl              ; and then print the new current item
42693 000044F2 EB12                     jmp     short print1
42694                           not_up:
42695 000044F4 3C50                     cmp     al,50h          ; down arrow?
42696 000044F6 7518                     jne short not_down ; no
42697 000044F8 3A1E[874C]               cmp     bl,[bMaxBlock]  ; are we as down as down can get?
42698 000044FC 7310                     jae short to_input_key ; yes, ignore it
42699 000044FE FE06[864C]               inc     byte [bDefBlock] ;
42700 00004502 E894FE                   call    print_item      ; re-print the current item
42701 00004505 43                       inc     bx              ; and then print the new current item
42702                           print1:
42703 00004506 88D8                 mov     al,bl           ;
42704                           print2:
42705 00004508 E88EFE               call    print_item      ;
42706 0000450B E86300                   call    disp_num        ;
42707                           to_input_key:
42708 0000450E EB86                     jmp short input_key ; 10/09/2023
42709                           not_down:
42710 00004510 F606[814C]01             test    byte [bDisableUI],1
42711 00004515 75F7                     jnz short to_input_key ; don't allow F8 or F5
42712 00004517 3C42                     cmp     al,42h          ; F8 function key?
42713 00004519 750B                     jne short not_f8   ; no
42714 0000451B 8036[854C]01             xor     byte [bQueryOpt],1
42715 00004520 E85E00                   call    show_status     ;
42716 00004523 E970FF                   jmp     input_key ;
42717                           not_f8:
42718 00004526 3C3F                     cmp     al,3Fh          ; F5 function key?
42719 00004528 75E4                     jne short to_input_key ; no
42720                               ; 02/08/2023 - Retro DOS v4.2 IO.SYS (optimization)
42721                               ; MSDOS 6.21 IO.SYS - SYSINIT:49EBh
42722                               ; (PCDOS 7.1 IBMBIO.COM - SYSINIT:4C32h)
42723 0000452A 800E[854C]04             or      byte [bQueryOpt],4 ; no more queries
42724 0000452F BBFFFF                   mov     bx,-1           ; special return code (-1) indicating clean boot
42725 00004532 B020                     mov     al,' '          ; don't want to display anything really;
42726 00004534 EB26                     jmp     short disp_input ; just want to display the cr/lf sequence...
42727
42728                           normal_key:
42729 00004536 3C0D                     cmp     al,0Dh          ; Enter?
42730 00004538 741C                     je  short select_done ; yes
42731 0000453A 3C08                     cmp     al,08h          ; backspace?
42732 0000453C 7504                     jne short not_backspace ; no
42733 0000453E BBFEFF                   mov     bx,-2 ; 0FFFEh    ; yes, special return code
42734 00004541 C3                       retn                     ;
42735                           not_backspace:
42736 00004542 2C30                     sub     al,'0'          ; is greater than '0'?
42737 00004544 76C8                     jbe short to_input_key ; no
42738 00004546 3A06[874C]               cmp     al,[bMaxBlock]  ; is less than or equal to the maximum digit?
42739 0000454A 77C2                     ja  short to_input_key ; no
42740 0000454C A2[864C]                 mov [bDefBlock],al  ;
42741 0000454F E847FE                   call    print_item      ; redisplay the current selection
42742 00004552 88C3                     mov     bl,al           ; set new selection
42743 00004554 EBB2                     jmp short print2
42744
42745                           select_done:
42746 00004556 B700                     mov     bh,0            ; return a full 16-bit value (for indexing)
42747 00004558 88D8                     mov     al,bl           ;
42748 0000455A 0430                     add     al,'0'          ; convert it into a digit, then display it
42749
42750                               ; fall into disp_input
42751
```

```
                                ; 16/04/2019 - Retro DOS v4.0
                                
                                ;--------------------------------------------------------------------------
                                ;
                                ;   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
```

```
42876                                   ;       CX == remaining bytes in config.sys
42877                                   ;
42878                                   ;   OUTPUT
42879                                   ;       CF set if EOL/EOF hit
42880                                   ;       AL == 1st char of delimiter
42881                                   ;    ES:SI -> just past the delimiter
42882                                   ;       CX == # bytes remaining from that point
42883                                   ;
42884                                   ;   OTHER REGS USED
42885                                   ;       AX
42886                                   ;
42887                                   ;   NOTES
42888                                   ;       None
42889                                   ;
42890                                   ;   HISTORY
42891                                   ;       Created 16-Mar-1992 by JeffPar
42892                                   ;
42893                                   ;--------------------------------------------------------------------------
42894
42895                                   skip_token:
42896 000045BE E84D00                           call    get_char
42897 000045C1 7210                             jc  short skip_token_done
42898 000045C3 E8B401                           call    any_delim
42899 000045C6 75F6                             jne short skip_token
42900                                   skip_check_eol:
42901 000045C8 3C0D                             cmp     al,cr ; 0Dh
42902 000045CA 7406                             je  short skip_token_eol
42903 000045CC 3C0A                             cmp     al,lf ; 0Ah
42904 000045CE 7402                             je  short skip_token_eol
42905 000045D0 F8                               clc
42906                                           ;jmp       short skip_token_done
42907 000045D1 C3                           retn
42908                                   skip_token_eol:
42909 000045D2 F9                               stc
42910                                   skip_token_done:
42911 000045D3 C3                               retn
42912
42913                                   ;--------------------------------------------------------------------------
42914                                   ;
42915                                   ;   skip_delim: advances ES:SI/CX past the current delimiter
42916                                   ;
42917                                   ;   INPUT
42918                                   ;    ES:SI -> position in config.sys
42919                                   ;       CX == remaining bytes in config.sys
42920                                   ;
42921                                   ;   OUTPUT
42922                                   ;       CF set if EOF hit
42923                                   ;       AL == 1st char of token
42924                                   ;    ES:SI -> just past the token
42925                                   ;       CX == # bytes remaining from that point
42926                                   ;    ES:BX -> new token (since ES:SI is already pointing 1 byte past token)
42927                                   ;
42928                                   ;   OTHER REGS USED
42929                                   ;       AX
42930                                   ;
42931                                   ;   NOTES
42932                                   ;       None
42933                                   ;
42934                                   ;   HISTORY
42935                                   ;       Created 16-Mar-1992 by JeffPar
42936                                   ;
42937                                   ;--------------------------------------------------------------------------
42938
42939                                   skip_delim: ; returns carry set if eol/eof
42940 000045D4 E83700                           call    get_char        ;
42941 000045D7 8D5CFF                           lea     bx,[si-1]       ; also returns BX -> next token
42942 000045DA 72F7                             jc  short skip_token_done ;
42943 000045DC E8AB01                           call    delim           ;
42944 000045DF 74F3                             je  short skip_delim ;
42945 000045E1 EBE5                             jmp short skip_check_eol  ; 13/05/2019
42946
42947                                   ;--------------------------------------------------------------------------
42948                                   ;
42949                                   ;   skip_opt_line: same as skip_line provided AL != LF
42950                                   ;
42951                                   ;   INPUT
42952                                   ;       AL == last character read
42953                                   ;    ES:SI -> position in config.sys
42954                                   ;       CX == remaining bytes in config.sys
42955                                   ;
42956                                   ;   OUTPUT
42957                                   ;       CF set if EOF hit
42958                                   ;       AL == 1st char of new line
42959                                   ;    ES:SI -> just past 1st char of new line
42960                                   ;       CX == # bytes remaining from that point
42961                                   ;
42962                                   ;   OTHER REGS USED
42963                                   ;       AX
42964                                   ;
42965                                   ;   NOTES
42966                                   ;       In other words, the purpose here is to skip to the next line,
42967                                   ;       unless ES:SI is already sitting at the front of the next line (which
42968                                   ;       it would be if the last character fetched -- AL -- was a linefeed)
42969                                   ;
42970                                   ;   HISTORY
42971                                   ;       Created 16-Mar-1992 by JeffPar
42972                                   ;
42973                                   ;--------------------------------------------------------------------------
42974
42975                                   ; 03/08/2023 - Retro DOS v4.2 IO.SYS (optimization)
42976                                   ;skip_opt_line:
42977                                   ;   cmp     al,lf ; 0Ah
42978                                   ;   je      short skip_line_done
42979
42980                                       ; fall into skip_line
42981
42982                                   ;--------------------------------------------------------------------------
42983                                   ;
42984                                   ;   skip_line: skip to the next line
42985                                   ;
42986                                   ;   INPUT
42987                                   ;    ES:SI -> position in config.sys
42988                                   ;       CX == remaining bytes in config.sys
42989                                   ;
42990                                   ;   OUTPUT
42991                                   ;       CF set if EOF hit
42992                                   ;    ES:SI -> just past 1st char of new line
42993                                   ;       CX == # bytes remaining from that point
42994                                   ;
42995                                   ;   OTHER REGS USED
42996                                   ;       AX
42997                                   ;
42998                                   ;   NOTES
42999                                   ;       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
                                ;
                                ;-----------------------------------------------------------------------------

                                    ; 31/12/2022 - Retro UNIX 386 v4.2 (Modified MSDOS 6.21 IO.SYS)
                                    ; (SYSINIT:4AE5h)

                                    ; 12/12/2022
                                query_user:
0000461C F606[854C]04                   test    byte [bQueryOpt],4      ; answer no to everything?
                                    ; 01/01/2023
00004621 7403                       jz      short qu_1              ;
00004623 E9B900                     jmp     skip_all
                                    ; 12/12/2022
                                    ;;jmp   short skip_all          ;
                                    ;jnz    short skip_all
                                qu_1:
00004626 F606[854C]02               test    byte [bQueryOpt],2      ; answer yes to everything?
0000462B 75EE                       jnz short nearby_ret            ; yes (and return carry clear!)
0000462D 50                         push    ax                      ;
0000462E A0[6419]                   mov     al,[config_cmd]         ;
00004631 F606[854C]01               test    byte [bQueryOpt],1      ; query every command?
00004636 7506                       jnz short query_all             ; yes
00004638 A880                       test    al,CONFIG_OPTION_QUERY  ;
                                    ; 01/01/2023
0000463A 7502                       jnz     short query_all         ;
                                    ; 12/12/2022
                                    ;;jmp   short do_cmd            ;
                                    ;jz     short do_cmd ; cf=0

                                    ; 01/01/2023
0000463C 58                         pop     ax
0000463D C3                         retn

                                query_all:
                                ;   Search for the command code (AL) in "comtab", and then print
                                ;   out the corresponding keyword, followed by the rest of the actual
                                ;   line pointed to by ES:SI

0000463E 56                             push    si                      ; save pointer to rest of CONFIG.SYS line
0000463F BA[8253]                       mov     dx,_$AutoPrmpt          ;
00004642 247F                           and     al,~CONFIG_OPTION_QUERY ; 7Fh
00004644 7450                           jz  short generic_prompt  ; config_cmd must have been 0

00004646 88C6                           mov     dh,al                   ; save config_cmd in DH
00004648 29DB                           sub     bx,bx                   ;
0000464A BF[D24C]                       mov     di,comtab           ;
                                find_match:                             ;
0000464D 8A1D                           mov     bl,[di]                 ; get size of current keyword
0000464F 08DB                           or      bl,bl                   ;
00004651 7425                           jz  short line_print        ; end of table
00004653 47                             inc     di                      ;
00004654 3A01                           cmp     al,[di+bx]              ; match?
00004656 7405                           je  short cmd_match              ; yes
00004658 8D7901                         lea     di,[di+bx+1]            ; otherwise, skip this command code
                                    ; 13/05/2019
0000465B EBF0                           jmp short find_match        ; loop
                                cmd_match:                              ;
0000465D 8A4DFF                         mov     cl,[di-1]               ;
00004660 B500                           mov ch,0                    ;
00004662 B402                           mov     ah,STD_CON_OUTPUT ; 2
                                cmd_print:                              ;
00004664 8A05                           mov     al,[di]                 ;
00004666 47                             inc     di                      ;
00004667 88C2                           mov     dl,al                   ;
00004669 CD21                           int     21h                     ;
0000466B E2F7                           loop    cmd_print               ;
0000466D B23D                           mov     dl,'='                  ;
0000466F 80FE56                         cmp     dh,CONFIG_SET  ; 'V'    ; for SET commands, don't display a '='
00004672 7502                           jne short cmd_notset        ;
00004674 B220                           mov     dl,' '                  ;
                                cmd_notset:
00004676 CD21                           int     21h                     ; '=' looks funny on SET commands
                                line_print:                             ;
                                    ;lods   byte ptr es:[si]        ;
00004678 26                             es
00004679 AC                         lodsb
0000467A 08C0                       or      al,al                   ;
0000467C 7502                           jnz short non_null          ;
0000467E B020                           mov     al,' '                  ;
                                non_null:                               ;
00004680 3C20                           cmp     al,' '                  ; control code?
00004682 720F                           jb  short prompt_user       ; yes, assume end of line
00004684 7505                           jne short non_space             ;
                                        ; 10/09/2023
00004686 263804                     cmp     [es:si],al ; 20h
                                    ;cmp    byte [es:si],' '        ;
00004689 7208                           jb  short prompt_user       ;
                                non_space:                              ;
0000468B 88C2                           mov     dl,al                   ;
0000468D B402                           mov     ah,STD_CON_OUTPUT ; 2   ;
0000468F CD21                           int     21h                     ;
00004691 EBE5                           jmp short line_print        ;

                                prompt_user:                            ;
00004693 BA[1353]                       mov     dx,_$InterPrmpt         ;

                                generic_prompt:
00004696 E8BB03                         call    print                   ;
                                input_loop:                             ;
00004699 B400                           mov     ah,0                    ; read a key
0000469B CD16                           int     16h                     ;
0000469D 08C0                           or      al,al                   ; is it a function key?
0000469F 750F                           jnz short not_func          ; no
000046A1 80FC3F                         cmp     ah,3Fh                  ; F5 function key?
000046A4 75F3                           jne short input_loop        ; no
000046A6 A0[2353]                       mov     al,[_$NO]               ;
000046A9 800E[854C]04                   or      byte [bQueryOpt],4      ; no more queries
000046AE EB21                           jmp     short legal_char        ;
                                not_func:
000046B0 24DF                           and     al,~20h ; 0DFh          ; converting to upper case
000046B2 3A06[2353]                     cmp     al,[_$NO]           ; verify character is legal
000046B6 7419                           je  short legal_char        ;
000046B8 3A06[1F53]                     cmp     al,[_$YES]          ;
000046BC 7413                           je  short legal_char        ;
000046BE 803E[6419]00                   cmp     byte [config_cmd],0     ;
000046C3 74D4                           je  short input_loop        ; don't allow Esc on this query
000046C5 3C1B                           cmp     al,1Bh                  ; Esc?
000046C7 75D0                           jne short input_loop        ;
000046C9 800E[854C]02                   or      byte [bQueryOpt],2      ; no more interactive boot prompts
000046CE A0[1F53]                       mov     al,[_$YES]
                                legal_char:                             ;
```

```
43248 000046D1 E888FE                       call    disp_input              ;
43249 000046D4 5E                           pop     si                      ; restore pointer to rest of CONFIG.SYS line
43250
43251 000046D5 3A06[2353]                   cmp     al,[_$NO]           ; process line?
43252 000046D9 7403                         je  short skip_cmd          ; no
43253                                   ; 12/12/2022
43254 000046DB F8                           clc
43255                                 do_cmd:
43256 000046DC 58                           pop     ax                  ;
43257                                       ; 12/12/2022
43258                                       ; cf=0
43259                                       ;clc                        ; just do the command
43260 000046DD C3                           retn
43261
43262                                 skip_cmd:
43263 000046DE 58                           pop     ax                  ;
43264                                 skip_all:
43265 000046DF B430                         mov     ah,CONFIG_REM ; '0'   ; fake out the rest of sysinit's processing
43266 000046E1 F9                           stc
43267 000046E2 C3                           retn
43268
43269                                 ;--------------------------------------------------------------------------
43270                                 ;
43271                                 ;   print_error: displays multi-config error conditions
43272                                 ;
43273                                 ;   INPUT
43274                                 ;    Carry set to pause, clear to not
43275                                 ;    ES:SI -> current command line in config.sys
43276                                 ;
43277                                 ;   OUTPUT
43278                                 ;       None
43279                                 ;
43280                                 ;   OTHER REGS USED
43281                                 ;       None
43282                                 ;
43283                                 ;   NOTES
43284                                 ;       None
43285                                 ;
43286                                 ;   HISTORY
43287                                 ;       Created 16-Mar-1992 by JeffPar
43288                                 ;
43289                                 ;--------------------------------------------------------------------------
43290
43291                                 print_error:
43292 000046E3 50                           push    ax
43293 000046E4 53                           push    bx
43294 000046E5 51                           push    cx
43295 000046E6 52                           push    dx
43296 000046E7 1E                           push    ds
43297 000046E8 0E                           push    cs
43298 000046E9 1F                           pop     ds
43299 000046EA 9C                           pushf
43300 000046EB E820FC                       call    get_linenum
43301 000046EE 891E[AF02]                   mov     [linecount],bx
43302 000046F2 E8C5E7                       call    error_line
43303 000046F5 9D                           popf
43304 000046F6 7319                         jnc short pe_ret
43305 000046F8 BA[DD51]                     mov     dx,_$PauseMsg
43306 000046FB E85603                       call    print
43307 000046FE B8070C                       mov     ax,0C07h            ; flush input buffer, then wait for key
43308 00004701 CD21                         int     21h                         ; wait for a key
43309 00004703 08C0                         or      al,al                       ; extended key?
43310 00004705 7504                         jnz short pe_1              ; no
43311 00004707 B407                         mov     ah,07h                      ; yes
43312 00004709 CD21                         int     21h                         ; eat it too
43313                                 pe_1:
43314 0000470B BA[7050]                     mov     dx,crlfm
43315 0000470E E84303                       call    print
43316                                 pe_ret:
43317 00004711 1F                           pop     ds
43318 00004712 5A                           pop     dx
43319 00004713 59                           pop     cx
43320 00004714 5B                           pop     bx
43321 00004715 58                           pop     ax
43322 00004716 C3                           retn
43323
43324                                 ;--------------------------------------------------------------------------
43325
43326                                 ;   This function is very simple: it merely prepends a "/D" to the
43327                                 ;   command-line for the shell; this (undocumented) switch disables
43328                                 ;   AUTOEXEC.BAT processing and the date/time prompt that is usually
43329                                 ;   displayed when there's no AUTOEXEC.BAT.
43330
43331                                 disable_autoexec:
43332                                     ; MSDOS 6.21 IO.SYS -  SYSINIT:4BE2h
43333                                     ; 17/04/2019 - Retro DOS v4.0
43334
43335 00004717 F606[854C]04              test    byte [bQueryOpt],4
43336 0000471C 7443                     jz      short disable_exit
43337 0000471E F606[7B4C]01             test    byte [dae_flag],1
43338 00004723 753C                     jnz     short disable_exit
43339 00004725 800E[7B4C]01             or      byte [dae_flag],1
43340                                       ;or byte [bQueryOpt],2 ; MSDOS 6.0
43341 0000472A 810E[854C]0201           or       word [bQueryOpt],102h ; [bDefBlock] = 1
43342 00004730 BA4420                   mov      dx,'D ' ; 2044h
43343                                 dae_1:
43344                                     ; 03/08/2023 - Retro DOS v4.2 IO.SYS (optimization)
43345 00004733 A0[BA4B]                     mov al,[def_swchr]
43346                                   ;mov    al,[command_line-1]     ; get default switchchar
43347 00004736 08C0                       or      al,al                       ; anything there?
43348 00004738 7427                       jz  short disable_exit    ; no, disable_autoexec already called
43349 0000473A 8A1E[BB4B]                 mov     bl,[command_line]           ;
43350 0000473E B700                       mov     bh,0                        ; BX == command-line length
43351 00004740 89D9                       mov     cx,bx                       ;
43352 00004742 80C303                     add     bl,3                        ;
43353 00004745 80FB7E                     cmp     bl,126                      ;
43354 00004748 7717                       ja  short disable_exit    ;
43355 0000474A 881E[BB4B]                 mov     [command_line],bl       ; update length
43356 0000474E 81C3[BC4B]                 add     bx,command_line+1 ; make sure we move the NULL too
43357 00004752 41                         inc     cx                          ; (just for consistency sake)
43358                                 disable_loop:                           ;
43359 00004753 8A67FD                     mov     ah,[bx-3]                   ;
43360 00004756 8827                       mov     [bx],ah                     ;
43361 00004758 4B                         dec     bx                          ;
43362 00004759 E2F8                       loop    disable_loop                ;
43363 0000475B 8847FE                     mov     [bx-2],al                   ;
43364                                   ;mov    word [bx-1],'D ' ; 2044h ; /D is stuffed into place now
43365 0000475E 8957FF                   mov     [bx-1],dx  ; MSDOS 6.21 IO.SYS - SYSINIT:4C29h
43366                                       ;mov       byte [command_line-1],0 ;
43367                                 disable_exit:                           ;
43368 00004761 C3                         retn
43369
43370                                 CheckQueryOpt:     ; MSDOS 6.21 IO.YSYS - SYSINIT:4C2Dh
43371 00004762 803E[854C]01             cmp     byte [bQueryOpt],1
```

```
00004767 75F8                       jnz     short disable_exit
00004769 F606[7B4C]02               test    byte [dae_flag],2
0000476E 75F1                       jnz     short disable_exit
00004770 800E[7B4C]02               or      byte [dae_flag],2
                                    ;mov    dx,' Y' ; (MASM syntax) ; 2059h
                                    ; 10/09/2023 (BugFix)
00004775 BA5920                     mov     dx,'Y ' ; (NASM syntax) ; 2059h
00004778 EBB9                       jmp     short dae_1

                                ;endif  ;MULTI_CONFIG

                                ;%endif     ; 02/11/2022


                                ; 19/04/2019 - Retro DOS v4.0

                                ;--------------------------------------------------------------------------
                                ;
                                ; procedure : delim
                                ;
                                ;--------------------------------------------------------------------------

                                ; 01/01/2023 - Retro DOS v4.2 (Modified MSDOS 6.21 IO.SYS)
                                ; (SYSINIT:4C45h)

                                ; 02/11/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
                                ;%if 0
                                ;;ifdef     MULTI_CONFIG
                                ;
                                any_delim:
0000477A 3C0D                       cmp     al,cr
0000477C 7427                       je      short delim_ret
0000477E 3C0A                       cmp     al,lf
00004780 7423                       je      short delim_ret
00004782 3C5B                       cmp     al,'['
00004784 741F                       je      short delim_ret
00004786 3C5D                       cmp     al,']'
00004788 741B                       je      short delim_ret
                                ;
                                ;;endif ;MULTI_CONFIG
                                ;%endif ; 02/11/2022

                                    ; 02/11/2022 - Retrodos v4.0 (Modified MSDOS 5.0 IO.SYS)
                                    ; (SYSINIT:3450h)
                                delim:
0000478A 3C2F                       cmp     al,'/'          ; ibm will assume "/" as an delimeter.
0000478C 7417                       je      short delim_ret

0000478E 3C00                       cmp     al,0            ; special case for sysinit!!!
00004790 7413                       je      short delim_ret

                                org_delim:                  ; used by organize routine except for getting
00004792 3C20                       cmp     al,' '          ; the filename.
00004794 740F                       je      short delim_ret
00004796 3C09                           cmp     al,tab ; 9
00004798 740B                       je      short delim_ret
0000479A 3C3D                       cmp     al,'='
0000479C 7407                       je      short delim_ret
0000479E 3C2C                       cmp     al,','
000047A0 7403                       je      short delim_ret
000047A2 3C3B                       cmp     al,';'

                                    ; 02/11/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)

                                ; 04/01/2023 - Retro DOS v4.2
                                ;ifdef     MULTI_CONFIG
                                ;   Make sure there's no chance of a false EOF indication
000047A4 F8                         clc
                                ;endif
                                    ; 02/11/2022
                                delim_ret:
                                    ; 04/01/2023
                                    ; cf = 0
                                nl_ret:     ; 10/09/2023
000047A5 C3                         retn

                                ;--------------------------------------------------------------------------
                                ;
                                ; procedure : newline
                                ;
                                ;  newline returns with first character of next line
                                ;
                                ;--------------------------------------------------------------------------

                                newline:
000047A6 E80600                     call    getchr              ;skip non-control characters
000047A9 72FA                       jc      short nl_ret
000047AB 3C0A                       cmp     al,lf               ;look for line feed
000047AD 75F7                       jne     short newline

                                    ; 10/09/2023 - Retro DOS v4.2 IO.SYS (Optimization)
                                    ;call   getchr
                                ;nl_ret:
                                    ;retn
                                    ; 10/09/2023
                                    ;jmp    short getchr

                                ; 10/09/2023 - Retro DOS v4.2 IO.SYS (Optimization)
                                %if 1

                                ;--------------------------------------------------------------------------
                                ;
                                ; procedure : getchr
                                ;
                                ;--------------------------------------------------------------------------

                                    ; 24/10/2022
                                getchr:
                                    ; 12/12/2022
                                    ;push   cx
                                    ;mov    cx,[count]
                                    ;jcxz   nochar
                                    ; 12/12/2022
000047AF 833E[5603]01               cmp     word [count],1
000047B4 720F                       jb      short nochar ; cf=1 ([count] = 0)

000047B6 8B36[5A03]                 mov     si,[chrptr]
000047BA 268A04                     mov     al,[es:si]
000047BD FF0E[5603]                 dec     word [count]
000047C1 FF06[5A03]                 inc     word [chrptr]
                                    ; 12/12/202
                                    ; cf=0
                                    ;clc
                                ;get_ret:
```

```
43496                                   ;pop    cx
43497                                   ;retn
43498                                 nochar:
43499                                   ; 12/12/2022
43500                                   ; cf=1
43501                                   ;stc
43502                                   ;jmp    short get_ret
43503
43504 000047C5 C3                       retn
43505                                 %endif
43506
43507                                 ;--------------------------------------------------------------------------
43508                                 ;
43509                                 ; procedure : mapcase
43510                                 ;
43511                                 ;--------------------------------------------------------------------------
43512
43513                                   ; 02/11/2022 - Retro DOS 4.0 (Modified MSDOS 5.0 IO.SYS)
43514
43515                                   ; 04/01/2023 - Retro DOS 4.2 (Modified MSDOS 6.21 IO.SYS)
43516                                   ; (SYSINIT:4C7Eh)
43517                                 mapcase:
43518 000047C6 51                       push    cx
43519 000047C7 56                       push    si
43520 000047C8 1E                       push    ds
43521
43522 000047C9 06                       push    es
43523 000047CA 1F                       pop     ds
43524
43525                                   ; 02/11/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
43526
43527                                 ; 04/01/2023 - Retro DOS 4.2
43528
43529                                 ;ifdef      MULTI_CONFIG
43530 000047CB 88C3                     mov     bl,al                   ; same cmd code this line
43531                                 ;else
43532                                 ;   xor     si,si
43533                                 ;endif
43534                                   ; 02/11/2022
43535                                   ; 04/01/2023 - Retro DOS 4.2
43536                                   ;xor    si, si
43537
43538                                 convloop:
43539 000047CD AC                       lodsb
43540 000047CE 3C61                     cmp     al,'a'
43541 000047D0 7209                     jb      short noconv
43542 000047D2 3C7A                     cmp     al,'z'
43543 000047D4 7705                     ja      short noconv
43544 000047D6 2C20                     sub     al,20h
43545 000047D8 8844FF                   mov     [si-1],al
43546                                 noconv:
43547
43548                                   ; 02/11/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
43549
43550                                 ; 04/01/2023 - Retro DOS 4.2
43551                                 ;ifdef      MULTI_CONFIG
43552
43553                                 ;   When MULTI_CONFIG enabled, "mapcase" is used to map everything to
43554                                 ;   upper-case a line at a time, after we've been able to figure out whether
43555                                 ;   the line is a SET command or not (since we don't want to upper-case
43556                                 ;   anything after the "=" in a SET)
43557                                 ;
43558 000047DB 80FB56                     cmp     bl,CONFIG_SET  ; 'V'  ; preserve case for part of the line?
43559 000047DE 7504                     jne     short check_eol             ; no, just check for end-of-line
43560 000047E0 3C3D                     cmp     al,'='                  ; separator between SET var and value?
43561 000047E2 740A                     je      short convdone          ; yes
43562                                 check_eol:
43563 000047E4 3C0D                     cmp     al,cr
43564 000047E6 7406                     je      short convdone
43565 000047E8 3C0A                     cmp     al,lf
43566 000047EA 7402                     je      short convdone
43567                                 ;endif
43568                                   ; 02/11/2022
43569 000047EC E2DF                     loop    convloop
43570                                 convdone:
43571 000047EE 1F                       pop     ds
43572 000047EF 5E                       pop     si
43573 000047F0 59                       pop     cx
43574 000047F1 C3                       retn
43575
43576                                 ;--------------------------------------------------------------------------
43577                                 ;
43578                                 ; procedure : round
43579                                 ;
43580                                 ; round the values in memlo and memhi to paragraph boundary.
43581                                 ; perform bounds check.
43582                                 ;
43583                                 ;--------------------------------------------------------------------------
43584
43585                                 round:
43586                                   ; 10/09/2023 - Retro DOS v4.2 IO.SYS (Optimization)
43587 000047F2 1E                       push    ds
43588 000047F3 0E                       push    cs
43589 000047F4 1F                       pop     ds
43590
43591 000047F5 50                       push    ax
43592                                   ;mov    ax,[cs:memlo]
43593 000047F6 A1[6203]                 mov     ax,[memlo]
43594
43595 000047F9 E819CB                   call    ParaRound               ; para round up
43596
43597                                   ;add    [cs:memhi],ax
43598 000047FC 0106[6403]               add     [memhi],ax
43599                                   ;mov    word [cs:memlo],0
43600 00004800 C706[6203]0000           mov     word [memlo],0
43601                                   ;mov    ax,[cs:memhi]           ; ax = new memhi
43602 00004806 A1[6403]                 mov     ax,[memhi]
43603                                   ;cmp    ax,[cs:ALLOCLIM]        ; if new memhi >= alloclim, error
43604 00004809 3B06[A502]               cmp     ax,[ALLOCLIM]
43605                                   ;jae    short mem_err
43606                                   ; 13/04/2024
43607 0000480D 7322                     jae     short mem_err2 ; ds = cs
43608                                   ;test   byte [cs:setdevmarkflag],for_devmark ; 2
43609 0000480F F606[6919]02             test    byte [setdevmarkflag],for_devmark ; 2
43610 00004814 7416                     jz      short skip_set_devmarksize
43611 00004816 06                       push    es
43612 00004817 56                       push    si
43613                                   ;mov    si,[cs:devmark_addr]
43614 00004818 8B36[6719]               mov     si,[devmark_addr]
43615 0000481C 8EC6                     mov     es,si
43616 0000481E 29F0                     sub     ax,si
43617 00004820 48                       dec     ax
43618                                   ;mov    [es:3],ax
43619 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
0000482A 5E                         pop     si
0000482B 07                         pop     es
                                skip_set_devmarksize:
0000482C 58                         pop     ax

                                    ; 10/09/2023
0000482D 1F                         pop     ds

                                    ; 11/12/2022
                                    ; cf = 0
                                    ; 02/11/2022
                                    ;clc    ; ? (not needed here) ; clear carry
0000482E C3                         retn

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

                                mem_err:
                                    ; 11/12/2022
0000482F 0E                         push    cs
00004830 1F                         pop     ds
                                mem_err2:
00004831 BA[4951]                   mov     dx,badmem
                                    ;push   cs
                                    ;pop    ds
00004834 E81D02                     call    print
00004837 E914CB                     jmp     stall

                                ;--------------------------------------------------------------------------
                                ;
                                ; procedure : calldev
                                ;
                                ;--------------------------------------------------------------------------

                                    ; 02/11/2022 - Retrodos v4.0 (Modified MSDOS 5.0 IO.SYS)
                                    ; (SYSINIT:34E0h)
                                    ; 13/04/2024 - Retrodos v5.0 (Modified PCDOS 7.1 IBMBIO.COM)
                                    ; (PCDOS 7.1 IBMBIO.COM - SYSINIT:4F3Eh)

                                calldev:
0000483A 2E8E1E[3B24]               mov     ds,[cs:DevEntry+2]
0000483F 2E031E[3924]               add     bx,[cs:DevEntry]      ; do a little relocation
00004844 8B07                       mov     ax,[bx]

00004846 2EFF36[3924]               push    word [cs:DevEntry]
0000484B 2EA3[3924]                 mov     [cs:DevEntry],ax
0000484F BB[6F03]                   mov     bx,packet
00004852 2EFF1E[3924]               call    far [cs:DevEntry]
00004857 2E8F06[3924]               pop     word [cs:DevEntry]
0000485C C3                         retn

                                ;--------------------------------------------------------------------------
                                ;
                                ; procedure : todigit
                                ;
                                ;--------------------------------------------------------------------------

                                todigit:
0000485D 2C30                       sub     al,'0'
                                    ;jb     short notdig  ; 02/11/2022
                                    ; 12/12/2022
0000485F 7203                       jb      short notdig2
                                    ;cmp    al,9
                                    ;ja     short notdig
                                    ;clc
                                    ;retn
                                    ; 12/12/2022
00004861 3C0A                       cmp     al,10
00004863 F5                         cmc
                                notdig:
                                    ;stc
                                notdig2:
00004864 C3                         retn

                                ;--------------------------------------------------------------------------
                                ;
                                ; procedure : getnum
                                ;
                                ; getnum parses a decimal number.
                                ; returns it in ax, sets zero flag if ax = 0 (may be considered an
                                ; error), if number is bad carry is set, zero is set, ax=0.
                                ;
                                ;--------------------------------------------------------------------------

                                getnum:
00004865 53                         push    bx
00004866 31DB                       xor     bx,bx                 ; running count is zero
                                b2:
00004868 E8F2FF                     call    todigit               ; do we have a digit ?
0000486B 7247                       jc      short badnum          ; no, bomb

0000486D 93                         xchg    ax,bx                 ; put total in ax
0000486E 53                         push    bx                    ; save digit (0 to 9)
                                    ;mov    bx,10                 ; base of arithmetic
                                    ; 12/12/2022
0000486F B30A                       mov     bl,10
00004871 F7E3                       mul     bx                    ; shift by one decimal digit
00004873 5B                         pop     bx                    ; get back digit (0 to 9)
00004874 00D8                       add     al,bl                 ; get total
00004876 80D400                     adc     ah,0                  ; make that 16 bits
00004879 7239                       jc      short badnum          ; too big a number

0000487B 93                         xchg    ax,bx                 ; stash total

0000487C E830FF                     call    getchr                ;get next digit
0000487F 722D                       jc      short b1              ; no more characters
00004881 3C20                       cmp     al,' '                ; space?
00004883 741F                       je      short b15             ; then end of digits
00004885 3C2C                       cmp     al,','                ; ',' is a seperator!!!
00004887 741B                       je      short b15             ; then end of digits.
00004889 3C09                       cmp     al, tab ; 9           ; tab
0000488B 7417                       je      short b15
0000488D 2E3A06[AE02]               cmp     al,[cs:sepchr]        ; allow 0 or special separators
00004892 7410                       je      short b15
00004894 3C2F                       cmp     al,'/'                ; see if another switch follows
                                    ; 12/12/2022
                                    ; 02/11/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
                                    ;nop                           ; cas - remnant of old bad code
                                    ;nop
00004896 740C                       je      short b15
00004898 3C0A                       cmp     al,lf                 ; line-feed?
0000489A 7408                       je      short b15
```

```
43744 0000489C 3C0D                     cmp     al,cr                   ; carriage return?
43745 0000489E 7404                     je      short b15
43746 000048A0 08C0                     or      al,al                   ; end of line separator?
43747 000048A2 75C4                     jnz     short b2                ; no, try as a valid char...
43748                                   b15:
43749 000048A4 2EFF06[5603]                 inc     word [cs:count]             ; one more character to s...
43750 000048A9 2EFF0E[5A03]                 dec     word [cs:chrptr]        ; back up over separator
43751                                   b1:
43752 000048AE 89D8                         mov     ax,bx                   ; get proper count
43753 000048B0 09C0                         or      ax,ax                   ; clears carry, sets zero accordingly
43754 000048B2 5B                           pop     bx
43755 000048B3 C3                           retn
43756                                   badnum:
43757                                       ; 02/11/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
43758                                       ;mov   byte [cs:sepchr],0
43759 000048B4 31C0                         xor     ax,ax                   ; set zero flag, and ax = 0
43760                                       ; 12 /12/2022
43761 000048B6 2EA2[AE02]                   mov     [cs:sepchr],al ; 0
43762 000048BA 5B                           pop     bx
43763 000048BB F9                           stc                             ; and carry set
43764 000048BC C3                           retn
43765
43766                                   ;******************************************************************************
43767
43768                                   setdoscountryinfo:
43769
43770                                   ;------------------------------------------------------------------------------
43771                                   ;input: es:di -> pointer to dos_country_cdpg_info
43772                                   ;   ds:0  -> buffer.
43773                                   ;   si = 0
43774                                   ;   ax = country id
43775                                   ;   dx = code page id. (if 0, then use ccsyscodepage as a default.)
43776                                   ;   bx = file handle
43777                                   ;   this routine can handle maximum 438 country_data entries.
43778                                   ;
43779                                   ;output: dos_country_cdpg_info set.
43780                                   ;    carry set if any file read failure or wrong information in the file.
43781                                   ;    carry set and cx = -1 if cannot find the matching country_id,
43782                                   ;    codepage_id in the file.
43783                                   ;------------------------------------------------------------------------------
43784
43785                                   ; 04/01/2023 - Retrodos v4.2 (Modified MSDOS 6.21 IO.SYS)
43786                                   ; (SYSINIT:4D83h)
43787
43788                                   ; 10/09/2023 - Retrodos v4.2 IO.SYS (Optimization)
43789                                   ; (PCDOS 7.1 IBMBIO.COM - SYSINIT:4FCAh)
43790
43791 000048BD 57                     push    di
43792 000048BE 50                     push    ax
43793 000048BF 52                     push    dx
43794
43795 000048C0 31C9                   xor     cx,cx
43796 000048C2 31D2                   xor     dx,dx
43797 000048C4 B80002                 mov     ax,512                  ;read 512 bytes
43798 000048C7 E84301                 call    readincontrolbuffer     ;read the file header
43799 000048CA 724A                   jc      short setdosdata_fail
43800
43801 000048CC 06                     push    es
43802 000048CD 56                     push    si
43803
43804 000048CE 0E                     push    cs
43805 000048CF 07                     pop     es
43806
43807 000048D0 BF[204B]               mov     di,country_file_signature ; db 0FFh,'COUNTRY'
43808 000048D3 B90800                 mov     cx,8                    ;length of the signature
43809 000048D6 F3A6                   repz    cmpsb
43810
43811 000048D8 5E                     pop     si
43812 000048D9 07                     pop     es
43813 000048DA 753A                   jnz     short setdosdata_fail ;signature mismatch
43814
43815 000048DC 83C612                 add     si,18                   ;si -> county info type
43816 000048DF 803C01                 cmp     byte [si],1             ;only accept type 1 (currently only 1 header type)
43817 000048E2 7532                   jne     short setdosdata_fail ;cannot proceed. error return
43818
43819 000048E4 46                     inc     si                      ;si -> file offset
43820 000048E5 8B14                   mov     dx,[si]                 ;get the info file offset.
43821 000048E7 8B4C02                 mov     cx,[si+2]
43822 000048EA B80018                 mov     ax,6144                 ;read 6144 bytes.
43823 000048ED E81D01                 call    readincontrolbuffer     ;read info
43824 000048F0 7224                   jc      short setdosdata_fail
43825
43826 000048F2 8B0C                   mov     cx,[si]                 ;get the # of country, codepage combination entries
43827 000048F4 81F9B601               cmp     cx,438                  ;cannot handle more than 438 entries.
43828 000048F8 771C                   ja      short setdosdata_fail
43829
43830 000048FA 46                     inc     si
43831 000048FB 46                     inc     si                      ;si -> entry information packet
43832 000048FC 5A                     pop     dx                      ;restore code page id
43833 000048FD 58                     pop     ax                      ;restore country id
43834 000048FE 5F                     pop     di
43835
43836                                 setdoscntry_find:               ;search for desired country_id,codepage_id.
43837 000048FF 3B4402                     cmp     ax,[si+2]           ;compare country_id
43838 00004902 7509                       jne     short setdoscntry_next
43839
43840                                     ;cmp    dx,0                ;no user specified code page ?
43841                                     ;je     short setdoscntry_any_codepage ;then no need to match code page id.
43842                                     ; 10/09/2023
43843 00004904 09D2                       or      dx,dx ; cmp dx,0
43844 00004906 7413                       jz      short setdoscntry_any_codepage
43845 00004908 3B5404                     cmp     dx,[si+4]           ;compare code page id
43846 0000490B 7411                       je      short setdoscntry_got_it
43847
43848                                 setdoscntry_next:
43849 0000490D 0334                       add     si,[si]             ;next entry
43850 0000490F 46                         inc     si
43851 00004910 46                         inc     si                  ;take a word for size of entry itself
43852 00004911 E2EC                       loop    setdoscntry_find
43853
43854                                     ;mov    cx,-1               ;signals that bad country id entered.
43855                                     ; 10/09/2023
43856 00004913 49                         dec     cx ; 0 -> -1
43857                                 setdoscntry_fail:
43858 00004914 F9                         stc
43859 00004915 C3                         retn
43860
43861                                 setdosdata_fail:
43862 00004916 5E                         pop     si
43863 00004917 59                         pop     cx
43864 00004918 5F                         pop     di
43865 00004919 EBF9                       jmp     short setdoscntry_fail
43866
43867                                 setdoscntry_any_codepage:       ;use the code_page_id of the country_id found.
```

```
43868 0000491B 8B5404                   mov     dx,[si+4]
43869
43870                                   setdoscntry_got_it:                     ;found the matching entry
43871 0000491E 2E8916[284B]             mov     [cs:cntrycodepage_id],dx ;save code page id for this country.
43872 00004923 8B540A                   mov     dx,[si+10]              ;get the file offset of country data
43873 00004926 8B4C0C                   mov     cx,[si+12]
43874 00004929 B80002                   mov     ax,512                  ;read 512 bytes
43875 0000492C E8DE00                   call    readincontrolbuffer
43876 0000492F 72E3                     jc      short setdoscntry_fail
43877
43878 00004931 8B0C                     mov     cx,[si]                 ;get the number of entries to handle.
43879 00004933 46                       inc     si
43880 00004934 46                       inc     si                      ;si -> first entry
43881
43882                                   setdoscntry_data:
43883 00004935 57                       push    di                      ;es:di -> dos_country_cdpg_info
43884 00004936 51                       push    cx                      ;save # of entry left
43885 00004937 56                       push    si                      ;si -> current entry in control buffer
43886
43887 00004938 8A4402                   mov     al,[si+2]               ;get data entry id
43888 0000493B E8A400                   call    getcountrydestination ;get the address of destination in es:di
43889 0000493E 727C                     jc      short setdoscntry_data_next ;no matching data entry id in dos
43890
43891 00004940 8B5404                   mov     dx,[si+4]               ;get offset of data
43892 00004943 8B4C06                   mov     cx,[si+6]
43893 00004946 B80042                   mov     ax,4200h
43894 00004949 F9                       stc
43895 0000494A CD21                     int     21h                     ;move pointer
43896 0000494C 72C8                     jc      short setdosdata_fail
43897
43898 0000494E BA0002                   mov     dx,512                  ;start of data buffer
43899 00004951 B91400                   mov     cx,20                   ;read 20 bytes only. we only need to
43900 00004954 B43F                     mov     ah,3Fh                  ;look at the length of the data in the file.
43901 00004956 F9                       stc
43902 00004957 CD21                     int     21h                     ;read the country.sys data
43903 00004959 72BB                     jc      short setdosdata_fail ;read failure
43904
43905 0000495B 39C8                     cmp     ax,cx
43906 0000495D 75B7                     jne     short setdosdata_fail ; 13/05/2019
43907
43908 0000495F 8B5404                   mov     dx,[si+4]               ;get offset of data again.
43909 00004962 8B4C06                   mov     cx,[si+6]
43910 00004965 B80042                   mov     ax,4200h
43911 00004968 F9                       stc
43912 00004969 CD21                     int     21h                     ;move pointer back again
43913 0000496B 72A9                     jc      short setdosdata_fail
43914
43915 0000496D 56                       push    si
43916 0000496E BE0802                   mov     si,(512+8)              ;get length of the data from the file
43917 00004971 8B0C                     mov     cx,[si]
43918 00004973 5E                       pop     si
43919 00004974 BA0002                   mov     dx,512                  ;start of data buffer
43920 00004977 83C10A                   add     cx,10                   ;signature + a word for the length itself
43921 0000497A B43F                     mov     ah,3Fh                  ;read the data from the file.
43922 0000497C F9                       stc
43923 0000497D CD21                     int     21h
43924 0000497F 7295                     jc      short setdosdata_fail
43925
43926 00004981 39C8                     cmp     ax,cx
43927 00004983 7591                     jne     short setdosdata_fail
43928
43929 00004985 8A4402                   mov     al,[si+2]               ;save data id for future use.
43930 00004988 BE0802                   mov     si,(512+8)              ;si-> data buffer + id tag field
43931 0000498B 8B0C                     mov     cx,[si]                 ;get the length of the file
43932 0000498D 41                       inc     cx                      ;take care of a word for lenght of tab
43933 0000498E 41                       inc     cx                      ;itself.
43934 0000498F 81F9F805                 cmp     cx,(2048-512-8)         ; 1528 ;fit into the buffer?
43935 00004993 7781                     ja      short setdosdata_fail
43936
43937                                   ;if     bugfix
43938 00004995 E83100                   call    setdbcs_before_copy
43939                                   ;endif
43940
43941 00004998 3C01                     cmp     al,SetCountryInfo ; 1 ;is the data for setcountryinfo table?
43942 0000499A 7511                     jne     short setdoscntry_mov ;no, don't worry
43943
43944 0000499C 26FF7518                 push    word [es:di+country_cdpg_info.ccMono_Ptr-country_cdpg_info.ccCountryInfoLen]
43945                                   ;push   word [es:di+24]                 ;cannot destroy ccmono_ptr address. save them.
43946 000049A0 26FF751A                 push    word [es:di+country_cdpg_info.ccMono_Ptr-country_cdpg_info.ccCountryInfoLen+2]
43947                                   ;push   word [es:di+26]                 ;at this time di -> cccountryinfolen
43948
43949 000049A4 57                       push    di                      ;save di
43950
43951                                   ;push   ax
43952                                   ;mov    ax,[cs:cntrycodepage_id] ;do not use the code page info in country_info
43953                                   ;mov    [si+4],ax               ;use the saved one for this !!!!
43954                                   ;pop    ax
43955                                   ; 10/09/2023
43956 000049A5 2EFF36[284B]             push    word [cs:cntrycodepage_id]
43957 000049AA 8F4404                   pop     word [si+4]
43958
43959                                   setdoscntry_mov:
43960 000049AD F3A4                     rep     movsb                   ;copy the table into dos
43961 000049AF 3C01                     cmp     al,SetCountryInfo       ;was the ccmono_ptr saved?
43962 000049B1 7509                     jne     short setdoscntry_data_next
43963
43964 000049B3 5F                       pop     di                      ;restore di
43965 000049B4 268F451A                 pop     word [es:di+country_cdpg_info.ccMono_Ptr-country_cdpg_info.ccCountryInfoLen+2]
43966                                   ;pop    word [es:di+26]                 ;restore
43967 000049B8 268F4518                 pop     word [es:di+country_cdpg_info.ccMono_Ptr-country_cdpg_info.ccCountryInfoLen]
43968                                   ;pop    word [es:di+24]
43969
43970                                   setdoscntry_data_next:
43971 000049BC 5E                       pop     si                      ;restore control buffer pointer
43972 000049BD 59                       pop     cx                      ;restore # of entries left
43973 000049BE 5F                       pop     di                      ;restore pointer to dso_country_cdpg
43974 000049BF 0334                     add     si,[si]                 ;try to get the next entry
43975 000049C1 46                       inc     si
43976 000049C2 46                       inc     si                      ;take a word of entry length itself
43977 000049C3 49                       dec     cx
43978                                   ; 10/09/2023
43979 000049C4 741B                     jz      short setdoscntry_ok
43980                                   ;cmp    cx,0
43981                                   ;je     short setdoscntry_ok
43982 000049C6 E96CFF                   jmp     setdoscntry_data
43983
43984                                   ; 18/12/2022
43985                                   ;setdoscntry_ok:
43986                                   ;retn
43987
43988                                   ;--------------------------------------------------------------------------
43989
43990                                   ;if     bugfix
43991
```

```
                                    ; 10/09/2023 - Retrodos v4.2 IO.SYS (Optimization)

                                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
                                    ;db     "by Erdogan Tan [2024] " ; 01/01/2024
00004AA5 6279204572646F6761-        db      "by Erdogan Tan [2026] " ; 19/01/2026
00004AAE 6E2054616E205B3230-
00004AB7 32365D20
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
```

```
44481
44482                                   ;ifdef ROMEXEC
44483                                   ;   db      7                           ; size of commnd line (excl. null)
44484                                   ;commnd: db "COMMAND",0
44485                                   ;   db      56 dup (0)
44486                                   ;else
44487                                       ; 02/11/2022
44488 00004B2C 0C                         db      12                          ; size of commnd line (excl. null)
44489 00004B2D 5C434F4D4D414E442E-      commnd:     db      "\COMMAND.COM",0
44489 00004B36 434F4D00
44490
44491 00004B3A 00<rep 33h>                ;db     51 dup (0)
44492                                     times   51 db 0
44493                                   ;endif
44494
44495                                   ; 02/11/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
44496 00004B6D 5C434F4D4D414E442E-      ;ifdef      MULTI_CONFIG
44496 00004B76 434F4D00                 commnd2: db         "\COMMAND.COM",0      ; alternate commands to exec,
44497 00004B7A 022F5000                      db     2,"/P",0              ; followed by their respective alternate
44498 00004B7E 5C4D53444F535C434F-      commnd3: db "\MSDOS\COMMAND.COM",0; command lines
44498 00004B87 4D4D414E442E434F4D-
44498 00004B90 00
44499 00004B91 0B413A5C4D53444F53-           db     11,"A:\MSDOS /P",0    ;(the drive letter are dynamically replaced)
44499 00004B9A 202F5000
44500 00004B9E 5C444F535C434F4D4D-      commnd4: db "\DOS\COMMAND.COM",0  ;
44500 00004BA7 414E442E434F4D00
44501 00004BAF 09413A5C444F53202F-           db     9,"A:\DOS /P",0               ;
44501 00004BB8 5000
44502                                   def_swchr:
44503 00004BBA 00                            db     0                      ; default switchchar (referenced as command_line-1)
44504                                   ;endif
44505                                       ; 30/10/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
44506                                   command_line:
44507 00004BBB 022F50                     db      2,"/P"                  ; default command.com args
44508                                     ;db     125 dup (0)
44509 00004BBE 00<rep 7Dh>                times   125 db 0
44510
44511                                   pathstring:
44512                                     ;db     64 dup (0)
44513 00004C3B 00<rep 40h>                times   64 db 0
44514
44515                                   ; 01/01/2023 - Retro DOS v4.2 (Modified MSDOS 6.21 IO.SYS)
44516                                   ; (SYSINIT:51D3h)
44517                                   ; 02/11/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
44518                                   ;%if 0
44519
44520                                   dae_flag:
44521 00004C7B 00                         db      0 ; MSDOS 6.21 IO.SYS - SYSINIT:51D2h
44522
44523                                   ;ifdef      MULTI_CONFIG
44524
44525                                   ; 04/03/2022- Retro DOS v4.2 (Modified MSDOS 6.21 IO.SYS, SYSINIT)
44526                                   MAX_MULTI_CONFIG equ 9    ; max # of multi-config menu items supported
44527
44528                                   ;   Beware of byte pairs accessed as words (see all "KEEP AFTER" notes below)
44529
44530 00004C7C 07                      bMenuColor: db      07h ; 1Fh       ; default fgnd/bgnd color
44531 00004C7D 00                      bMenuPage:  db      0               ; menu video page (KEEP AFTER bMenuColor)
44532 00004C7E 05                                  db      5               ; video page function # (KEEP AFTER bMenuPage)
44533 00004C7F 00                      bLastCol:   db      0               ; ending column on status line
44534 00004C80 18                      bLastRow:   db      24              ; row # of status line (KEEP AFTER bLastCol)
44535 00004C81 00                      bDisableUI: db      0               ; 1=disable clean/interactive
44536                                                                          ; 2=disable default 2-second delay
44537 00004C82 00                      bCRTPage:   db      0               ; value saved from BIOS data area
44538 00004C83 0000                    wCRTStart:  dw      0               ; value saved from BIOS data area
44539 00004C85 00                      bQueryOpt:  db      0               ; 0=off, 1=prompt all, 2=prompt none, 4=skip all
44540 00004C86 01                      bDefBlock:  db      1               ; default block #
44541 00004C87 00                      bMaxBlock:  db      0               ; maxmimum block #
44542 00004C88 0000                    offDefBlock:        dw      0               ; offset of name of default block (if any)
44543 00004C8A FF                      secTimeOut: db      -1 ; 0FFh       ; # of seconds for timeout (-1 == indefinite)
44544 00004C8B 00                      secElapsed: db      0               ; # of seconds elapsed so far (KEEP AFTER secTimeOut)
44545 00004C8C 00<rep Ah>              abBlockType:        times MAX_MULTI_CONFIG+1 db 0 ; array of block types
44546 00004C96 0000<rep Ah>            aoffBlockName:      times MAX_MULTI_CONFIG+1 dw 0 ; array of offsets of block names
44547 00004CAA 0000<rep Ah>            aoffBlockDesc:      times MAX_MULTI_CONFIG+1 dw 0 ; array of offsets of block descriptions
44548
44549 00004CBE 434F4E4649473D00        szBoot:             db      "CONFIG=",0
44550 00004CC6 4D454E5500              szMenu:             db      "MENU",0
44551 00004CCB 434F4D4D4F4E00          szCommon:   db      "COMMON",0
44552
44553                                   ;endif      ;MULTI_CONFIG
44554
44555                                       ; 10/09/2023
44556                                       ; MSDOS 6.21 IO.SYS - SYSINIT:5229h
44557                                       ; (PCDOS 7.1 IBMBIO.COM - SYSINIT:546Eh)
44558
44559                                   comtab:      ; label byte
44560
44561                                   ;           cmd len    command         cmd code
44562                                   ;           -------    -------         --------
44563
44564                                   ;ifdef MULTI_CONFIG
44565 00004CD2 015B5B                             db      1,      "[",            CONFIG_BEGIN
44566                                   ;endif
44567 00004CD5 05425245414B43                     db      5,      "BREAK",        CONFIG_BREAK
44568 00004CDC 074255464645525342                 db      7,      "BUFFERS",      CONFIG_BUFFERS
44569 00004CE5 07434F4D4D454E5459                 db      7,      "COMMENT",      CONFIG_COMMENT
44570 00004CEE 07434F554E54525951                 db      7,      "COUNTRY",      CONFIG_COUNTRY
44571 00004CF7 0644455649434544                   db      6,      "DEVICE",       CONFIG_DEVICE
44572 00004CFF 0A4445564943454849-                db      10,     "DEVICEHIGH",   CONFIG_DEVICEHIGH
44572 00004D08 474855
44573 00004D0B 03444F5348                         db      3,      "DOS",          CONFIG_DOS
44574 00004D10 0844452495650415 24D-              db      8,      "DRIVPARM",     CONFIG_DRIVPARM
44574 00004D19 50
44575 00004D1A 044643425358                       db      4,      "FCBS",         CONFIG_FCBS
44576 00004D20 0546494C455346                     db      5,      "FILES",        CONFIG_FILES
44577                                   ;ifdef MULTI_CONFIG
44578 00004D27 07494E434C5544454A                 db      7,      "INCLUDE",      CONFIG_INCLUDE
44579                                   ;endif
44580 00004D30 07494E5354414C4C49                 db      7,      "INSTALL",      CONFIG_INSTALL
44581 00004D39 0B494E5354414C4C48-                db      11,     "INSTALLHIGH",  CONFIG_INSTALLHIGH
44581 00004D42 49474857
44582 00004D46 094C41535444524956-                db      9,      "LASTDRIVE",    CONFIG_LASTDRIVE
44582 00004D4F 454C
44583                                   ;ifdef MULTI_CONFIG
44584 00004D51 075355424D454E554F                 db      7,      "SUBMENU",      CONFIG_SUBMENU
44585 00004D5A 094D454E55434F4C4F-                db      9,      "MENUCOLOR",    CONFIG_MENUCOLOR
44585 00004D63 5252
44586 00004D65 0B4D454E5544454641-                db      11,     "MENUDEFAULT",  CONFIG_MENUDEFAULT
44586 00004D6E 554C5441
44587 00004D72 084D454E554954454D-                db      8,      "MENUITEM",     CONFIG_MENUITEM
44587 00004D7B 45
44588                                   ;endif
44589 00004D7C 0A4D554C5449545241-                db      10,     "MULTITRACK",   CONFIG_MULTITRACK
44589 00004D85 434B4D
```

```
                                ;ifdef MULTI_CONFIG
00004D88 074E554D4C4F434B4E             db      7,      "NUMLOCK",      CONFIG_NUMLOCK
                                ;endif
00004D91 0352454D30                     db      3,      "REM",          CONFIG_REM
                                ;ifdef MULTI_CONFIG
00004D96 0353455456                     db      3,      "SET",          CONFIG_SET
                                ;endif
00004D9B 055348454C4C53                 db      5,      "SHELL",        CONFIG_SHELL
                                ;if    STACKSW
00004DA2 06535441434B534B               db      6,      "STACKS",       CONFIG_STACKS
                                ;endif
00004DAA 085357495443484553-            db      8,      "SWITCHES",     CONFIG_SWITCHES
00004DB3 31
                                    ; 18/03/2025 (BugFix)
                                    ;db    0

                                    ; 10/09/2023
                                ;aDosdata:  ; PCDOS 7.1 IBMBIO.COM - SYSINIT:5550h
                                    ; 13/04/2024 - Retro DOS v5.0
00004DB4 07444F534441544154         db      7,      "DOSDATA",      CONFIG_DOSDATA ; 'T'
00004DBD 00                         db      0

                                ;%endif ; 02/11/2022

                                ; 01/01/2023 - Retro DOS v4.2
                                %if 0

                                comtab:
                                    ; 02/11/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
                                    ; (SYSINIT:38EDh)
                                    db      7,      "BUFFERS",      CONFIG_BUFFERS
                                    db      5,      "BREAK",        CONFIG_BREAK
                                    db      6,      "DEVICE",       CONFIG_DEVICE
                                    db      10,     "DEVICEHIGH",   CONFIG_DEVICEHIGH
                                    db      5,      "FILES",        CONFIG_FILES
                                    db      4,      "FCBS",         CONFIG_FCBS
                                    db      9,      "LASTDRIVE",    CONFIG_LASTDRIVE
                                    db      10,     "MULTITRACK",   CONFIG_MULTITRACK
                                    db      8,      "DRIVPARM",     CONFIG_DRIVPARM
                                    db      6,      "STACKS",       CONFIG_STACKS
                                    db      7,      "COUNTRY",      CONFIG_COUNTRY
                                    db      5,      "SHELL",        CONFIG_SHELL
                                    db      7,      "INSTALL",      CONFIG_INSTALL
                                    db      7,      "COMMENT",      CONFIG_COMMENT
                                    db      3,      "REM",          CONFIG_REM
                                    db      8,      "SWITCHES",     CONFIG_SWITCHES
                                    db      3,      "DOS",          CONFIG_DOS
                                    db     0

                                %endif

                                ; 01/01/2023 - Retro DOS v4.2 (Modified MSDOS 6.21 IO.SYS)
                                ; (SYSINIT:530Ch)

                                ; 13/04/2024 - Retro DOS v5.0
                                ; (PCDOS 7.1 IBMBIO.COM - SYSINIT:555Ah)

                                deviceparameters:
                                    ; A_DEVICEPARAMETERS <0,dev_3inch720kb,0,80>
                                devp.specialfunc:  ; deviceparameters +
00004DBE 00                         db      0       ; A_DEVICEPARAMETERS.DP_SPECIALFUNCTIONS
                                devp.devtype:
00004DBF 02                         db      2       ; A_DEVICEPARAMETERS.DP_DEVICETYPE
                                devp.devattr:
00004DC0 0000                       dw      0       ; A_DEVICEPARAMETERS.DP_DEVICEATTRIBUTES
                                devp.cylinders:
00004DC2 5000                       dw      80      ; A_DEVICEPARAMETERS.DP_CYLINDERS

                                ; 04/08/2023 - Retro DOS v4.2 IO.SYS (optimization)

                                    ;times  286     db 0
                                devp.mediatype:             ; A_DEVICEPARAMETERS.DP_MEDIATYPE
00004DC4 00                         db      0
                                devp.bpb:           ; A_DEVICEPARAMETERS.DP_BPB
                                devp.bps:           ; A_DEVICEPARAMETERS.DP_BPB+A_BPB.BYTESPERSECTOR
00004DC5 0000                       dw      0
                                devp.secperclus:    ; A_DEVICEPARAMETERS.DP_BPB+A_BPB.SECTORSPERCLUSTER
00004DC7 00                         db      0
00004DC8 0000                       dw      0       ; A_DEVICEPARAMETERS.DP_BPB+A_BPB.RESERVEDSECTORS
00004DCA 00                         db      0       ; A_DEVICEPARAMETERS.DP_BPB+A_BPB.NUMBEROFFATS
00004DCB 0000                       dw      0       ; A_DEVICEPARAMETERS.DP_BPB+A_BPB.ROOTENTRIES
                                devp.totalsecs:             ; A_DEVICEPARAMETERS.DP_BPB+A_BPB.TOTALSECTORS
00004DCD 0000                       dw      0
                                devp.mediaid:               ; A_DEVICEPARAMETERS.DP_BPB+A_BPB.MEDIADESCRIPTOR
00004DCF 00                         db      0
00004DD0 0000                       dw      0       ; A_DEVICEPARAMETERS.DP_BPB+A_BPB.SECTORSPERFAT
                                devp.spt:           ; A_DEVICEPARAMETERS.DP_BPB+A_BPB.SECTORSPERTRACK
00004DD2 0000                       dw      0
                                devp.heads:         ; A_DEVICEPARAMETERS.DP_BPB+A_BPB.HEADS
00004DD4 0000                       dw      0

                                    ; 13/04/2024 - Retro DOS v5.0
                                    ; (PCDOS 7.1 IBMBIO.COM)
00004DD6 00<rep 44h>                times   68 db 0 ; PCDOS 7.1 (FAT32 BPB)
                                    ;;times 14 db 0 ; MSDOS 6.21
                                    ;dw     0       ; A_DEVICEPARAMETERS.DP_BPB+A_BPB.HIDDENSECTORS
                                    ;dw     0
                                    ;dw     0       ; A_DEVICEPARAMETERS.DP_BPB+A_BPB.BIGTOTALSECTORS
                                    ;dw     0
                                    ;times  6 db 0

                                devp.trktblents:
00004E1A 0000                       dw      0       ; A_DEVICEPARAMETERS.DP_TRACKTABLEENTRIES
                                devp.sectbl:                ; A_DEVICEPARAMETERS.DP_SECTORTABLE
00004E1C 00<rep FCh>                times   252 db 0 ; MAX_SECTORS_IN_TRACK * A_SECTORTABLE.size
                                                    ; 63*4 bytes

                                ; 01/01/2023 - Retro DOS v4.2 (Modified MSDOS 6.21 IO.SYS)
                                ; (SYSINIT:5430h)

                                ; 13/04/2024 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMBIO.COM)
                                ; (PCDOS 7.1 IBMBIO.COM - SYSINIT:56B4h)

00004F18 0200                   hlim:       dw      2
00004F1A 0900                   slim:       dw      9

00004F1C 00                     drive:      db      0

                                switches:
00004F1D 0000                       dw      0

                                ; 01/01/2023 - Retro DOS v4.2 (Modified MSDOS 6.21 IO.SYS)
                                ; (SYSINIT:5437h)
```

```
                                ; the following are the recommended bpbs for the media that
                                ; we know of so far.
                                
                                ; 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,'$'
00005205 20627970617373696E-
0000520E 6720796F757220434F-
00005217 4E4649472E53595320-
00005220 616E64204155544F45-
00005229 5845432E4241542066-
00005232 696C65732E0D0A24
                                _$InterMsg:
                                    ;db     'MS-DOS will prompt you to confirm each CONFIG.SYS command.',0Dh,0Ah,'$'
                                    ; 17/12/2023
0000523A 504320444F53207769-        db      'PC DOS will prompt you to confirm each CONFIG.SYS command.',0Dh,0Ah,'$'
00005243 6C6C2070726F6D7074-
0000524C 20796F7520746F2063-
00005255 6F6E6669726D206561-
0000525E 636820434F4E464947-
00005267 2E53595320636F6D6D-
00005270 616E642E0D0A24
                                _$MenuHeader:
00005277 0D0A                       db      0Dh,0Ah
                                    ; 17/12/2023
                                    ;db     '  MS-DOS 6.2 Startup Menu',0Dh,0Ah
                                    ;db     '  '
                                    ;times  23 db (0CDh)  ; ALT 205 ; '=======================' ; 06/08/2023
                                    ;db     0Dh,0Ah,'$'
                                    ; 04/08/2023 (PCDOS 7.10 - IBMBIO.COM SYSINIT:59A7h)
00005279 2020504320444F5320-        db      '  PC DOS 7.1 Startup Menu',0Dh,0Ah
00005282 372E31205374617274-
0000528B 7570204D656E750D0A
00005294 2020                       db      '  '
00005296 CD<rep 17h>                times   23 db (0CDh)  ; ALT 205 ; '=======================' ; 06/08/2023
000052AD 0D0A24                     db      0Dh,0Ah,'$'
                                _$MenuPrmpt:
000052B0 2020456E7465722061-        db      '  Enter a choice: $'
000052B9 2063686F6963653A20-
000052C2 24
                                _$StatusLine:
000052C3 46353D427970617373-        db      'F5=Bypass startup files F8=Confirm each line of CONFIG.SYS '
000052CC 207374617274757020-
000052D5 66696C65732046383D-
000052DE 436F6E6669726D2065-
000052E7 616368206C696E6520-
000052F0 6F6620434F4E464947-
000052F9 2E53595320
000052FE 616E64204155544F45-        db      'and AUTOEXEC.BAT [ ]$'
00005307 5845432E424154205B-
00005310 205D24
                                _$InterPrmpt:
                                    ;db     ' [Y,N]?$'
                                    ; 13/04/2024
                                    ; 04/08/2023
00005313 205B592C4E2C455343-        db      ' [Y,N,ESC]?$' ; PCDOS 7.1 - IBMBIO.COM
0000531C 5D3F24
0000531F 59455324               _$YES:      db      'YES$'
00005323 4E4F2024               _$NO:       db      'NO $'
                                _$TimeOut:
00005327 54696D652072656D61-        db      'Time remaining: $'
00005330 696E696E673A2024
                                badcomprmpt:
                                    ;db     'Enter correct name of Command Interpreter (eg, C:\COMMAND.COM)'
                                    ; 13/04/2024 (PCDOS 7.1 IBMBIO.COM)
00005338 456E74657220636F72-        db      'Enter correct name of Command Interpreter (for example, C:\COMMAND.COM)'
00005341 72656374206E616D65-
0000534A 206F6620436F6D6D61-
00005353 6E6420496E74657270-
0000535C 726574657220028666F-
00005365 72206578616D706C65-
0000536E 2C20433A5C434F4D4D-
00005377 414E442E434F4D29
0000537F 0D0A24                     db      0Dh,0Ah,'$'
                                _$AutoPrmpt:
00005382 50726F636573732041-        db      'Process AUTOEXEC.BAT [Y,N]?$'
0000538B 55544F455845432E42-
00005394 4154205B592C4E5D3F-
0000539D 24

                                ;%endif ; 02/11/2022

                                ; 01/01/2023 - Retro DOS v4.2 (Modified MSDOS 6.21 IO.SYS)
                                ; (SYSINIT:5840h)

                                ; 02/11/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 IO.SYS)
                                ; MSDOS 5.0 IO.SYS - SYSINIT:3CE0h

                                TooManyDrivesMsg:
0000539E 5741524E494E472120-        db      'WARNING! Logical drives past Z: exist and will be ignored',0Dh,0Ah,'$'
000053A7 4C6F676963616C2064-
000053B0 726976657320706173-
000053B9 74205A3A2065786973-
000053C2 7420616E642077696C-
000053CB 6C2062652069676E6F-
000053D4 7265640D0A24

                                ; MSDOS 6.21 IO.SYS - SYSINIT:587Ch
                                    ;db     'Wrong DBLSPACE.BIN version',0Dh,0Ah,'$'
                                    ;db     7 dup(0)

                                    ;times  7 db 0
                                    ; 02/11/2022 (MSDOS 5.0 IO.SYS SYSINIT compatibility)
                                ; MSDOS 5.0 IO.SYS - SYSINIT:3D1Ch
                                    ; 09/12/2022
                                    ;times 4 db 0

                                    ; 08/04/2024 - Retro DOS v5.0
                                ; PCDOS 7.1 IBMBIO.COM - SYSINIT:5B0Bh
                                baddblspace:
000053DA 526571756972656420-        db      'Required system component is not installed',0Dh,0Ah,'$',0
000053E3 73797374656D20636F-
000053EC 6D706F6E656E742069-
000053F5 73206E6F7420696E73-
000053FE 74616C6C65640D0A24-
00005407 00
                                    ;db     7 dup(0)

                                ;--------------------------------------------------------------------------
                                            ; 09/12/2022
                                            ;db 0

                                number3div  equ ($-SYSINIT$)
                                number3mod  equ (number3div % 16)

                                %if (number3mod>0) & (number3mod<16) ; 17/09/2023
00005408 00<rep 8h>                         times (16-number3mod) db 0
                                %endif
```

```
45002
45003                                   ;--------------------------------------------------------------------------
45004                                   ; 09/12/2022 - MSDOS 5.0 IO.SYS:3D20h ;;; SI_end = 3D20h for MSDOS 5.0 IO.SYS
45005                                   ;--------------------------------------------------------------------------
45006
45007                                   ;MSDOS 6.21 IO.SYS - SYSINIT:5899h
45008
45009                                   ;--------------------------------------------------------------------------
45010                                   ; 20/04/2019 - Retro DOS v4.0
45011
45012                                   ; 09/12/2022
45013                                   ;
45014                                   ;bss_start:
45015                                   ;
45016                                   ;ABSOLUTE bss_start
45017                                   ;
45018                                   ;alignb 16
45019
45020                                   SI_end:  ; SI_end equ $
45021
45022                                   ;--------------------------------------------------------------------------
45023
45024                                   ;sysinitseg ends
45025
45026                                   ; ****************************************************************************
45027
45028                                   ; 04/01/2023 - MSDOS 6.21 SYSINIT:SI_end = SYSINIT:58A0h (IOSYS:9F46h)
45029                                   ; 09/12/2022 - MSDOS 5.0 SYSINIT:SI_end = SYSINIT:3D20h
45030
45031                                   SYSINITSIZE equ SI_end - SYSINIT$
45032                                   DOSLOADSEG  equ SYSINITSEG+((SYSINITSIZE+15)/16)
45033
45034                                   ;--------------------------------------------------------------------------
45035                                   ; End of Retro DOS v5.0 IBMBIO.COM (IO.SYS) source by Erdogan Tan (2023)
45036                                   ;--------------------------------------------------------------------------
45037
45038                                   ; 21/12/2022 - Retro DOS v4.0 (Modified MSDOS 5.0)
45039                                   ; 02/10/2023 - Retro DOS v5.0 (Modified PCDOS 7.1)
45040                                   ;--------------------------------------------------------------------------
45041                                   ;--------------------------------------------------------------------------
45042
45043                                   ; ----------------------------------------------------------------------------
45044                                   ; START OF PCDOS 7.1 -IBMDOS.COM- KERNEL CODE (MSDOS.SYS) -will be relocated-
45045                                   ; ----------------------------------------------------------------------------
45046                                   ; 02/10/2023 - Retro DOS v5.0
45047                                   ; 04/01/2023 - Retro DOS v4.2
45048                                   ; 29/12/2022 - Retro DOS v4.1
45049                                   ; 18/03/2019 - Retro DOS v4.0
45050                                   ; 11/06/2018 - Retro DOS v3.0
45051
45052                                   ;MSDOS_BIN_OFFSET:
45053                                   IBMDOS_BIN_OFFSET: ; this offset must be paragraph aligned
45054                                               ;; 28/06/2019 ('msdos6.s')
45055                                               ;incbin 'MSDOS6.BIN' ; Retro DOS 4.0 - MSDOS 6.21 KERNEL
45056
45057                                               ; 29/12/2022
45058                                               ;incbin 'MSDOS51.BIN' ; Retro DOS 4.1 - MSDOS 5.0+ KERNEL
45059
45060                                               ; 29/09/2023 (PARASTART=3DE0h)
45061                                               ; 27/09/2023 (BugFix) ((PARASTART=3DD0h))
45062                                               ; 04/01/2023
45063                                               ;incbin 'MSDOS6.BIN' ; Retro DOS 4.2 - MSDOS 6.21+ KERNEL
45064
45065                                               ; 06/08/2025
45066                                               ; 10/07/2024
45067                                               ; 07/07/2024
45068                                               ; 08/05/2024
45069                                               ; 14/04/2024
45070                                               ; 02/10/2023 - Retro DOS v5.0 - PCDOS 7.1 KERNEL
45071 00005410 <bin A326h>                          incbin  'IBMDOS7.BIN'
45072
45073                                               ;; 28/12/2022 (BugFix)
45074                                               ;; 22/12/2022
45075                                               ;; 21/12/2022 ('msdos5.s')
45076                                               ;incbin 'MSDOS5.BIN'  ; Retro DOS 4.0 - MSDOS 5.0+ KERNEL
45077
45078                                   ; 28/09/2023
45079                                   ;msdos_bin_size equ $ - MSDOS_BIN_OFFSET
45080
45081                                   align 2
45082
45083                                   ; 21/12/2022
45084                                   ;;END_OF_KERNEL:
45085                                   ;END_OF_KERNEL equ $
45086
45087                                   ; 28/09/2023
45088                                   S3SIZE equ $-$$
45089                                   KERNEL_SIZE equ S1SIZE+S2SIZE+S3SIZE
45090
45091                                   ;============================================================================
45092                                   ;   END
45093                                   ;============================================================================
45094                                   ; Retro DOS v5.0 by Erdogan Tan (Redevelopment of PC-DOS 7.1 KERNEL via NASM)
```