```
;******************************************************************************
; IBMDOS7.S (PCDOS 7.1 Kernel) - RETRO DOS v5.0 by ERDOGAN TAN - 01/01/2024
; ----------------------------------------------------------------------------
; Last Update: 06/08/2025 - Retro DOS v5.0 (Modified PCDOS 7.1)
; ----------------------------------------------------------------------------
; Beginning: 22/04/2019 (Retro DOS 4.0), 03/11/2022 (Retro DOS 4.2)
; ----------------------------------------------------------------------------
; Assembler: NASM version 2.15
; ----------------------------------------------------------------------------
;      ((nasm ibmdos7.s -l ibmdos7.txt -o IBMDOS.COM -Z error.txt))
;                                  -o IBMDOS7.BIN
;******************************************************************************
; main file: 'retrodos5.s'
; incbin 'IBMDOS7.BIN'
;==============================================================================
; Modified from 'msdos6.s' (modified MSDOS 6.21 kernel src as Retro DOS v4.2)
; 29/09/2023 /// Retro DOS v4.2 (2023) -> Modified MSDOS 6.22 IO.SYS+MSDOS.SYS
;==============================================================================

; 30/12/2022 - Retro DOS v4.2 Kernel ('msdos6.s')
; Modified from 'msdos5.s' (29/12/2022, Retro DOS v4.1 Kernel) file
; as below:
;   1) MS-DOS version has been changed to 6.22 (It was 5.0)
;   2) Retro DOS version has been changed to 4.2 (It was 4.1)
; (The content has not been changed except kernel version because the kernel
;  code is already compatible with MSDOS 6.x and it is optimized before.)
;   (But IO.SYS part of the kernel is not same with Retro DOS v4.1 code.)

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

; 03/11/2022 - Erdogan Tan (Istanbul)

; Note:    This code is a part of Retro DOS 4.0 kernel source code
;   (as included binary, 'MSDOS5.BIN')
;   Equivalent of MSDOS 5.0 MSDOS.SYS kernel file
;
;   ((MSDOS 6.0 kernel source code has been modified by using disassembled
;   MSDOS 5.0 MSDOS.SYS)) -- Disassembler: HEX-RAYS IDA Pro --
;   ((Disassembly -Reverse engineering- reference: MSDOS 6.0 kernel src))

;------ 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 (4D6h for current version)..
;   SYSINIT code relocates itself and DOSBIOSCODE and MSDOS.SYS
;   (MSDOS5.BIN) according to request/setting in 'config.sys' file.

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

; 01/01/2024 - Retro DOS v5.0 Kernel ('ibmdos7.s')
; Modified from 'msdos6.s' (29/09/2023, Retro DOS v4.2 kernel: MSDOS.SYS) file
; as below:
;
;    1) Retro DOS v5.0 IBMDOS.COM is based on disassembled source code
;   of PCDOS 7.1 IBMDOS.COM (2003) and it is derived using Retro DOS v4.2
;   MSDOS.SYS source code. Retro DOS v5.0 (IBMDOS.COM) kernel source code
;   is modified (and optimized) and so, it is not same with the original
;   PCDOS 7.1 IBMDOS.COM.
;
;    2) Retro DOS v4.2 MSDOS.SYS is based on disassembled source code
;   of MSDOS 6.21 MSDOS.SYS, derived using MSDOS 6.0 source code.
;   (And then it has been verified and updated by comparing it with
;   the disassembled source code of MSDOS 6.22 kernel file MSDOS.SYS.)
;
;    3) 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.)
;
; ('msdos6.s' has been converted to 'ibmdos7.s' and 'retrodos42.s' has been
; converted to 'retrodos5.s'. 'ibmdos7.s' is IBMDOS.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.)

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

;==============================================================================
; Most of comments in this file are from the original MSDOS 6.0 source code
;------------------------------------------------------------------------------

; MSDOS 6.0 Kernel source files:
;   MSDATA.ASM,
;           (MSHEAD.ASM, MSCONST.ASM,CONST2.ASM, MS_DATA.ASM,
;           DOSTAB.ASM, LMSTUB.ASM, WPATCH.INC, MPATCH.ASM)
;   MSTABLE.ASM, MSCODE.ASM, MSDOSME.ASM (DOSMES.INC), TIME.ASM,
;   GETSET.ASM, PARSE.ASM, MISC.ASM, MISC2.ASM, CRIT.ASM, CPMIO.ASM,
;   CPMIO2.ASM, FCBIO.ASM, FCBIO2.ASM, SEARCH.ASM, PATH.ASM, IOCTL.ASM,
;   DELETE.ASM, RENAME.ASM, FINFO.ASM, DUP.ASM, CREATE.ASM, OPEN.ASM,
;   DINFO.ASM, ISEARCH.ASM, BUF.ASM, ABORT.ASM,CLOSE.ASM, DIRCALL.ASM,
;   DISK.ASM, DISK2.ASM, DISK3.ASM, DIR.ASM, DIR2.ASM, DEV.ASM,
;   MKNODE.ASM, ROM.ASM, FCB.ASM, MSCTRLC.ASM, FAT.ASM, MSPROC.ASM
;   ALLOC.ASM, SRVCALL.ASM, UTIL.ASM, MACRO.ASM, MACRO2.ASM, HANDLE.ASM
;   FILE.ASM, LOCK.ASM, ROMFIND.ASM, SHARE.ASM, MSINIT.ASM, ORIGIN.ASM
;
; MSDOS 2.0 Kernel source files:
;   MSDOS.ASM (STDSW.ASM + MSHEAD.ASM + MSDATA.ASM)
;   MSCODE.ASM
;   DOSMES.ASM ... STDIO.ASM, TIME.ASM, XENIX.ASM, XENIX2.ASM

;==============================================================================
; DOSLINK
;==============================================================================
;msdos mscode dosmes misc getset dircall alloc dev dir +
;disk fat rom stdbuf stdcall stdctrlc stdfcb stdproc +
;stdio time xenix xenix2

;==============================================================================
; This MSDOS source code is verified & modified by using IDA Pro Disassembler
; output in TASM syntax (July 2018 -> NASM syntax) [ IBMDOS.COM, 17/03/1987 ]
;==============================================================================
;
; ###########################################################################
; # This file is generated by The Interactive Disassembler (IDA)          #
; # Copyright (c) 2010 by Hex-Rays SA, <support@hex-rays.com>      #
; #                  Licensed to: Freeware version                          #
; ###########################################################################
;
; Input    MD5   : 75959BC417C19135B982F7959EE9C92A

; ----------------------------------------------------------------------------
; File Name   :    C:\Documents and Settings\Erdoğan Tan\Desktop\MSDOS621.BIN
; Format      :    Binary file
```

```
                        ;==============================================================================
                        ; MSDOS621.BIN = MSDOS.SYS, 13/02/1994, 38138 bytes (MSDOS 6.21 kernel) 2019
                        ;------------------------------------------------------------------------------
                        ; MSDOS5.BIN = MSDOS.SYS, 11/11/1991, 37394 bytes (MSDOS 5.0 kernel) 2022

                        ;==============================================================================
                        ; MSDOS.ASM
                        ;==============================================================================

                        ;TITLE   Standard MSDOS
                        ;NAME    MSDOS_2

                        ; Number of disk I/O buffers

                        ;   INCLUDE STDSW.ASM
                        ;       INCLUDE MSHEAD.ASM
                        ;       INCLUDE MSDATA.ASM

                        ;   END

                        ;==============================================================================
                        ; STDSW.ASM
                        ;==============================================================================

                        TRUE    EQU     0FFFFH
                        FALSE   EQU     ~TRUE ; NOT TRUE

                        ; Use the switches below to produce the standard Microsoft version or the IBM
                        ; version of the operating system
                        ;MSVER   EQU       false
                        ;IBM     EQU       true
                        ;WANG    EQU       FALSE
                        ;ALTVECT EQU       FALSE

                        ; Set this switch to cause DOS to move itself to the end of memory
                        ;HIGHMEM EQU     FALSE

                        ;   IF      IBM
                        ESCCH    EQU        0                   ;character to begin escape seq.
                        CANCEL   EQU        27                  ;Cancel with escape
                        TOGLINS  EQU       TRUE                 ;One key toggles insert mode
                        TOGLPRN  EQU       TRUE                 ;One key toggles printer echo
                        ZEROEXT  EQU       TRUE
                        ;       ELSE
                        ;       IF      WANG                    ;Are we assembling for WANG?
                        ;ESCCH       EQU   1FH                  ;Yes. Use 1FH for escape character
                        ;       ELSE
                        ;ESCCH       EQU   1BH
                        ;       ENDIF
                        ;CANCEL  EQU       "X"-"@"               ;Cancel with Ctrl-X
                        ;TOGLINS EQU       WANG                  ;Separate keys for insert mode on
                                                           ;and off if not WANG
                        ;TOGLPRN EQU       FALSE                 ;Separate keys for printer echo on
                                                           ;and off
                        ;ZEROEXT EQU       TRUE
                        ;       ENDIF

                        ;==============================================================================
                        ; MSHEAD.ASM
                        ;==============================================================================

                        ;------------------------------------------------------------
                        ; TITLE   MSHEAD.ASM -- MS-DOS DEFINITIONS
                        ;------------------------------------------------------------

                        ; MS-DOS High-performance operating system for the 8086  version 1.28
                        ;       by Microsoft MSDOS development group:
                        ;          Tim Paterson (Ret.)
                        ;          Aaron Reynolds
                        ;          Nancy Panners (Parenting)
                        ;          Mark Zbikowski
                        ;          Chris Peters (BIOS) (ret.)

                        ; ****************** Revision History ************************
                        ;         >> EVERY change must noted below!! <<
                        ;
                        ; 0.34 12/29/80 General release, updating all past customers
                        ; 0.42 02/25/81 32-byte directory entries added
                        ; 0.56 03/23/81 Variable record and sector sizes
                        ; 0.60 03/27/81 Ctrl-C exit changes, including register save on user stack
                        ; 0.74 04/15/81 Recognize I/O devices with file names
                        ; 0.75 04/17/81 Improve and correct buffer handling
                        ; 0.76 04/23/81 Correct directory size when not 2^N entries
                        ; 0.80 04/27/81 Add console input without echo, Functions 7 & 8
                        ; 1.00 04/28/81 Renumber for general release
                        ; 1.01 05/12/81 Fix bug in `STORE'
                        ; 1.10 07/21/81 Fatal error trapping, NUL device, hidden files, date & time,
                        ;               RENAME fix, general cleanup
                        ; 1.11 09/03/81 Don't set CURRENT BLOCK to 0 on open; fix SET FILE SIZE
                        ; 1.12 10/09/81 Zero high half of CURRENT BLOCK after all (CP/M programs don't)
                        ; 1.13 10/29/81 Fix classic "no write-through" error in buffer handling
                        ; 1.20 12/31/81 Add time to FCB; separate FAT from DPT; Kill SMALLDIR; Add
                        ;               FLUSH and MAPDEV calls; allow disk mapping in DSKCHG; Lots
                        ;               of smaller improvements
                        ; 1.21 01/06/82 HIGHMEM switch to run DOS in high memory
                        ; 1.22 01/12/82 Add VERIFY system call to enable/disable verify after write
                        ; 1.23 02/11/82 Add defaulting to parser; use variable escape character Don't
                        ;               zero extent field in IBM version (back to 1.01!)
                        ; 1.24 03/01/82 Restore fcn. 27 to 1.0 level; add fcn. 28
                        ; 1.25 03/03/82 Put marker (00) at end of directory to speed searches
                        ; 1.26 03/03/82 Directory buffers searched as a circular queue, current buffer
                        ;               is searched first when possible to minimize I/O
                        ;       03/03/82 STORE routine optimized to tack on partial sector tail as
                        ;               full sector write when file is growing
                        ;       03/09/82 Multiple I/O buffers
                        ;       03/29/82 Two bugs:  Delete all case resets search to start at beginning
                        ;               of directory (infinite loop possible otherwise), DSKRESET
                        ;               must invalidate all buffers (disk and directory).
                        ; 1.27 03/31/82 Installable device drivers
                        ;                 Function call 47 - Get pointer to device table list
                        ;                 Function call 48 - Assign CON AUX LIST
                        ;       04/01/82 Spooler interrupt (INT 28) added.
                        ; 1.28 04/15/82 DOS retructured to use ASSUMEs and PROC labels around system
                        ;               call entries.  Most CS relative references changed to SS
                        ;               relative with an eye toward putting a portion of the DOS in
                        ;               ROM.  DOS source also broken into header, data and code pieces
                        ;       04/15/82 GETDMA and GETVECT calls added as 24 and 32.  These calls
                        ;               return the current values.
                        ;       04/15/82 INDOS flag implemented for interrupt processing along with
                        ;               call to return flag location (call 29)
                        ;       04/15/82 Volume ID attribute added
                        ;       04/17/82 Changed ABORT return to user to a long ret from a long jump to
                        ;               avoid a CS relative reference.
                        ;       04/17/82 Put call to STATCHK in dispatcher to catch ^C more often
```

```
249                             ;       04/20/82 Added INT int_upooler into loop ^S wait
250                             ;       04/22/82 Dynamic disk I/O buffer allocation and call to manage them
251                             ;                call 49.
252                             ;       04/23/82 Added GETDSKPTDL as call 50, similar to GETFATPT(DL), returns
253                             ;                address of DPB
254                             ;       04/29/82 Mod to WRTDEV to look for ^C or ^S at console input when
255                             ;                writting to console device via file I/O.  Added a console
256                             ;                output attribute to devices.
257                             ;       04/30/82 Call to en/dis able ^C check in dispatcher Call 51
258                             ;       04/30/82 Code to allow assignment of func 1-12 to disk files as well
259                             ;                as devices....  pipes, redirection now possible
260                             ;       04/30/82 Expanded GETLIST call to 2.0 standard
261                             ;       05/04/82 Change to INT int_fatal_abort callout int HARDERR.  DOS SS
262                             ;                (data segment) stashed in ES, INT int_fatal_abort routines must
263                             ;                preserve ES.  This mod so HARDERR can be ROMed.
264                             ; 1.29 06/01/82 Installable block and character devices as per 2.0 spec
265                             ;       06/04/82 Fixed Bug in CLOSE regarding call to CHKFATWRT.  It got left
266                             ;                out back about 1.27 or so (oops).  ARR
267                             ; 1.30 06/07/82 Directory sector buffering added to main DOS buffer queue
268                             ; 1.40 06/15/82 Tree structured directories.  XENIX Path Parser MKDIR CHDIR
269                             ;                RMDIR Xenix calls
270                             ; 1.41 06/13/82 Made GETBUFFR call PLACEBUF
271                             ; 1.50 06/17/82 FATs cached in buffer pool, get FAT pointer calls disappear
272                             ;                Frees up lots of memory.
273                             ; 1.51 06/24/82 BREAKDOWN modified to do EXACT one sector read/write through
274                             ;                system buffers
275                             ; 1.52 06/30/82 OPEN, CLOSE, READ, WRITE, DUP, DUP2, LSEEK implemented
276                             ; 1.53 07/01/82 OPEN CLOSE mod for Xenix calls, saves and gets remote dir
277                             ; 1.54 07/11/82 Function calls 1-12 make use of new 2.0 PDB. Init code
278                             ;                changed to set file handle environment.
279                             ; 2.00 08/01/82 Number for IBM release
280                             ;       01/19/83 No environ bug in EXEC
281                             ;       01/19/83 MS-DOS OEM INT 21 extensions (SET_OEM_HANDLER)
282                             ;       01/19/83 Performance bug fix in cooked write to NUL
283                             ;       01/27/83 Growcnt fixed for 32-bits
284                             ;       01/27/83 Find-first problem after create
285                             ; 2.01 02/17/83 International DOS
286                             ; 2.11 08/12/83 Dos split into several more modules for assembly on
287                             ;                an IBM PC
288                             ; 08/07/2018 - Retro DOS v3.0 by Erdogan Tan
289                             ; (MSHEAD.ASM, MSDOS 6.0, 1991) - mshead.asm 1.1 85/04/10 -
290                             ; 2.10 03/09/83 Start of NETWORK support
291                             ;          New Buffer structure
292                             ;          New Sytem file table structure
293                             ;          FCB moved to internal representation
294                             ;          DOS re-organized
295                             ; 2.11 04/21/83 Continuation of 2.10, preliminary Network
296                             ;          device interface.
297                             ; 2.11 08/12/83 Dos split into several more modules for assembly on
298                             ;                an IBM PC
299                             ; 2.50 09/12/83 More network stuff
300                             ;
301                             ; ****************************************************************
302
303                             ; --------------------------------------------------------------------------
304                             ; EQUATES
305
306                             ; Interrupt Entry Points:
307
308                             ; INTBASE:      ABORT
309                             ; INTBASE+4:    COMMAND
310                             ; INTBASE+8:    BASE EXIT ADDRESS
311                             ; INTBASE+C:    CONTROL-C ABORT
312                             ; INTBASE+10H:  FATAL ERROR ABORT
313                             ; INTBASE+14H:  BIOS DISK READ
314                             ; INTBASE+18H:  BIOS DISK WRITE
315                             ; INTBASE+1CH:  END BUT STAY RESIDENT (NOT SET BY DOS)
316                             ; INTBASE+20H:  SPOOLER INTERRUPT
317                             ; INTBASE+40H:  Long jump to CALL entry point
318
319                             ENTRYPOINTSEG   EQU     0Ch
320                             MAXDIF          EQU     0FFFh
321                             SAVEXIT         EQU     10
322                             ; 06/05/2019
323                             WRAPOFFSET  EQU    0FEF0h  ; (MISC.ASM, MSDOS 6.0, 1991)
324
325                                    ; INCLUDE DOSSYM.ASM
326                                    ; INCLUDE DEVSYM.ASM
327
328                             ; SUBTTL ^C, terminate/abort/exit and Hard error actions
329                             ; PAGE
330                             ; There are three kinds of context resets that can occur during normal DOS
331                             ; functioning:  ^C trap, terminate/abort/exit, and Hard-disk error.  These must
332                             ; be handles in a clean fashion that allows nested executions along with the
333                             ; ability to trap one's own errors.
334                             ;
335                             ; ^C trap - A process may elect to catch his own ^Cs.  This is achieved by
336                             ;           using the $GET_INTERRUPT_VECTOR and $SET_INTERRUPT_VECTOR as
337                             ;           follows:
338                             ;
339                             ;           $GET_INTERRUPT_VECTOR for INT int_ctrl_c
340                             ;           Save it in static memory.
341                             ;           $SET_INTERRUPT_VECTOR for INT int_ctrl_c
342                             ;
343                             ;           The interrupt service routine must preserve all registers and
344                             ;           return carry set iff the operation is to be aborted (via abort
345                             ;           system call), otherwise, carry is reset and the operation is
346                             ;           restarted.  ANY DEVIATION FROM THIS WILL LEAD TO UNRELIABLE
347                             ;           RESULTS.
348                             ;
349                             ;           To restore original ^C processing (done on terminate/abort/exit),
350                             ;           restore INT int_ctrl_c from the saved vector.
351                             ;
352                             ; Hard-disk error -- The interrupt service routine for INT int_fatal_abort must
353                             ;           also preserve registers and return one of three values in AL: 0 and
354                             ;           1 imply retry and ignore (???)  and 2 indicates an abort.  The user
355                             ;           himself is not to issue the abort, rather, the dos will do it for
356                             ;           him by simulating a normal abort/exit system call.  ANY DEVIATION
357                             ;           FROM THIS WILL LEAD TO UNRELIABLE RESULTS.
358                             ;
359                             ; terminate/abort/exit -- The user may not, under any circumstances trap an
360                             ;           abort call.  This is reserved for knowledgeable system programs.
361                             ;           ANY DEVIATION FROM THIS WILL LEAD TO UNRELIABLE RESULTS.
362
363                             ;SUBTTL SEGMENT DECLARATIONS
364
365                             ; The following are all of the segments used.  They are declared in the order
366                             ; that they should be placed in the executable
367
368                             ;
369                             ; segment ordering for MSDOS
370                             ;
371
372                             ;START           SEGMENT BYTE PUBLIC 'START'
```

```
                        ;START          ENDS
                        ;
                        ;CONSTANTS      SEGMENT BYTE PUBLIC 'CONST'
                        ;CONSTANTS      ENDS
                        ;
                        ;DATA           SEGMENT WORD PUBLIC 'DATA'
                        ;DATA           ENDS
                        ;
                        ;CODE           SEGMENT BYTE PUBLIC 'CODE'
                        ;CODE           ENDS
                        ;
                        ;LAST           SEGMENT BYTE PUBLIC 'LAST'
                        ;LAST           ENDS
                        ;
                        ;DOSGROUP    GROUP   CODE,CONSTANTS,DATA,LAST
                        ;
                        ; The following segment is defined such that the data/const classes appear
                        ; before the code class for ROMification
                        ;
                        ;START            SEGMENT BYTE PUBLIC 'START'
                        ;                 ASSUME  CS:DOSGROUP,DS:NOTHING,ES:NOTHING,SS:NOTHING
                        ;          JMP     DOSINIT
                        ;START            ENDS
                        ;
                        ;============================================================================
                        ; BPB.INC, MSDOS 6.0, 1991
                        ;============================================================================
                        ; 09/07/2018 - Retro DOS v3.0
                        ;
                        ;----+----+----+----+----+----+----+----+----+----+----+----+----+----+----;
                        ;       C A V E A T   P R O G R A M M E R                                   ;
                        ;                                                                           ;
                        ;
                        ;** 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
                        ;
                        ;                                                                           ;
                        ;           C A V E A T    P R O G R A M M E R                            ;
                        ;----+----+----+----+----+----+----+----+----+----+----+----+----+----+----;
                        ;
                        ;============================================================================
                        ; BUFFER.INC, MSDOS 6.0, 1991
                        ;============================================================================
                        ; 04/05/2019 - Retro DOS v4.0
                        ; 03/01/2024 - Retro DOS v5.0
                        ;
                        ; <Disk I/O Buffer Header>
                        ;
                        ;----+----+----+----+----+----+----+----+----+----+----+----+----+----+----;
                        ;       C A V E A T   P R O G R A M M E R                                   ;
                        ;                                                                           ;
                        ;
                        ; Field definition for I/O buffer information
                        ;
                            ; 03/01/2024 - Retro DOS v5.0 (PCDOS 7.1 IBMDOS.COM)
                        ;
                        struc BUFFINFO
```

```
00000000 ????                   .buf_next:  resw 1          ; Pointer to next buffer in list
00000002 ????                   .buf_prev:  resw 1          ; Pointer to prev buffer in list
00000004 ??                     .buf_ID:    resb 1          ; Drive of buffer (bit 7 = 0)
                                                            ; SFT table index (bit 7 = 1)
                                                            ; = FFH if buffer free
00000005 ??                     .buf_flags: resb 1          ; Bit 7 = 1 if Remote file buffer
                                                            ;       = 0 if Local device buffer
                                                            ; Bit 6 = 1 if buffer dirty
                                                            ; Bit 5 = Reserved
                                                            ; Bit 4 = Search bit (bit 7 = 1)
                                                            ; Bit 3 = 1 if buffer is DATA
                                                            ; Bit 2 = 1 if buffer is DIR
                                                            ; Bit 1 = 1 if buffer is FAT
                                                            ; Bit 0 = Reserved
00000006 ????????               .buf_sector:        resd 1          ; Sector number of buffer (flags bit 7 = 0)
                                ; The next two items are often refed as a word (flags bit 7 = 0)
0000000A ??                     .buf_wrtcnt:        resb 1          ; For FAT sectors, # times sector written out
0000000B ????                   .buf_wrtcntinc:     resw 1          ; "   "     "    , # sectors between each write
0000000D ????                               resw 1 ; * ; 03/01/2024 ; PCDOS 7.1
                                                       ; hw of sectors per FAT
0000000F ????????               .buf_DPB:   resd 1          ; Pointer to drive parameters
00000013 ????                   .buf_fill:  resw 1          ; How full buffer is (flags bit 7 = 1)
00000015 ??                     .buf_reserved:      resb 1          ; make DWORD boundary for 386
00000016 ????                               resw 1 ; * ; 03/01/2024 ; PCDOS 7.1
                                                       ; reserved word for dword boundary
                                .size:      ; 20 bytes ; MSDOS 5.0 to 6.22
                                    ; 24 bytes ; PCDOS 7.1 ; 03/01/2024
                                endstruc

                                %define buf_offset BUFFINFO.buf_sector ; 22/07/2019
                                                        ;For buf_flags bit 7 = 1, this is the byte
                                                        ;offset of the start of the buffer in
                                                        ;the file pointed to by buf_ID. Thus
                                                        ;the buffer starts at location
                                                        ;buf_offset in the file and contains
                                                        ;buf_fill bytes.

                                BUFINSIZ        EQU     BUFFINFO.size

                                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

                                ;                                                               ;
                                ;           C  A  V  E  A  T     P  R  O  G  R  A  M  M  E  R    ;
                                ;---+----+----+----+----+----+----+----+----+----+----+----+----+----+----+----;

                                ;===============================================================================
                                ; DOSSSYM.INC, MSDOS 6.0, 1991
                                ;===============================================================================
                                ; 04/05/2019 - Retro DOS v4.0

                                ; <Control character definitions>

                                C_DEL           EQU     7Fh         ;    ASCII rubout or delete previous char
                                C_BS    EQU      08h        ; ^H ASCII backspace
                                C_CR    EQU      0Dh        ; ^M ASCII carriage return
                                C_LF    EQU      0Ah        ; ^J ASCII linefeed
                                C_ETB           EQU     17h         ; ^W ASCII end of transmission
                                C_NAK           EQU     15h         ; ^U ASCII negative acknowledge
                                C_ETX           EQU     03h         ; ^C ASCII end of text
                                C_HT    EQU      09h        ; ^I ASCII tab

                                ; <User stack inside of system call>
                                ; Location of user registers relative user stack pointer

                                struc       user_env    ; user_environ
00000000 ????                   .user_AX: resw 1
00000002 ????                   .user_BX: resw 1
00000004 ????                   .user_CX: resw 1
00000006 ????                   .user_DX: resw 1
00000008 ????                   .user_SI: resw 1
0000000A ????                   .user_DI: resw 1
0000000C ????                   .user_BP: resw 1
0000000E ????                   .user_DS: resw 1
00000010 ????                   .user_ES: resw 1
00000012 ????                   .user_IP: resw 1
00000014 ????                   .user_CS: resw 1
00000016 ????                   .user_F:  resw 1
                                .size:
                                endstruc

                                ; ---- <Disk map> ----

                                ;   MSDOS partitions the disk into 4 sections:
                                ;
                                ;  phys sector 0:   +-------------------+
                                ;    |              | boot/reserved     |
                                ;    |              +-------------------+
                                ;    |              |  File allocation  |
                                ;    v              |      table(s)     |
                                ;                   |  (multiple copies |
                                ;                   |     are kept)     |
                                ;                   +-------------------+
                                ;                   |     Directory     |
                                ;                   +-------------------+
                                ;                   |     File space    |
                                ;                   +-------------------+
                                ;                   |   Unaddressable   |
                                ;                   |  (to end of disk) |
                                ;                   +-------------------+
                                ;
                                ; All partition boundaries are sector boundaries. The size of the FAT is
                                ; adjusted to maximize the file space addressable.

                                ; <File allocation Table information>

                                ; The File Allocation Table uses a 12-bit entry for each allocation unit on
                                ; the disk. These entries are packed, two for every three bytes. The contents
                                ; of entry number N is found by 1) multiplying N by 1.5; 2) adding the result
                                ; to the base address of the Allocation Table; 3) fetching the 16-bit word
```

```
621                             ; at this address; 4) If N was odd (so that N*1.5 was not an integer), shift
622                             ; the word right four bits; 5) mask to 12 bits (AND with 0FFF hex). Entry
623                             ; number zero is used as an end-of-file trap in the OS and is passed to the
624                             ; BIOS to help determine disk format. Entry 1 is reserved for future use.
625                             ; The first available allocation unit is assigned entry number two, and even
626                             ; though it is the first, is called cluster 2. Entries greater than 0FF8H
627                             ; (12-bit fats) or 0FFF8H (16-bit fats) are end of file marks; entries of zero
628                             ; are unallocated. Otherwise, the contents of a FAT entry is the number of
629                             ; the next cluster in the file.
630                             ;
631                             ; Clusters with bad sectors are tagged with FF7H. Any non-zero number would
632                             ; do because these clusters show as allocated, but are not part of any
633                             ; allocation chain and thus will never be allocated to a file. A particular
634                             ; number is selected so that disk checking programs know what to do (ie. a
635                             ; cluster with entry FF7H which is not in a chain is not an error).
636
637                             ;** Character Type Flags
638                             ;
639                             ;   These flags are used in a lookup table indexed by the character code.
640                             ;   They're used to quickly classify characters when parsing paths.
641                             ;   I think that these are only used to parse FCBs - jgl
642
643                             FCHKequ 1  ; I think this means "normal name char, no chks needed" -jgl
644                             FDELIM     equ 2   ; is a delimiter
645                             FSPCHK     equ 4   ; set if character is not a space or equivalent
646                             FFCBequ 8  ; is valid in an FCB
647
648                             ;** Bit definitions for DOS_FLAG
649                             ;
650                             ; Bit 0 - this is set when a $open call is made from $exec. This is used in
651                             ;     $open to indicate to the redirector that this open is being made
652                             ;     by an exec call.
653                             ;
654                             ; Bit 2
655                             ;
656                             ; M003, M027:
657                             ;
658                             ; The start up code of MS PASCAL 3.2 programs depend on the 1M address wrap
659                             ; if they load below 64K. This is a likely possiblity in DOS 5.x with DOS in
660                             ; the HMA. By default DOS will turn A20 OFF before Xferring control to the
661                             ; user program in the case of an Exec call. The next call to DOS will turn
662                             ; A20 line ON. It has been observed that MS PASCAL 3.2 start up does an int
663                             ; 21 ah=25h call before executing the faulty code. This will turn A20 On.
664                             ; In order to support this we will set Bit 2 of this flag in the DOS exec
665                             ; call (msproc.asm) if DOS is running in the HMA. In $set_interrupt_vector in
666                             ; getset.asm A20OFF_COUNT is set to 1 if bit 2 of DOS_FLAG was previously set
667                             ; by a call to exec and if A20OFF_COUNT is 0. In msdisp.asm, if A20OFF_COUNT
668                             ; is non zero then A20 will be turned OFF before returning to the user.
669                             ; Bit 2 will be unconditionally cleared here.
670                             ;
671                             ; M009, M027:
672                             ;
673                             ; Mace utilities MKEYRATE.COM version 1.0 copyright 1987 is an execpacked
674                             ; program converted to a com file. Therefore if DOS is loaded high and if
675                             ; this program is loaded below 64K it will blurt out "packed file is corrupt".
676                             ; This program does an int 21 ah=49h before executing the buggy execpacked
677                             ; code. This int21 call turns a20 on and hence the problem. In $dealloc
678                             ; alloc.asm A20OFF_COUNT is set to 1 if bit 2 of DOS_FLAG was previously set
679                             ; by a call to exec and if A20OFF_COUNT is 0. In msdisp.asm, if A20OFF_COUNT
680                             ; is non zero then A20 will be turned OFF before returning to the user.
681                             ; Bit 2 will be unconditionally cleared here.
682
683                             EXECOPEN    EQU     00000001b      ; bit 0 of DOS_FLAG
684                             SUPPRESS_WINA20     EQU     00000010b      ; M025
685                             EXECA20OFF  EQU     00000100b      ; bit 2 of DOS_FLAG
686
687                             ;=============================================================================
688                             ; VECTOR.INC, MSDOS 6.0, 1991
689                             ;=============================================================================
690                             ; 04/05/2019 - Retro DOS v4.0
691
692                             ; 09/07/2018 - Retro DOS v3.0 (VECTOR.INC, MSDOS 3.3, 1987)
693
694                             ; <interrupt definitions>
695
696                             INTTAB          EQU     20H
697                             INTBASE         EQU     4 * INTTAB
698                             ENTRYPOINT      EQU     INTBASE+40H
699
700                             ;   IF      ALTVECT
701                             ;ALTTAB  EQU     0F0H
702                             ;ALTBASE EQU     4 * ALTTAB
703                             ;   ENDIF
704
705                             ;
706                             ; interrupt assignments
707                             ;
708                             ;   IF      NOT ALTVECT
709                             int_abort       EQU     INTTAB    ; abort process
710                             int_command     EQU     int_abort+1 ; call MSDOS
711                             int_terminate           EQU     int_abort+2 ; int to terminate address
712                             int_ctrl_c      EQU     int_abort+3 ; ^c trapper
713                             int_fatal_abort     EQU     int_abort+4 ; hard disk error
714                             int_disk_read           EQU     int_abort+5 ; logical sector disk read
715                             int_disk_write          EQU     int_abort+6 ; logical sector disk write
716                             int_keep_process    EQU     int_abort+7 ; terminate program and stay
717                                                         ; resident
718                             ;----+----+----+----+----+----+----+----+----+----+----+----+----+----+----;
719                             ;       C A V E A T   P R O G R A M M E R          ;
720                             ;                                                  ;
721                             int_spooler     EQU     int_abort+8 ; spooler call
722                             int_fastcon     EQU     int_abort+9 ; fast CON interrupt
723                             int_IBM         EQU     int_abort+10; critical section maintenance
724                             ;                                                  ;
725                             ;       C A V E A T   P R O G R A M M E R          ;
726                             ;----+----+----+----+----+----+----+----+----+----+----+----+----+----+----;
727                             ;   ELSE
728                             ;int_abort      EQU     INTTAB    ; abort process
729                             ;int_command           EQU     int_abort+1 ; call MSDOS
730                             ;int_terminate         EQU     ALTTAB    ; int to terminate address
731                             ;int_ctrl_c     EQU     int_terminate+1 ; ^c trapper
732                             ;int_fatal_abort    EQU     int_terminate+2 ; hard disk error
733                             ;int_disk_read         EQU     int_abort+5 ; logical sector disk read
734                             ;int_disk_write        EQU     int_abort+6 ; logical sector disk write
735                             ;int_keep_process   EQU     int_abort+7 ; terminate program and stay resident
736                             ;----+----+----+----+----+----+----+----+----+----+----+----+----+----+----;
737                             ;       C A V E A T   P R O G R A M M E R          ;
738                             ;                                                  ;
739                             ;int_spooler           EQU     int_terminate+3 ; spooler call
740                             ;int_fastcon           EQU     int_abort+9 ; fast CON interrupt
741                             ;                                                  ;
742                             ;       C A V E A T   P R O G R A M M E R          ;
743                             ;----+----+----+----+----+----+----+----+----+----+----+----+----+----+----;
744                             ;   ENDIF
```

```
                        addr_int_abort            EQU    4 * int_abort
                        addr_int_command   EQU    4 * int_command
                        addr_int_terminate EQU    4 * int_terminate
                        addr_int_ctrl_c    EQU    4 * int_ctrl_c
                        addr_int_fatal_abort      EQU    4 * int_fatal_abort
                        addr_int_disk_read EQU    4 * int_disk_read
                        addr_int_disk_write       EQU    4 * int_disk_write
                        addr_int_keep_process     EQU    4 * int_keep_process
                        ;----+----+----+----+----+----+----+----+----+----+----+----+----+----+----;
                        ;       C A  V  E  A  T    P  R O  G  R  A  M  M  E  R                    ;
                        ;                                                                          ;
                        addr_int_spooler   EQU    4 * int_spooler
                        addr_int_fastcon   EQU    4 * int_fastcon
                        addr_int_ibm              EQU    4 * int_IBM
                        ;                                                                          ;
                        ;       C A  V  E  A  T    P  R O  G  R  A  M  M  E  R                    ;
                        ;----+----+----+----+----+----+----+----+----+----+----+----+----+----+----;

                        ;==========================================================================
                        ; DIRENT.INC, MSDOS 6.0, 1991
                        ;==========================================================================
                        ; 04/05/2019 - Retro DOS v4.0

                        ; BREAK <Directory entry>

                        ;
                        ;       +--------------------------+
                        ;       |  (12 BYTE) filename/ext   |       0       0
                        ;       +--------------------------+
                        ;       |     (BYTE) attributes     |       11      B
                        ;       +--------------------------+
                        ;       |    (10 BYTE) reserved     |       12      C
                        ;       +--------------------------+
                        ;       | (WORD) time of last write |       22      16
                        ;       +--------------------------+
                        ;       | (WORD) date of last write |       24      18
                        ;       +--------------------------+
                        ;       |   (WORD) First cluster    |       26      1A
                        ;       +--------------------------+
                        ;       |     (DWORD) file size     |       28      1C
                        ;       +--------------------------+
                        ;
                        ;   First byte of filename  = E5 -> free directory entry
                        ;                            = 00 -> end of allocated directory
                        ;   Time:   Bits 0-4=seconds/2, bits 5-10=minute, 11-15=hour
                        ;   Date:   Bits 0-4=day, bits 5-8=month, bits 9-15=year-1980

                        ; 01/01/2024
                        %if 0

                        struc dir_entry
                        .dir_name:  resb 11               ; file name
                        .dir_attr:  resb 1                ; attribute bits
                        .dir_codepg:       resw 1                 ; code page DOS 4.00
                        .dir_extcluster: resw 1                   ; extended attribute starting cluster
                        .dir_attr2: resb 1                ; reserved
                        .dir_pad:   resb 5                ; reserved for expansion
                        .dir_time:  resw 1                ; time of last write
                        .dir_date:  resw 1                ; date of last write
                        .dir_first: resw 1                ; first allocation unit of file
                        .dir_size_l:       resw 1                 ; low 16 bits of file size
                        .dir_size_h:       resw 1                 ; high 16 bits of file size
                        .size:
                        endstruc

                        %else

                        ; 01/01/2024 - Retro DOS v5.0 (*)

                        struc dir_entry
00000000 <res Bh>       .dir_name:  resb 11               ; file name (short file name)
0000000B ??             .dir_attr:  resb 1                ; file attributes (*)
0000000C ??             .dir_nt_res:       resb 1                 ; reserved for use by Windows NT. 0
                        .dir_pad:   ;resb 7               ; (creation time, last access date
                                                          ;  for use by Windows)
                        .dir_crttime_tenth:
0000000D ??                        resb 1
0000000E ????           .dir_crttime:      resw 1
00000010 ????           .dir_crtdate:      resw 1
                        .dir_lstaccdate:
00000012 ????                      resw 1
00000014 ????           .dir_fclus_hi:     resw 1  ; FAT32 fs     ; high word of first cluster number
00000016 ????           .dir_time:  resw 1                ; time of last write
00000018 ????           .dir_date:  resw 1                ; date of last write
                        .dir_fclus:
0000001A ????           .dir_first: resw 1                ; first cluster (alloc. unit) of file
                        .dir_file_size:
0000001C ????           .dir_size_l:       resw 1                 ; low 16 bits of file size
0000001E ????           .dir_size_h:       resw 1                 ; high 16 bits of file size
                        .size:
                        endstruc

                        %endif

                        attr_read_only      EQU      1h
                        attr_hidden         EQU      2h
                        attr_system         EQU      4h
                        attr_volume_id      EQU      8h
                        attr_directory      EQU     10h
                        attr_archive        EQU     20h
                        attr_device     EQU     40h       ; This is a VERY special bit.
                                                          ;   NO directory entry on a disk EVER
                                                          ;   has this bit set. It is set non-zero
                                                          ;   when a device is found by GETPATH

                        attr_all            EQU     attr_hidden+attr_system+attr_directory
                                                            ; OR of hard attributes for FINDENTRY

                        attr_ignore         EQU     attr_read_only+attr_archive
                                                            ; ignore this(ese) attribute(s)
                                                            ; during search first/next

                        attr_changeable     EQU     attr_read_only+attr_hidden+attr_system+attr_archive
                                                            ; changeable via CHMOD

                        DIRFREE             equ     0E5h    ; stored in dir_name[0] to indicate free slot

                        ; --------------------------------------------------------------------------
                        ; 01/01/2024 - Retro DOS v5.0
                        ; ref: FAT32 File System Specification v1.03 (Microsoft, December 6, 2000) (*)

                        attr_long_name     equ attr_read_only|attr_hidden|attr_system|attr_volume_id
                        attr_longname_mask equ attr_long_name|attr_directory|attr_archive
```

```
                        ;============================================================================
                        ; DPB.INC, MSDOS 6.0, 1991
                        ;============================================================================
                        ; 24/04/2019 - Retro DOS v4.0

                        ; 19/07/2018 - Retro DOS v3.0 (DPB.INC, MSDOS 3.3, 1987)
                        ; 07/07/2018 - Retro DOS v3.0 (DPB.INC, MSDOS 6.0, 1991)

                        ; ----------------------------------------------------------------------------
                        ;** DPB - Drive Parameter Block
                        ;
                        ;   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.
                        ; ----------------------------------------------------------------------------

                        ; 01/01/2024
                        %if 0

                        struc      DPB
                        .DRIVE:            resb 1         ; Logical drive # assoc with DPB (A=0,B=1,...)
                        .UNIT:             resb 1         ; Driver unit number of DPB
                        .SECTOR_SIZE:      resw 1         ; Size of physical sector in bytes
                        .CLUSTER_MASK:     resb 1         ; Sectors/cluster - 1
                        .CLUSTER_SHIFT:    resb 1         ; Log2 of sectors/cluster
                        .FIRST_FAT: resw 1         ; Starting record of FATs
                        .FAT_COUNT: resb 1         ; Number of FATs for this drive
                        .ROOT_ENTRIES:     resw 1         ; Number of directory entries
                        .FIRST_SECTOR:     resw 1         ; First sector of first cluster
                        .MAX_CLUSTER:      resw 1         ; Number of clusters on drive + 1
                        ; MSDOS 3.3
                        ;.FAT_SIZE: resb 1         ; Number of records occupied by FAT
                        ; MSDOS 6.0
                        .FAT_SIZE:  resw 1         ; Number of records occupied by FAT
                        .DIR_SECTOR:       resw 1         ; Starting record of directory
                        .DRIVER_ADDR:      resd 1         ; Pointer to driver
                        .MEDIA:            resb 1         ; Media byte
                        .FIRST_ACCESS:     resb 1         ; This is initialized to -1 to force a media
                                                          ; check the first time this DPB is used
                        .NEXT_DPB:  resd 1         ; Pointer to next Drive parameter block
                        .NEXT_FREE: resw 1         ; Cluster # of last allocated cluster
                        .FREE_CNT:  resw 1         ; Count of free clusters, -1 if unknown
                        .size:
                        endstruc

                        %else

                        ; 01/01/2024 - Retro DOS v5.0 (PCDOS 7.1)

                        struc      DPB
 00000000 ??            .DRIVE:            resb 1  ; 0     ; Logical drive # assoc with DPB (A=0,B=1,...)
 00000001 ??            .UNIT:             resb 1  ; 1     ; Driver unit number of DPB
 00000002 ????          .SECTOR_SIZE:      resw 1  ; 2     ; Size of physical sector in bytes
 00000004 ??            .CLUSTER_MASK:     resb 1  ; 4     ; Sectors/cluster - 1
 00000005 ??            .CLUSTER_SHIFT:    resb 1  ; 5     ; Log2 of sectors/cluster
 00000006 ????          .FIRST_FAT: resw 1  ; 6     ; Starting record of FATs
 00000008 ??            .FAT_COUNT: resb 1  ; 8     ; Number of FATs for this drive
 00000009 ????          .ROOT_ENTRIES:     resw 1  ; 9     ; Number of directory entries
 0000000B ????          .FIRST_SECTOR:     resw 1  ; 11    ; First sector of first cluster
 0000000D ????          .MAX_CLUSTER:      resw 1  ; 13    ; Number of clusters on drive + 1
 0000000F ????          .FAT_SIZE:  resw 1  ; 15    ; Number of records occupied by FAT
 00000011 ????          .DIR_SECTOR:       resw 1  ; 17    ; Starting record of directory
 00000013 ????????      .DRIVER_ADDR:      resd 1  ; 19    ; Pointer to driver
 00000017 ??            .MEDIA:            resb 1  ; 23    ; Media byte
 00000018 ??            .FIRST_ACCESS:     resb 1  ; 24    ; This is initialized to -1 to force a media
                                                          ; check the first time this DPB is used
 00000019 ????????      .NEXT_DPB:  resd 1  ; 25    ; Pointer to next Drive parameter block
 0000001D ????          .NEXT_FREE: resw 1  ; 29    ; Cluster # of last allocated cluster
 0000001F ????          .FREE_CNT:  resw 1  ; 31    ; Count of free clusters, -1 if unknown
                        ; FAT32 fs ; 01/01/2024
                        ; ref: https://en.wikibooks.org/wiki/
                        ;      First_steps_towards_system_programming_under_MS-DOS_7/Appendix
                        ;   -- A.03-1. Structure of Drive Parameters Blocks (DPB) ---
 00000021 ????          .FREE_CNT_HW:      resw 1  ; 33    ; High word of free cluster count
 00000023 ????          .EXT_FLAGS: resw 1  ; 35    ; FAT32 extended flags (active FAT number)
 00000025 ????          .FSINFO_SECTOR:    resw 1  ; 37    ; (FAT32 fs) FSINFO structure sector address
 00000027 ????          .BKBOOT_SECTOR:    resw 1  ; 39    ; (FAT32 fs) Backup Boot Sector address
 00000029 ????????      .FCLUS_FSECTOR: resd 1     ; 41    ; The first cluster's first sector address
 0000002D ????????      .LAST_CLUSTER:     resd 1  ; 45    ; The last cluster number
 00000031 ????????      .FAT32_SIZE:       resd 1  ; 49    ; Number of FAT sectors (for FAT32 fs)
 00000035 ????????      .ROOT_CLUSTER:     resd 1  ; 53    ; Root directory's cluster number (FAT32 fs)
                        ; 01/01/2024 - Retro DOS v5.0
 00000039 ????????      .FAT32_NXTFREE:    resd 1  ; 57    ; The next free cluster (for FAT32 fs)
                        .size:             ; 61 bytes ; 01/01/2024 (PCDOS 7.1)
                        endstruc


                        %endif

                        DPBSIZ  EQU     DPB.size   ; Size of the structure in bytes

                        DSKSIZ  EQU DPB.MAX_CLUSTER       ; Size of disk (temp used during init only)

                        ;                                                                           ;
                        ;           C  A  V  E  A  T     P  R  O  G  R  A  M  M  E  R              ;
                        ;----+----+----+----+----+----+----+----+----+----+----+----+----+----+----;

                        ;============================================================================
                        ; SF.INC, MSDOS 6.0, 1991
                        ;============================================================================
                        ; 25/04/2019 - Retro DOS v4.0
                        ; 07/07/2018 - Retro DOS v3.0

                        ;----+----+----+----+----+----+----+----+----+----+----+----+----+----+----;
                        ;           C  A  V  E  A  T     P  R  O  G  R  A  M  M  E  R              ;
                        ;                                                                           ;

                        ; ----------------------------------------------------------------------------
                        ;** SF.INC - System File Table
                        ;
                        ;   AN000   version 4.00   Jan. 1988
                        ;   AN003   PTM 3680 --  make NAME offset the same as before (<=3.30)
                        ;   AN009   PTM 3839       reorder SFT for MS WINDOWS
                        ; ----------------------------------------------------------------------------
                        ;** System File Table SuperStructure
```

```
                                ;
                                ;   The system file table entries are allocated in contiguous groups.
                                ;   There may be more than one such groups; the SF "superstructure"
                                ;   tracks the groups.
                                ; --------------------------------------------------------------------------

                                struc       SFT
00000000 ????????               .SFLink:    resd 1
00000004 ????                   .SFCount:   resw 1          ; number of entries
00000006 ????                   .SFTable:   resw 1          ; beginning of array of the following
                                .size:
                                endstruc

                                ; --------------------------------------------------------------------------
                                ;** System file table entry
                                ;
                                ;   These are the structures which are at SFTABLE in the SF structure.
                                ; --------------------------------------------------------------------------

                                ; 01/01/2024 - Retro DOS v5.0
                                ; 25/04/2019 - Retro DOS v4.0

                                struc SF_ENTRY
00000000 ????                   .sf_ref_count:     resw 1  ; 0     ; number of processes sharing entry
                                                                   ;   if FCB then ref count
00000002 ????                   .sf_mode:   resw 1  ; 2     ; mode of access or high bit on if FCB
00000004 ??                     .sf_attr:   resb 1  ; 4     ; attribute of file
00000005 ????                   .sf_flags:  resw 1  ; 5     ;Bits 8-15
                                                                   ; Bit 15 = 1 if remote file
                                                                   ;        = 0 if local file or device
                                                                   ; Bit 14 = 1 if date/time is not to be
                                                                   ;   set from clock at CLOSE. Set by
                                                                   ;   FILETIMES and FCB_CLOSE. Reset by
                                                                   ;   other reseters of the dirty bit
                                                                   ;   (WRITE)
                                                                   ; Bit 13 = Pipe bit (reserved)
                                                                   ;
                                                                   ; Bits 0-7 (old FCB_devid bits)
                                                                   ; If remote file or local file, bit
                                                                   ; 6=0 if dirty Device ID number, bits
                                                                   ; 0-5 if local file.
                                                                   ; bit 7=0 for local file, bit 7
                                                                   ;      =1 for local I/O device
                                                                   ; If local I/O device, bit 6=0 if EOF (input)
                                                                   ;             Bit 5=1 if Raw mode
                                                                   ;             Bit 0=1 if console input device
                                                                   ;             Bit 1=1 if console output device
                                                                   ;             Bit 2=1 if null device
                                                                   ;             Bit 3=1 if clock device
00000007 ????????              .sf_devptr: resd    1 ; 7   ; Points to DPB if local file, points
                                                                   ; to device header if local device,
                                                                   ; points to net device header if
                                                                   ; remote
0000000B ????                   .sf_firclus:       resw    1 ; 11  ; First cluster of file (bit 15 = 0)
0000000D ????                   .sf_time:   resw    1 ; 13  ; Time associated with file
0000000F ????                   .sf_date:   resw    1 ; 15  ; Date associated with file
00000011 ????????               .sf_size:   resd    1 ; 17  ; Size associated with file
00000015 ????????               .sf_position:      resd    1 ; 21  ; Read/Write pointer or LRU count for FCBs

                                ; Starting here, the next 7 bytes may be used by the file system to store
                                ; an ID

                                ; 09/07/2018 - Retro DOS v3.0

                                ; MSDOS 3.3 SF.INC, 1987
                                ;.sf_cluspos:      resw    1       ; Position of last cluster accessed
                                ;.sf_lstclus       resw    1       ; Last cluster accessed
                                ;.sf_dirsec:       resw    1       ; Sector number of directory sector
                                ;                                  ; for this file
                                ;.sf_dirpos:       resb    1       ; Offset of this entry in the above

                                ; MSDOS 6.0, SF.INC, 1991
00000019 ????                   .sf_cluspos:       resw    1 ; 25  ; Position of last cluster accessed
0000001B ????????               .sf_dirsec: resd    1 ; 27  ; Sector number of directory sector
                                                                   ; for this file
0000001F ??                     .sf_dirpos: resb    1 ; 31  ; Offset of this entry in the above

                                ; End of 7 bytes of file-system specific info.

00000020 <res Bh>               .sf_name:   resb    11 ; 32 ; 11 character name that is in the
                                                                   ; directory entry. This is used by
                                                                   ; close to detect file deleted and
                                                                   ; disk changed errors.
                                .sf_fclus32:       ; 22/03/2024
                                ; SHARING INFO
0000002B ????????               .sf_chain:  resd    1 ; 43  ; link to next SF
0000002F ????                   .sf_UID:    resw    1 ; 47
00000031 ????                   .sf_PID:    resw    1 ; 49  ; owner process identifier (PSP segment)
00000033 ????                   .sf_MFT:    resw    1 ; 51

                                ; MSDOS 6.0, SF.INC, 1991
00000035 ????                   .sf_lstclus:       resw    1 ; 53  ;AN009; Last cluster accessed
00000037 ????????               .sf_IFS_HDR:       resd    1 ; 55  ; pointer to IFS drive or 0:0 for files

                                .size:
                                endstruc

                                ; 20/07/2018
                                ; MSDOS 3.3, SF.INC, 1987
                                %define sf_netid   SF_ENTRY.sf_cluspos    ; byte
                                %define sf_OpenAge SF_ENTRY.sf_position+2 ; word
                                %define sf_LRU        SF_ENTRY.sf_position        ; word
                                ; MSDOS 6.0, SF.INC, 1991
                                %define sf_fsda         SF_ENTRY.sf_cluspos  ; byte ;DOS 4.00
                                %define sf_serial_ID SF_ENTRY.sf_firclus  ; word ;DOS 4.00

                                ; 19/07/2018
                                ; MSDOS 3.3, SF.INC, 1987

                                sf_default_number  EQU    5

                                ; Note that we need to mark an SFT as being busy for OPEN/CREATE. This is
                                ; because an INT 24 may prevent us from 'freeing' it. We mark this as such
                                ; by placing a -1 in the ref_count field.

                                sf_busy EQU -1

                                ; mode mask for FCB detection
                                sf_isFCB           EQU     1000000000000000B

                                ; Flag word masks
                                sf_isnet           EQU     1000000000000000B
                                sf_close_nodate    EQU     0100000000000000B
                                sf_pipe            EQU     0010000000000000B
```

```
                                sf_no_inherit           EQU     0001000000000000B
                                sf_net_spool            EQU     0000100000000000B

                                ; 25/04/2019
                                sf_entry_size equ SF_ENTRY.size ; 59 (MSDOS 6.0)

                                ; --------------------------------------------------------------------------
                                ; Local file/device flag masks
                                ; --------------------------------------------------------------------------

                                devid_file_clean        EQU     40h     ; true if file and not written
                                devid_file_mask_drive   EQU     3Fh     ; mask for drive number

                                devid_device            EQU     80h     ; true if a device
                                devid_device_EOF        EQU     40h     ; true if end of file reached
                                devid_device_raw        EQU     20h     ; true if in raw mode
                                devid_device_special    EQU     10h     ; true if special device
                                devid_device_clock      EQU     08h     ; true if clock device
                                devid_device_null       EQU     04h     ; true if null device
                                devid_device_con_out    EQU     02h     ; true if console output
                                devid_device_con_in     EQU     01h     ; true if console input

                                ; --------------------------------------------------------------------------
                                ; structure of devid field as returned by IOCTL is:
                                ;
                                ;       BIT     7   6   5   4   3   2   1   0
                                ;             |---|---|---|---|---|---|---|---|
                                ;             | I | E | R | S | I | I | I | I |
                                ;             | S | O | A | P | S | S | S | S |
                                ;             | D | F | W | E | C | N | C | C |
                                ;             | E |   |   | C | L | U | O | I |
                                ;             | V |   |   | L | K | L | T | N |
                                ;             |---|---|---|---|---|---|---|---|
                                ;       ISDEV = 1 if this channel is a device
                                ;             = 0 if this channel is a disk file
                                ;
                                ;       If ISDEV = 1
                                ;             EOF = 0 if End Of File on input
                                ;             RAW = 1 if this device is in Raw mode
                                ;                 = 0 if this device is cooked
                                ;             ISCLK = 1 if this device is the clock device
                                ;             ISNUL = 1 if this device is the null device
                                ;             ISCOT = 1 if this device is the console output
                                ;             ISCIN = 1 if this device is the console input
                                ;
                                ;       If ISDEV = 0
                                ;             EOF = 0 if channel has been written
                                ;             Bits 0-5 are the block device number for
                                ;                 the channel (0 = A, 1 = B, ...)
                                ; --------------------------------------------------------------------------

                                devid_ISDEV     EQU     80h
                                devid_EOF       EQU     40h
                                devid_RAW       EQU     20h
                                devid_SPECIAL   EQU     10H
                                devid_ISCLK     EQU     08h
                                devid_ISNUL     EQU     04h
                                devid_ISCOT     EQU     02h
                                devid_ISCIN     EQU     01h

                                devid_block_dev EQU     1Fh             ; mask for block device number

                                ;==========================================================================
                                ; PDB.INC, MSDOS 6.0, 1991
                                ;==========================================================================
                                ; 04/05/2019 - Retro DOS v4.0
                                ; 08/07/2018 - Retro DOS v3.0

                                ; --------------------------------------------------------------------------
                                ; BREAK <Process data block>
                                ; --------------------------------------------------------------------------
                                ;** Process data block (otherwise known as program header)
                                ;

                                ;   These offset are documented in the MSDOS Encyclopedia, so nothing
                                ;   can be rearranged here, ever. Reserved areas are probably safe
                                ;   for use.
                                ; --------------------------------------------------------------------------

                                FILPERPROC  EQU     20

                                struc PDB   ; Process_data_block
00000000 ????                   .EXIT_CALL: resw 1          ; INT int_abort system terminate
00000002 ????                   .BLOCK_LEN: resw 1          ; size of execution block
00000004 ??                                 resb 1
00000005 ??????????             .CPM_CALL:  resb 5          ; ancient call to system
0000000A ????????               .EXIT:          resd 1      ; pointer to exit routine
0000000E ????????               .CTRL_C:    resd 1          ; pointer to ^C routine
00000012 ????????               .FATAL_ABORT:       resd 1      ; pointer to fatal error
00000016 ????                   .PARENT_PID:        resw 1      ; PID of parent (terminate PID)
00000018 <res 14h>              .JFN_TABLE:     resb FILPERPROC ; indices into system table
0000002C ????                   .ENVIRON:   resw 1          ; segment address of environment
0000002E ????????               .USER_STACK:        resd 1      ; stack of self during system calls
00000032 ????                   .JFN_Length:        resw 1      ; number of handles allowed
00000034 ????????               .JFN_Pointer:       resd 1      ; pointer to JFN table
00000038 ????????               .Next_PDB:  resd 1          ; pointer to nested PDB's
0000003C ??                     .InterCon:  resb 1  ; MSDOS 6.0 ; *** jh-3/28/90 ***
0000003D ??                     .Append:    resb 1  ; MSDOS 6.0 ; *** Not sure if still used ***
0000003E ????                   .Novell_Used:       resb 2  ; MSDOS 6.0 ; Novell shell (redir) uses these
00000040 ????                   .Version:   resw 1  ; MSDOS 6.0 ; DOS version reported to this app
00000042 <res Eh>               .PAD1:              resb 14 ; 0Eh
00000050 ??????????             .CALL_SYSTEM:       resb 5          ; portable method of system call
00000055 ????????????????       .PAD2:              resb 7          ; reserved so FCB 1 can be used as
                                                                    ;  an extended FCB
                                ;endstruc   ; MSDOS 3.3
                                            ; MSDOS 6.0
0000005C <res 10h>              .FCB1:              resb 16 ; 10h  ; default FCB 1
0000006C <res 10h>              .FCB2:              resb 16 ; 10h  ; default FCB 2
0000007C ????????               .PAD3:              resb 4          ; not sure if this is used by PDB_FCB2
00000080 <res 80h>              .TAIL:              resb 128        ; command tail and default DTA
                                endstruc

                                ;==========================================================================
                                ; EXE.INC, MSDOS 6.0, 1991
                                ;==========================================================================
                                ; 04/05/2019 - Retro DOS v4.0

                                ;** EXE.INC - Definitions for the EXEC command and EXE files
                                ; --------------------------------------------------------------------------
                                ; The following get used as arguments to the EXEC system call.  They indicate
                                ; whether or not the program is executed or whether or not a program header
                                ; gets created.

```

```
                                exec_func_no_execute EQU 1 ; no execute bit
                                exec_func_overlay    EQU 2 ; overlay bit

                                struc EXEC0
00000000 ????                   .ENVIRON:   resw 1         ; seg addr of environment
00000002 ????????               .COM_LINE:  resd 1         ; pointer to asciz command line
00000006 ????????               .5C_FCB:    resd 1         ; default fcb at 5C
0000000A ????????               .6C_FCB:    resd 1         ; default fcb at 6C
                                .size:
                                endstruc

                                struc EXEC1
00000000 ????                   .ENVIRON:   resw 1         ; seg addr of environment
00000002 ????????               .COM_LINE:  resd 1         ; pointer to asciz command line
00000006 ????????               .5C_FCB:    resd 1         ; default fcb at 5C
0000000A ????????               .6C_FCB:    resd 1         ; default fcb at 6C
0000000E ????                   .SP:        resw 1         ; stack pointer of program
00000010 ????                   .SS:        resw 1         ; stack seg register of program
00000012 ????                   .IP:        resw 1         ; entry point IP
00000014 ????                   .CS:        resw 1         ; entry point CS
                                .size:
                                endstruc

                                struc EXEC3
00000000 ????                   .load_addr: resw 1         ; seg address of load point
00000002 ????                   .reloc_fac: resw 1         ; relocation factor
                                endstruc

                                ;** Exit codes (in upper byte) for terminating programs

                                EXIT_TERMINATE             EQU    0
                                EXIT_ABORT         EQU     0
                                EXIT_CTRL_C        EQU     1
                                EXIT_HARD_ERROR    EQU     2
                                EXIT_KEEP_PROCESS  EQU     3

                                ;** EXE File Header Description

                                struc EXE
00000000 ????                   .signature:   resw 1             ; must contain 4D5A (yay zibo!)
00000002 ????                   .len_mod_512: resw 1             ; low 9 bits of length
00000004 ????                   .pages:       resw 1             ; number of 512b pages in file
00000006 ????                   .rle_count:   resw 1             ; count of reloc entries
00000008 ????                   .par_dir:     resw 1             ; number of paragraphs before image
0000000A ????                   .min_BSS:     resw 1             ; minimum number of para of BSS
0000000C ????                   .max_BSS:     resw 1             ; max number of para of BSS
0000000E ????                   .SS:          resw 1             ; stack of image
00000010 ????                   .SP:          resw 1             ; SP of image
00000012 ????                   .chksum:      resw 1             ; checksum of file (ignored)
00000014 ????                   .IP:          resw 1             ; IP of entry
00000016 ????                   .CS:          resw 1             ; CS of entry
00000018 ????                   .rle_table:   resw 1             ; byte offset of reloc table
0000001A ????                   .iov:         resw 1             ; overlay number (0 for root)
0000001C ????????               .sym_tab:     resd 1             ; offset of symbol table in file
                                .size:
                                endstruc

                                exe_valid_signature     EQU 5A4Dh
                                exe_valid_old_signature EQU 4D5Ah

                                ;** EXE file symbol info definitions

                                struc symbol_entry
00000000 ????????               .value:     resd 1
00000004 ????                   .type:      resw 1
00000006 ??                     .len:       resb 1
00000007 <res FFh>              .name:      resb 255
                                endstruc

                                ;** Data structure passed for ExecReady call

                                struc ERStruc
00000000 ????                    .ER_Reserved:     resw   1      ; reserved, should be zero
00000002 ????                    .ER_Flags: resw   1
00000004 ????????                .ER_ProgName:     resd   1      ; ptr to ASCIIZ str of prog name
00000008 ????                    .ER_PSP:   resw   1      ; PSP of the program
0000000A ????????                .ER_StartAddr:    resd   1      ; Start CS:IP of the program
0000000E ????????                .ER_ProgSize:     resd   1      ; Program size including PSP
                                 .size:
                                endstruc

                                ;** bit fields in ER_Flags

                                ER_EXE             equ    0001h
                                ER_OVERLAY  equ    0002h


                                ;=============================================================================
                                ; ARENA.INC, MSDOS 6.0, 1991
                                ;=============================================================================
                                ; 24/04/2019 - Retro DOS v4.0
                                ; 04/08/2018 - Retro DOS v3.0

                                ;BREAK <Memory arena structure>

                                ;** Arena Header

                                struc ARENA
00000000 ??                     .SIGNATURE: resb 1         ; 4D for valid item, 5A for last item
00000001 ????                   .OWNER:            resw 1         ; owner of arena item
00000003 ????                   .SIZE:             resw 1         ; size in paragraphs of item
00000005 ??????                 .RESERVED:  resb 3         ; reserved
00000008 ????????????????       .NAME:             resb 8         ; owner file name
                                .headersize:
                                endstruc

                                ; 20/05/2019 - Retro DOS v4.0
                                ARENAHEADERSIZE equ ARENA.headersize

                                ; CAUTION: The routines in ALLOC.ASM rely on the fact that arena_signature
                                ; and arena_owner_system are all equal to zero and are contained in DI.
                                ; Change them and change ALLOC.ASM.

                                arena_owner_system  EQU 0               ; free block indication

                                arena_signature_normal    EQU 4Dh        ; valid signature, not end of arena
                                arena_signature_end     EQU 5Ah         ; valid signature, last block in arena

                                FIRST_FIT   EQU     00000000B
                                BEST_FIT    EQU     00000001B
                                LAST_FIT    EQU     00000010B

                                ; MSDOS 6.0
                                LOW_FIRST   EQU     00000000B      ; M001
```

```
                        HIGH_FIRST  EQU     10000000B       ; M001
                        HIGH_ONLY   EQU     01000000B       ; M001

                        LINKSTATE   EQU     00000001B       ; M002

                        HF_MASK             EQU     ~HIGH_FIRST     ; M001
                        HO_MASK             EQU     ~HIGH_ONLY      ; M001

                        STRAT_MASK  EQU     HF_MASK & HO_MASK       ; M001;
                                                                    ; M026: used to mask of bits
                                                                    ; M026: 6 & 7 of AllocMethod

                        ;==========================================================================
                        ; MI.INC, MSDOS 6.0, 1991
                        ;==========================================================================
                        ; 07/07/2018 - Retro DOS v3.0

                        ;BREAK <Machine instruction, flag definitions and character types>

                        mi_INT              EQU     0CDh
                        mi_long_jmp EQU     0EAh
                        mi_Long_CALL        EQU     09Ah
                        mi_Long_RET EQU     0CBh
                        mi_Near_RET EQU     0C3h

                        ;                   xxxxoditszxaxpxc
                        f_Overflow  EQU     0000100000000000B
                        f_Direction EQU     0000010000000000B
                        f_Interrupt EQU     0000001000000000B
                        f_Trace     EQU     0000000100000000B
                        f_Sign              EQU     0000000010000000B
                        f_Zero              EQU     0000000001000000B
                        f_Aux               EQU     0000000000010000B
                        f_Parity    EQU     0000000000000100B
                        f_Carry     EQU     0000000000000001B

                        ;==========================================================================
                        ; FILEMODE.INC, MSDOS 6.0, 1991
                        ;==========================================================================
                        ; 13/07/2018 - Retro DOS v3.0
                        ; 29/04/2019 - Retro DOS v4.0

                        ;** Standard I/O file handles

                        stdin       EQU     0
                        stdout      EQU     1
                        stderr      EQU     2
                        stdaux      EQU     3
                        stdprn      EQU     4

                        ;** File Modes
                        ; <Xenix subfunction assignments>  ; MSDOS 3.3 FILEMODE.INC

                        open_for_read   EQU 0
                        open_for_write  EQU 1
                        open_for_both   EQU 2

                        ; MSDOS 6.0
                        OPEN_FOR_BOTH       equ 2
                        EXEC_OPEN   equ 3   ; access code of 3 indicates that open was
                                            ; made from exec

                        access_mask EQU 0Fh ; 09/08/2018

                        SHARING_MASK            equ 0F0h
                        SHARING_COMPAT          equ 000h
                        SHARING_DENY_BOTH   equ 010h
                        SHARING_DENY_WRITE  equ 020h
                        SHARING_DENY_READ   equ 030h
                        SHARING_DENY_NONE   equ 040h
                        SHARING_NET_FCB     equ 070h
                        SHARING_NO_INHERIT  equ 080h

                        ; 29/04/2019

                        ;** Extended Open Definitions

                        RESERVED_BITS_MASK equ 0FE00h     ; reserved bits for extended open flags
                        EXISTS_MASK    equ 0Fh     ; "file exists" action field
                        NOT_EXISTS_MASK    equ 0F0h

                        ;*  SF_MODE values

                        AUTO_COMMIT_WRITE  equ 4000h
                        INT_24_ERROR               equ 2000h

                        ;*  Flags in EXTOPEN_ON

                        EXT_OPEN_ON         equ 01h
                        EXT_FILE_NOT_EXISTS         equ 04h
                        EXT_OPEN_I24_OFF    equ 02h

                        ;*  Flags in EXTOPEN_FLAG

                        ACTION_OPENED               equ 01h
                        ACTION_CREATED_OPENED       equ 02h
                        ACTION_REPLACED_OPENED      equ 03h
                        EXT_EXISTS_OPEN     equ 01h
                        EXT_EXISTS_FAIL     equ 00h
                        EXT_NEXISTS_CREATE equ 10h

                        ;** Extended Open Structure

                        struc EXT_OPEN_PARM
00000000 ????????       .SET_LIST:  resd 1
00000004 ????           .NUM_OF_PARM:       resw 1
                        endstruc

                        ;==========================================================================
                        ; SYSCALL.INC, MSDOS 6.0, 1991
                        ;==========================================================================
                        ; 29/04/2019 - Retro DOS v4.0
                        ; 09/07/2018 - Retro DOS v3.0 (SYSCALL.INC, MSDOS 3.3, 1987)

                        ; <system call definitions>

                        ABORT                           EQU 0   ;  0      0
                        STD_CON_INPUT                   EQU 1   ;  1      1
                        STD_CON_OUTPUT                  EQU 2   ;  2      2
                        STD_AUX_INPUT                   EQU 3   ;  3      3
                        STD_AUX_OUTPUT                  EQU 4   ;  4      4
                        STD_PRINTER_OUTPUT              EQU 5   ;  5      5
                        RAW_CON_IO                      EQU 6   ;  6      6
                        RAW_CON_INPUT                   EQU 7   ;  7      7
```

```
1489                            STD_CON_INPUT_NO_ECHO           EQU 8   ;  8      8
1490                            STD_CON_STRING_OUTPUT           EQU 9   ;  9      9
1491                            STD_CON_STRING_INPUT            EQU 10  ; 10      A
1492                            STD_CON_INPUT_STATUS            EQU 11  ; 11      B
1493                            STD_CON_INPUT_FLUSH             EQU 12  ; 12      C
1494                            DISK_RESET                      EQU 13  ; 13      D
1495                            SET_DEFAULT_DRIVE               EQU 14  ; 14      E
1496                            FCB_OPEN                        EQU 15  ; 15      F
1497                            FCB_CLOSE                       EQU 16  ; 16     10
1498                            DIR_SEARCH_FIRST                EQU 17  ; 17     11
1499                            DIR_SEARCH_NEXT                 EQU 18  ; 18     12
1500                            FCB_DELETE                      EQU 19  ; 19     13
1501                            FCB_SEQ_READ                    EQU 20  ; 20     14
1502                            FCB_SEQ_WRITE                   EQU 21  ; 21     15
1503                            FCB_CREATE                      EQU 22  ; 22     16
1504                            FCB_RENAME                      EQU 23  ; 23     17
1505                            GET_DEFAULT_DRIVE               EQU 25  ; 25     19
1506                            SET_DMA                         EQU 26  ; 26     1A
1507                            ;----+----+----+----+----+----+----+----+----+----+----+----+----+----+----;
1508                            ;           C  A  V  E  A  T     P  R  O  G  R  A  M  M  E  R           ;
1509                            ;                                                                        ;
1510                            GET_DEFAULT_DPB                 EQU 31  ; 31     1F
1511                            ;                                                                        ;
1512                            ;           C  A  V  E  A  T     P  R  O  G  R  A  M  M  E  R           ;
1513                            ;----+----+----+----+----+----+----+----+----+----+----+----+----+----+----;
1514                            FCB_RANDOM_READ                 EQU 33  ; 33     21
1515                            FCB_RANDOM_WRITE                EQU 34  ; 34     22
1516                            GET_FCB_FILE_LENGTH             EQU 35  ; 35     23
1517                            GET_FCB_POSITION                EQU 36  ; 36     24
1518                            SET_INTERRUPT_VECTOR            EQU 37  ; 37     25
1519                            CREATE_PROCESS_DATA_BLOCK       EQU 38  ; 38     26
1520                            FCB_RANDOM_READ_BLOCK           EQU 39  ; 39     27
1521                            FCB_RANDOM_WRITE_BLOCK          EQU 40  ; 40     28
1522                            PARSE_FILE_DESCRIPTOR           EQU 41  ; 41     29
1523                            GET_DATE                        EQU 42  ; 42     2A
1524                            SET_DATE                        EQU 43  ; 43     2B
1525                            GET_TIME                        EQU 44  ; 44     2C
1526                            SET_TIME                        EQU 45  ; 45     2D
1527                            SET_VERIFY_ON_WRITE             EQU 46  ; 46     2E
1528                            ; Extended functionality group
1529                            GET_DMA                         EQU 47  ; 47     2F
1530                            GET_VERSION                     EQU 48  ; 48     30
1531                            KEEP_PROCESS                    EQU 49  ; 49     31
1532                            ;----+----+----+----+----+----+----+----+----+----+----+----+----+----+----;
1533                            ;           C  A  V  E  A  T     P  R  O  G  R  A  M  M  E  R           ;
1534                            ;                                                                        ;
1535                            GET_DPB                         EQU 50  ; 50     32
1536                            ;                                                                        ;
1537                            ;           C  A  V  E  A  T     P  R  O  G  R  A  M  M  E  R           ;
1538                            ;----+----+----+----+----+----+----+----+----+----+----+----+----+----+----;
1539                            SET_CTRL_C_TRAPPING             EQU 51  ; 51     33
1540                            GET_INDOS_FLAG                  EQU 52  ; 52     34
1541                            GET_INTERRUPT_VECTOR            EQU 53  ; 53     35
1542                            GET_DRIVE_FREESPACE             EQU 54  ; 54     36
1543                            CHAR_OPER                       EQU 55  ; 55     37
1544                            INTERNATIONAL                   EQU 56  ; 56     38
1545                            ; XENIX CALLS
1546                            ;   Directory Group
1547                            MKDIR                           EQU 57  ; 57     39
1548                            RMDIR                           EQU 58  ; 58     3A
1549                            CHDIR                           EQU 59  ; 59     3B
1550                            ;   File Group
1551                            CREAT                           EQU 60  ; 60     3C
1552                            OPEN                            EQU 61  ; 61     3D
1553                            CLOSE                           EQU 62  ; 62     3E
1554                            READ                            EQU 63  ; 63     3F
1555                            WRITE                           EQU 64  ; 64     40
1556                            UNLINK                          EQU 65  ; 65     41
1557                            LSEEK                           EQU 66  ; 66     42
1558                            CHMOD                           EQU 67  ; 67     43
1559                            IOCTL                           EQU 68  ; 68     44
1560                            XDUP                            EQU 69  ; 69     45
1561                            XDUP2                           EQU 70  ; 70     46
1562                            CURRENT_DIR                     EQU 71  ; 71     47
1563                            ;    Memory Group
1564                            ALLOC                           EQU 72  ; 72     48
1565                            DEALLOC                         EQU 73  ; 73     49
1566                            SETBLOCK                        EQU 74  ; 74     4A
1567                            ;    Process Group
1568                            EXEC                            EQU 75  ; 75     4B
1569                            EXIT                            EQU 76  ; 76     4C
1570                            _WAIT                             EQU 77  ; 77     4D
1571                            FIND_FIRST                      EQU 78  ; 78     4E
1572                            ;   Special Group
1573                            FIND_NEXT                       EQU 79  ; 79     4F
1574                            ; SPECIAL SYSTEM GROUP
1575                            ;----+----+----+----+----+----+----+----+----+----+----+----+----+----+----;
1576                            ;           C  A  V  E  A  T     P  R  O  G  R  A  M  M  E  R           ;
1577                            ;                                                                        ;
1578                            SET_CURRENT_PDB                 EQU 80  ; 80     50
1579                            GET_CURRENT_PDB                 EQU 81  ; 81     51
1580                            GET_IN_VARS                     EQU 82  ; 82     52
1581                            SETDPB                          EQU 83  ; 83     53
1582                            ;                                                                        ;
1583                            ;           C  A  V  E  A  T     P  R  O  G  R  A  M  M  E  R           ;
1584                            ;----+----+----+----+----+----+----+----+----+----+----+----+----+----+----;
1585                            GET_VERIFY_ON_WRITE             EQU 84  ; 84     54
1586                            ;----+----+----+----+----+----+----+----+----+----+----+----+----+----+----;
1587                            ;           C  A  V  E  A  T     P  R  O  G  R  A  M  M  E  R           ;
1588                            ;                                                                        ;
1589                            DUP_PDB                         EQU 85  ; 85     55
1590                            ;                                                                        ;
1591                            ;           C  A  V  E  A  T     P  R  O  G  R  A  M  M  E  R           ;
1592                            ;----+----+----+----+----+----+----+----+----+----+----+----+----+----+----;
1593                            RENAME                          EQU 86  ; 86     56
1594                            FILE_TIMES                      EQU 87  ; 87     57
1595                            ALLOCOPER                 EQU 88  ; 88     58
1596                            ; Network extention system calls
1597                            GETEXTENDEDERROR          EQU 89  ; 89     59
1598                            CREATETEMPFILE                    EQU 90  ; 90     5A
1599                            CREATENEWFILE                     EQU 91  ; 91     5B
1600                            LOCKOPER                  EQU 92  ; 92     5C Lock and Unlock
1601                            ;----+----+----+----+----+----+----+----+----+----+----+----+----+----+----;
1602                            ;        C  A  V  E  A  T   P  R  O  G  R  A  M  M  E  R          ;
1603                            ;                                                                        ;
1604                            SERVERCALL                EQU 93  ; 93     5D CommitAll, ServerDOSCall,
1605                                                              ;           CloseByName, CloseUser,
1606                                                              ;           CloseUserProcess,
1607                                                              ;           GetOpenFileList
1608                            ;                                                                        ;
1609                            ;        C  A  V  E  A  T   P  R  O  G  R  A  M  M  E  R          ;
1610                            ;----+----+----+----+----+----+----+----+----+----+----+----+----+----+----;
1611                            USEROPER                  EQU 94  ; 94     5E Get and Set
1612                            ASSINGOPER                EQU 95  ; 95     5F On, Off, Get, Set, Cancel
```

```
                        XNAMETRANS              EQU 96  ; 96     60
                        PATHPARSE               EQU 97  ; 97     61
                        GETCURRENTPSP                   EQU 98  ; 98     62
                        HONGEUL                 EQU 99  ; 99     63
                        ;----+----+----+----+----+----+----+----+----+----+----+----+----+----+----;
                        ;       C A V E A T   P R O G R A M M E R                               ;
                        ;                                                                       ;
                        SET_PRINTER_FLAG        EQU 100 ; 100   64
                        ;                                                                       ;
                        ;       C A V E A T   P R O G R A M M E R                               ;
                        ;----+----+----+----+----+----+----+----+----+----+----+----+----+----+----;
                        GETEXTCNTRY             EQU 101 ; 101   65
                        GETSETCDPG              EQU 102 ; 102   66
                        EXTHANDLE               EQU 103 ; 103   67
                        COMMIT                          EQU 104 ; 104   68

                        ; 29/04/2019 - Retro DOS v4.0
                        ; (MSDOS 6.0, SYSCALL.INC, 1987)

                        GetSetMediaID                   EQU 105 ; 105   69
                        IFS_IOCTL               EQU 107 ; 107   6B
                        ExtOpen                 EQU 108 ; 108   6C

                        ;----+----+----+----+----+----+----+----+----+----+----+----+----+----+----;
                        ;           C A V E A T    P R O G R A M M E R                          ;
                        ;                                                                       ;
                        ;ifdef ROMEXEC
                        ;ROM_FIND_FIRST                 EQU 109 ; 109    6D
                        ;ROM_FIND_NEXT                  EQU 110 ; 110    6E
                        ;ROM_EXCLUDE                    EQU 111 ; 111   6F           ; M035
                        ;endif
                        ;                                                                       ;
                        ;           C A V E A T    P R O G R A M M E R                          ;
                        ;----+----+----+----+----+----+----+----+----+----+----+----+----+----+----;

                        SET_OEM_HANDLER                 EQU 248 ; 248    F8
                        ;OEM_C1                         EQU 249 ; 249    F9
                        ;OEM_C2                         EQU 250 ; 250    FA
                        ;OEM_C3                         EQU 251 ; 251    FB
                        ;OEM_C4                         EQU 252 ; 252    FC
                        ;OEM_C5                         EQU 253 ; 253    FD
                        ;OEM_C6                         EQU 254 ; 254    FE
                        ;OEM_C7                         EQU 255 ; 255    FF

                        ; 01/01/2024 - Retro DOS v5.0 - PCDOS 7.1 IBMDOS.COM
                        ; (PCDOS 7.1 extension to MSDOS 6.22 system calls)

                        ExtCountryInfo                  equ 112 ; 70h
                        LONGNAME                equ 113 ; 71h
                        LFNFINDCLOSE                    equ 114 ; 72h
                        FAT32EXT                equ 115 ; 73h

                        ;==========================================================================
                        ; VERSIONA.INC (MSDOS 6.0, 1991)
                        ;==========================================================================
                        ; 24/04/2019 - Retro DOS 4.0

                        ;MAJOR_VERSION     EQU     6
                        ;;MINOR_VERSION    EQU     00
                        ;MINOR_VERSION     EQU     21  ; MSDOS 6.21

                        ; 04/11/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
                        ;MAJOR_VERSION     EQU     5
                        ;MINOR_VERSION     EQU     0

                        ; 30/12/2022 - Retro DOS v4.2
                        ;MAJOR_VERSION     EQU     6
                        ;MINOR_VERSION     EQU     22

                        ; 01/01/2024 - Retro DOS v5.0
                        MAJOR_VERSION      EQU     7
                        MINOR_VERSION      EQU     10  ; PCDOS 7.1      (7.10)

                        ;==========================================================================
                        ; INTNAT.INC, MSDOS 3.3, 1987
                        ;==========================================================================
                        ; 09/07/2018 - Retro DOS 3.0

                        ; Current structure of the data returned by the international call

                        struc       INTERNAT_BLOCK      ; (-*-) Same with MSDOS 2.11 & MSDOS 6.0
                        .Date_tim_format:
 00000000 ????                      RESW 1          ; 0-USA, 1-EUR, 2-JAP
                        .Currency_sym:
 00000002 ??????????                RESB 5          ; Currency Symbol 5 bytes
                        .Thous_sep:
 00000007 ????                      RESB 2          ; Thousands separator 2 bytes
                        .Decimal_sep:
 00000009 ????                      RESB 2          ; Decimal separator 2 bytes
                        .Date_sep:
 0000000B ????                      RESB 2          ; Date separator 2 bytes
                        .Time_sep:
 0000000D ????                      RESB 2          ; Time separator 2 bytes
                        .Bit_field:
 0000000F ??                        RESB 1          ; Bit values
                                                    ;   Bit 0 = 0 if currency symbol first
                                                    ;         = 1 if currency symbol last
                                                    ;   Bit 1 = 0 if No space after currency symbol
                                                    ;         = 1 if space after currency symbol
                        .Currency_cents:
 00000010 ??                        RESB    1       ; Number of places after currency dec point
                        .Time_24:
 00000011 ??                        RESB    1       ; 1 if 24 hour time, 0 if 12 hour time
                        .Map_call:
 00000012 ????                      RESW    1       ; Address of case mapping call (DWORD)
 00000014 ????                          RESW    1       ; THIS IS TWO WORDS SO IT CAN BE INITIALIZED
                                                    ;  in pieces.
                        .Data_sep:
 00000016 ????                      RESB    2       ; Data list separator character
                        .size:
                        endstruc

                        ; Max size of the block returned by the INTERNATIONAL call

                        internat_block_max EQU     32

                        ;==========================================================================
                        ; SYSVAR.INC (MSDOS 6.0, 1991)
                        ;==========================================================================
                        ; 08/07/2018 - Retro DOS v3.0

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

                        ;SysInitVars STRUC
```

```
1737                                    struc SYSI
1738 00000000 ????????                 .DPB:           resd 1     ; 0    ; DPB chain
1739 00000004 ????????                 .SFT:           resd 1     ; 4    ; SFT chain
1740 00000008 ????????                 .CLOCK:         resd 1     ; 8    ; CLOCK device
1741 0000000C ????????                 .CON:           resd 1     ; 12   ; CON device
1742 00000010 ????                     .MAXSEC:    resw 1 ; 16    ; maximum sector size
1743 00000012 ????????                 .BUF:           resd 1     ; 18   ; points to Hashinitvar
1744 00000016 ????????                 .CDS:           resd 1     ; 22   ; CDS list
1745 0000001A ????????                 .FCB:           resd 1     ; 26   ; FCB chain
1746 0000001E ????                     .Keep:          resw 1     ; 30   ; keep count
1747 00000020 ??                       .NUMIO:         resb 1     ; 32   ; Number of block devices
1748 00000021 ??                       .NCDS:          resb 1     ; 33   ; number of CDS's
1749 00000022 ????????                 .DEV:           resd 1     ; 34   ; device list
1750                                    ; 09/07/2018
1751                                    ; Above parameters are described in MSDOS 3.3 SYSVAR.INC (85/04/10)
1752                                    ; Following parameters are used with MSDOS 6.0 (Retro DOS v4.0)
1753 00000026 ????                     .ATTR:          resw 1     ; 38   ; null device attribute word
1754 00000028 ????                     .STRAT:         resw 1     ; 40   ; null device strategy entry point
1755 0000002A ????                     .INTER:         resw 1     ; 42   ; null device interrupt entry point
1756 0000002C ????????????????         .NAME:          resb 8     ; 44   ; null device name
1757 00000034 ??                       .SPLICE:    resb 1 ; 52    ; TRUE -> splicees being done
1758 00000035 ????                     .IBMDOS_SIZE: resw 1       ; 53   ; DOS size in paragraphs
1759 00000037 ????????                 .IFS_DOSCALL@: resd 1      ; 55   ; IFS DOS service routine entry
1760 0000003B ????????                 .IFS:           resd 1     ; 59   ; IFS header chain
1761 0000003F ????????                 .BUFFERS:   resw 2 ; 63    ; BUFFERS= values (m,n)
1762 00000043 ??                       .BOOT_DRIVE: resb 1        ; 67   ; boot drive A=1 B=2,..
1763 00000044 ??                       .DWMOVE:    resb 1 ; 68    ; 1 if 386 machine
1764 00000045 ????                     .EXT_MEM:   resw 1 ; 69    ; Extended memory size in KB.
1765                                    endstruc
1766                                    ;SysInitVars ENDS
1767
1768                                    ;This is added for more information exchange between DOS, BIOS.
1769                                    ;DOS will give the pointer to SysInitTable in ES:DI. - J.K. 5/29/86
1770
1771                                    ;SysInitVars_Ext struc
1772                                    struc SYSI_EXT
1773 00000000 ????????                 .SysInitVars:      resd 1        ; Points to the above structure.
1774 00000004 ????????                 .Country_Tab:      resd 1        ; DOS_Country_cdpg_info
1775                                    endstruc
1776                                    ;SysInitVars_Ext ends
1777
1778                                    ;=============================================================================
1779                                    ; IOCTL.INC - MSDOS 6.0 - 1991
1780                                    ;=============================================================================
1781                                    ; 09/07/2018 - Retro DOS v3.0
1782
1783                                    ;*** J.K.
1784                                    ;General Guide -
1785                                    ;Category Code:
1786                                    ; 0... .... DOS Defined
1787                                    ; 1... .... User defined
1788                                    ; .xxx xxxx Code
1789
1790                                    ;Function Code:
1791                                    ; 0... .... Return error if unsupported
1792                                    ; 1... .... Ignore if unsupported
1793                                    ; .0.. .... Intercepted by DOS
1794                                    ; .1.. .... Passed to driver
1795                                    ; ..0. .... Sends data/commands to device
1796                                    ; ..1. .... Quries data/info from device
1797                                    ; ...x .... Subfunction
1798                                    ;
1799                                    ; Note that "Sends/queries" data bit is intended only to regularize the
1800                                    ; function set.  It plays no critical role; some functions may contain both
1801                                    ; command and query elements. The convention is that such commands are
1802                                    ; defined as "sends data".
1803
1804                                    ;*****************************;*
1805                                    ; BLOCK DRIVERS           ;*
1806                                    ;*****************************;*
1807
1808                                    ; IOCTL SUB-FUNCTIONS
1809                                    ; (MSDOS 3.3 + MSDOS 6.0)
1810                                    IOCTL_GET_DEVICE_INFO     EQU    0
1811                                    IOCTL_SET_DEVICE_INFO     EQU    1
1812                                    IOCTL_READ_HANDLE  EQU    2
1813                                    IOCTL_WRITE_HANDLE EQU    3
1814                                    IOCTL_READ_DRIVE   EQU    4
1815                                    IOCTL_WRITE_DRIVE  EQU    5
1816                                    IOCTL_GET_INPUT_STATUS    EQU    6
1817                                    IOCTL_GET_OUTPUT_STATUS EQU      7
1818                                    IOCTL_CHANGEABLE?  EQU    8
1819                                    IOCTL_DeviceLocOrRem?     EQU    9
1820                                    IOCTL_HandleLocOrRem?     EQU    0Ah   ;10
1821                                    IOCTL_SHARING_RETRY       EQU    0Bh   ;11
1822                                    GENERIC_IOCTL_HANDLE      EQU    0Ch   ;12
1823                                    GENERIC_IOCTL             EQU    0Dh   ;13
1824                                    ; (MSDOS 6.0 + MSDOS 3.3)
1825                                    IOCTL_GET_DRIVE_MAP       EQU    0Eh   ;14
1826                                    IOCTL_SET_DRIVE_MAP       EQU    0Fh   ;15
1827                                    ; (MSDOS 6.0)
1828                                    IOCTL_QUERY_HANDLE EQU    10h   ;16
1829                                    IOCTL_QUERY_BLOCK  EQU    11h   ;17
1830
1831                                    ; GENERIC IOCTL CATEGORY CODES
1832                                    IOC_OTHER          EQU    0      ; Other device control J.K. 4/29/86
1833                                    IOC_SE                    EQU    1      ; SERIAL DEVICE CONTROL
1834                                    IOC_TC                    EQU    2      ; TERMINAL CONTROL
1835                                    IOC_SC                    EQU    3      ; SCREEN CONTROL
1836                                    IOC_KC                    EQU    4      ; KEYBOARD CONTROL
1837                                    IOC_PC                    EQU    5      ; PRINTER CONTROL
1838                                    IOC_DC                    EQU    8      ; DISK CONTROL (SAME AS RAWIO)
1839
1840                                    ; GENERIC IOCTL SUB-FUNCTIONS
1841                                    RAWIO                     EQU    8
1842
1843                                    ; RAWIO SUB-FUNCTIONS
1844                                    ; (MSDOS 3.3 + MSDOS 6.0)
1845                                    GET_DEVICE_PARAMETERS     EQU    60H
1846                                    SET_DEVICE_PARAMETERS     EQU    40H
1847                                    READ_TRACK         EQU    61H
1848                                    WRITE_TRACK        EQU    41H
1849                                    VERIFY_TRACK              EQU    62H
1850                                    FORMAT_TRACK              EQU    42H
1851                                    ; (MSDOS 6.0)
1852                                    GET_MEDIA_ID              EQU    66h    ;AN000;AN003;changed from 63h
1853                                    SET_MEDIA_ID              EQU    46h    ;AN000;AN003;changed from 43h
1854                                    GET_ACCESS_FLAG    EQU    67h    ;AN002;AN003;Unpublished function.Changed from 64h
1855                                    SET_ACCESS_FLAG    EQU    47h    ;AN002;AN003;Unpublished function.Changed from 44h
1856                                    SENSE_MEDIA_TYPE   EQU    68H    ;Added for 5.00
1857
1858                                    ; SPECIAL FUNCTION FOR GET DEVICE PARAMETERS
1859                                    BUILD_DEVICE_BPB   EQU    000000001B
1860
```

```
                                ; 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
                                ; (MSDOS 3.3 + MSDOS 6.0)
                                STATUS_FOR_FORMAT  EQU     000000001B
                                ; (MSDOS 6.0)
                                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
                                ; (MSDOS 3.3 + MSDOS 6.0)
                                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
                                ; (MSDOS 6.0)
                                ;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

                                ; (MSDOS 3.3)
                                ;MAX_DEV_TYPE              EQU     7

                                ; (MSDOS 6.0)
                                MAX_DEV_TYPE               EQU     10      ; MAXIMUM DEVICE TYPE THAT WE
                                                                           ; CURRENTLY SUPPORT.
                                struc A_SECTORTABLE
00000000 ????                   .ST_SECTORNUMBER:  resw    1
00000002 ????                   .ST_SECTORSIZE:            resw    1
                                .size:
                                endstruc

                                ;=============================================================================
                                ; DEVSYM.INC
                                ;=============================================================================
                                ; 07/07/2018 - Retro DOS v3.0
                                ; 30/04/2019 - Retro DOS v4.0 (DEVSYM.INC, MSDOS 6.0, 1991)
                                ; 21/02/2024 - Retro DOS v5.0

                                ;** DevSym.inc - Device Symbols

                                ; The device table list has the form:
                                struc       SYSDEV
00000000 ????????               .NEXT:              resd 1         ;Pointer to next device header
00000004 ????                   .ATT:               resw 1         ;Attributes of the device
00000006 ????                   .STRAT:             resw 1         ;Strategy entry point
00000008 ????                   .INT:               resw 1         ;Interrupt entry point
0000000A ????????????????       .NAME:              resb 8         ;Name of device (only first byte used for block)
                                .size:
                                endstruc

                                ;
                                ; 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
                                ;

                                ;Attribute bit masks
                                DEVTYP      EQU     8000H           ;Bit 15 - 1 if Char, 0 if block
                                DEVIOCTL EQU    4000H           ;Bit 14 - CONTROL mode bit
                                ISFATBYDEV EQU  2000H           ;Bit 13 - Device uses FAT ID bytes, comp media.

                                ; 09/07/2018 - Retro DOS (DEVSYM.INC, MSDOS 3.3, 1987)

                                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,..).

                                IOQUERY     EQU     0080H           ;Bit 7 - Supports generic IOCtl query

                                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 ; 15/03/2018
                                ;ISIBM      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.

                                EXTDRVR     EQU     0002h           ;BIT 1 - BLOCK DEVICE EXTENDED DRIVER
                                                                ; (MSDOS 6.0, DEVSYM.INC, 1991) ; 30/04/2019

                                ;Static Reguest Header
                                struc       SRHEAD
00000000 ??                     .REQLEN:    resb 1          ;Length in bytes of request block
00000001 ??                     .REQUNIT:   resb 1          ;Device unit number
00000002 ??                     .REQFUNC:   resb 1          ;Type of request
00000003 ????                   .REQSTAT:   resw 1          ;Status Word
00000005 ??????????????????                resb 8          ;Reserved for queue links
                                .size:
                                endstruc

                                ;Status word masks
                                STERR   EQU     8000H           ;Bit 15 - Error
                                STBUI   EQU     0200H           ;Bit 9 - Buisy
                                STDON   EQU     0100H           ;Bit 8 - Done
                                STECODE EQU     00FFH           ;Error code
                                WRECODE EQU     0

                                ;Function codes
                                ;DINITHL EQU    26              ;Size of init header
                                ; 11/04/2024 - Retro DOS 5.0
                                DINITHL EQU     25 ; PCDOS 7.1  ;Size of init header
                                DMEDHL  EQU     15              ;Size of media check header
                                DBPBHL  EQU     22              ;Size of Get BPB header
                                DRDWRHL EQU     22              ;Size of RD/WR header
                                DRDNDHL EQU     14              ;Size of non destructive read header
                                DSTATHL EQU     13              ;Size of status header
                                ; 21/02/2024
                                ;DFLSHL EQU     15              ;Size of flush header
                                DFLSHL      equ     13      ; PCDOS 7.1 IBMDOS.COM  ; 21/02/2024

                                DEVINIT EQU     0               ;Initialization
                                DEVMDCH EQU     1               ;Media check
                                DEVBPB  EQU     2               ;Get BPB
                                DEVRDIOCTL EQU  3               ;IOCTL read
                                DEVRD   EQU     4               ;Read
                                DEVRDND EQU     5               ;Non destructive read no wait (character devs)
                                DEVIST  EQU     6               ;Input status
                                DEVIFL  EQU     7               ;Input flush
                                DEVWRT  EQU     8               ;Write
                                DEVWRTV EQU     9               ;Write with verify
                                DEVOST  EQU     10              ;Output status
                                DEVOFL  EQU     11              ;Output flush
                                DEVWRIOCTL EQU  12              ;IOCTL write

                                ; 09/07/2018 - Retro DOS v3.0 (DEVSYM.INC, MSDOS 3.3, 1987)
                                DEVOPN      EQU     13              ;DEVICE OPEN
                                DEVCLS      EQU     14              ;DEVICE CLOSE
                                DOPCLHL EQU 13              ;SIZE OF OPEN/CLOSE HEADER
                                DEVRMD      EQU     15              ;REMOVABLE MEDIA
                                ; 07/08/2018 - Retro DOS v3.0
                                REMHL       EQU     13              ;SIZE OF REMOVABLE MEDIA HEADER
                                GENIOCTL EQU        19

                                ; THE NEXT THREE ARE USED IN DOS 4.0
                                ;               20
                                ;               21
                                ;               22

                                DEVGETOWN       EQU   23            ;GET DEVICE OWNER
                                DEVSETOWN       EQU   24            ;SET DEVICE OWNER
                                ; 18/05/2019 - Retro DOS v4.0
                                IOCTL_QUERY     EQU   25            ;Query generic ioctl support

                                OWNHL               EQU   13               ;SIZE OF DEVICE OWNER HEADER

                                DEVOUT              EQU   16               ; OUTPUT UNTIL BUSY.
                                DEVOUTL         EQU   DEVWRT        ; LENGTH OF OUTPUT UNTIL BUSY

                                ; ADDED FOR DOS 5.00

                                ; GENERIC IOCTL REQUEST STRUCTURE
                                ;   SEE THE DOS 4.0 DEVICE DRIVER SPEC FOR FURTHER ELABORATION.

                                struc IOCTL_REQ
00000000 <res Dh>               .SRHEAD:    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
                                .size: ; 07/08/2018
                                endstruc

                                ; 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

                                ;; 32-bit absolute read/write input list structure

                                struc ABS_32RW
00000000 ????????               .SECTOR_RBA:        resd 1      ; relative block address
00000004 ????                   .ABS_RW_COUNT:      resw 1      ; number of sectors to be transferred
```

```
00000006 ????????               .BUFFER_ADDR:      resd 1         ; data addrress
                                .size:
                                endstruc

                                ;; media ID info

                                struc MEDIA_ID_INFO
00000000 ????                   .MEDIA_level:      resw   1       ; info level
00000002 ????????               .MEDIA_Serial:     resd   1       ; serial #
00000006 <res Bh>               .MEDIA_Label:      resb   11      ; volume label
00000011 ??????????????????     .MEDIA_System:     resb   8       ; system type
                                .size:
                                endstruc

                                ; equates for DOS34_FLAG
                                ; (BUGBUG: why are bits 0,1,3 and 4 not defined.)

                                FROM_DISK_RESET       EQU   000000000100b   ;from disk reset
                                Force_I24_Fail           EQU   000000100000b   ;form IFS CALL BACK
                                Disable_EOF_I24       EQU   000001000000b   ;disable EOF int24 for input status
                                DBCS_VOLID        EQU   000010000000b   ;indicate from volume id
                                DBCS_VOLID2       EQU   000100000000b   ;indicate 8th char is DBCS
                                CTRL_BREAK_FLAG       EQU   001000000000b   ;indicate control break is input
                                SEARCH_FASTOPEN       EQU   010000000000b   ;set fastopen flag for search
                                EXEC_AWARE_REDIR      EQU   100000000000b   ;M018: this bit is set by a redir
                                                                            ;M018: that knows how to handle
                                                                            ;M018: open for exec

                                NO_FROM_DISK_RESET    EQU   ~FROM_DISK_RESET    ;not from disk reset
                                NO_Force_I24_Fail     EQU   ~Force_I24_Fail     ;not form IFS CALL BACK
                                NO_Disable_EOF_I24    EQU   ~Disable_EOF_I24

                                ;===========================================================================
                                ; ERROR.INC (MSDOS 6.0, 1991)
                                ;===========================================================================
                                ; 16/07/2018 - Retro DOS v3.0

                                ;** ERROR.INC - DOS Error Codes
                                ;
                                ;    The newer (DOS 2.0 and above) "XENIX-style" calls
                                ;    return error codes through AX. If an error occurred then
                                ;    the carry bit will be set and the error code is in AX. If no error
                                ;    occurred then the carry bit is reset and AX contains returned info.
                                ;
                                ;    Since the set of error codes is being extended as we extend the operating
                                ;    system, we have provided a means for applications to ask the system for a
                                ;    recommended course of action when they receive an error.
                                ;
                                ;    The GetExtendedError system call returns a universal error, an error
                                ;    location and a recommended course of action. The universal error code is
                                ;    a symptom of the error REGARDLESS of the context in which GetExtendedError
                                ;    is issued.

                                ;   2.0 error codes

                                error_invalid_function            EQU    1
                                error_file_not_found              EQU    2
                                error_path_not_found              EQU    3
                                error_too_many_open_files EQU    4
                                error_access_denied               EQU    5
                                error_invalid_handle              EQU    6
                                error_arena_trashed               EQU    7
                                error_not_enough_memory   EQU    8
                                error_invalid_block               EQU    9
                                error_bad_environment             EQU    10
                                error_bad_format          EQU    11
                                error_invalid_access              EQU    12
                                error_invalid_data        EQU    13
                                ;**** reserved                    EQU    14     ; *****
                                error_invalid_drive               EQU    15
                                error_current_directory   EQU    16
                                error_not_same_device             EQU    17
                                error_no_more_files               EQU    18

                                ;   These are the universal int 24 mappings for the old INT 24 set of errors

                                error_write_protect               EQU    19
                                error_bad_unit                    EQU    20
                                error_not_ready           EQU    21
                                error_bad_command         EQU    22
                                error_CRC                 EQU    23
                                error_bad_length          EQU    24
                                error_seek                EQU    25
                                error_not_DOS_disk        EQU    26
                                error_sector_not_found            EQU    27
                                error_out_of_paper        EQU    28
                                error_write_fault         EQU    29
                                error_read_fault          EQU    30
                                error_gen_failure         EQU    31

                                ;   the new 3.0 error codes reported through INT 24

                                error_sharing_violation   EQU    32
                                error_lock_violation              EQU    33
                                error_wrong_disk          EQU    34
                                error_FCB_unavailable             EQU    35
                                error_sharing_buffer_exceeded     EQU    36
                                error_Code_Page_Mismatched EQU    37    ; DOS 4.00  ;AN000;
                                error_handle_EOF          EQU    38    ; DOS 4.00  ;AN000;
                                error_handle_Disk_Full            EQU    39    ; DOS 4.00  ;AN000;

                                ;   New OEM network-related errors are 50-79

                                error_not_supported               EQU    50

                                error_net_access_denied           EQU    65     ;M028

                                ;   End of INT 24 reportable errors

                                error_file_exists         EQU    80
                                error_DUP_FCB                     EQU    81     ; *****
                                error_cannot_make         EQU    82
                                error_FAIL_I24                    EQU    83

                                ;   New 3.0 network related error codes

                                error_out_of_structures   EQU    84
                                error_already_assigned            EQU    85
                                error_invalid_password            EQU    86
                                error_invalid_parameter   EQU    87
                                error_NET_write_fault             EQU    88
                                error_sys_comp_not_loaded EQU    90    ; DOS 4.00  ;AN000;

                                ;   BREAK <Interrupt 24 error codes>
```

```
2233
2234                            ;** Int24 Error Codes
2235
2236                            error_I24_write_protect    EQU     0
2237                            error_I24_bad_unit         EQU     1
2238                            error_I24_not_ready                EQU     2
2239                            error_I24_bad_command              EQU     3
2240                            error_I24_CRC                      EQU     4
2241                            error_I24_bad_length               EQU     5
2242                            error_I24_Seek                     EQU     6
2243                            error_I24_not_DOS_disk             EQU     7
2244                            error_I24_sector_not_found EQU     8
2245                            error_I24_out_of_paper             EQU     9
2246                            error_I24_write_fault              EQU     0Ah
2247                            error_I24_read_fault               EQU     0Bh
2248                            error_I24_gen_failure              EQU     0Ch
2249                            ; NOTE: Code 0DH is used by MT-DOS.
2250                            error_I24_wrong_disk               EQU     0Fh
2251
2252                            ;   THE FOLLOWING ARE MASKS FOR THE AH REGISTER ON Int 24
2253                            ;
2254                            ;   NOTE: ABORT is ALWAYS allowed
2255
2256                            Allowed_FAIL                       EQU     00001000B
2257                            Allowed_RETRY                      EQU     00010000B
2258                            Allowed_IGNORE                     EQU     00100000B
2259
2260                            I24_operation                      EQU     00000001B  ;Z if READ,NZ if Write
2261                            I24_area                   EQU     00000110B  ; 00 if DOS
2262                                                                          ; 01 if FAT
2263                                                                          ; 10 if root DIR
2264                                                                          ; 11 if DATA
2265                            I24_class                  EQU     10000000B  ;Z if DISK, NZ if FAT or char
2266
2267                            ;   BREAK <GetExtendedError CLASSes ACTIONs LOCUSs>
2268
2269                            ;** The GetExtendedError call takes an error code and returns CLASS,
2270                            ;   ACTION and LOCUS codes to help programs determine the proper action
2271                            ;   to take for error codes that they don't explicitly understand.
2272
2273                            ;   Values for error CLASS
2274
2275                            errCLASS_OutRes    EQU     1       ; Out of Resource
2276                            errCLASS_TempSit   EQU     2       ; Temporary Situation
2277                            errCLASS_Auth              EQU     3       ; Permission problem
2278                            errCLASS_Intrn             EQU     4       ; Internal System Error
2279                            errCLASS_HrdFail   EQU     5       ; Hardware Failure
2280                            errCLASS_SysFail   EQU     6       ; System Failure
2281                            errCLASS_Apperr    EQU     7       ; Application Error
2282                            errCLASS_NotFnd    EQU     8       ; Not Found
2283                            errCLASS_BadFmt    EQU     9       ; Bad Format
2284                            errCLASS_Locked    EQU     10      ; Locked
2285                            errCLASS_Media             EQU     11      ; Media Failure
2286                            errCLASS_Already   EQU     12      ; Collision with Existing Item
2287                            errCLASS_Unk               EQU     13      ; Unknown/other
2288
2289                            ;   Values for error ACTION
2290
2291                            errACT_Retry               EQU     1       ; Retry
2292                            errACT_DlyRet              EQU     2       ; Delay Retry, retry after pause
2293                            errACT_User        EQU     3       ; Ask user to regive info
2294                            errACT_Abort               EQU     4       ; abort with clean up
2295                            errACT_Panic               EQU     5       ; abort immediately
2296                            errACT_Ignore              EQU     6       ; ignore
2297                            errACT_IntRet              EQU     7       ; Retry after User Intervention
2298
2299                            ;   Values for error LOCUS
2300
2301                            errLOC_Unk         EQU     1       ; No appropriate value
2302                            errLOC_Disk        EQU     2       ; Random Access Mass Storage
2303                            errLOC_Net         EQU     3       ; Network
2304                            errLOC_SerDev              EQU     4       ; Serial Device
2305                            errLOC_Mem         EQU     5       ; Memory
2306
2307                            ;===========================================================================
2308                            ; INT2A.INC (MSDOS 6.0, 1991)
2309                            ;===========================================================================
2310                            ; 04/05/2019 - Retro DOS v4.0
2311
2312                            ;** Int 2A functions
2313                            ; --------------------------------------------------------------------------
2314                            ;   Int 2A is an interface to the network code; it's also overloaded
2315                            ;          as a critical section handler since critical sections
2316                            ;          were originally created to support the net.
2317                            ; --------------------------------------------------------------------------
2318
2319                            ; --------------------------------------------------------------------------
2320                            ;** This table was created by examining the source and may not be
2321                            ;   complete or completely accurate - JGL
2322                            ;
2323                            ;   M010    MD      8/31/90 - Added definition for AH = 5
2324
2325                            ;   (ah) = 0       installation check
2326                            ;                     (returns ah !=0 if installed)
2327                            ;   (ah) = 1       cooked net bios call
2328                            ;   (ah) = 3       query drive shared
2329                            ;                     (ds:si) = "n:" asciz string
2330                            ;   (ah) = 4       net bios
2331                            ;          (al) = 0          cooked net bios call
2332                            ;          (al) = 1          raw net bios call
2333                            ;          (al) = 2          ???
2334                            ;
2335                            ;   (ah) = 5       Get Net Adaptor Resources. CX returns the number of
2336                            ;                   NCBs available/outstanding. DX returns the number of
2337                            ;                   sessions. Supposedly, this is documented in an old
2338                            ;                   IBM PC-LAN reference. Lotus Notes uses it. DOS LAN
2339                            ;                   Manager 2.0 Enhanced responds to it. But it should
2340                            ;                   not be used, as it is a hack, only to get Lotus
2341                            ;                   Notes running.
2342                            ;
2343                            ;   (ah) = 80h     enter critical section
2344                            ;   (ah) = 81h     leave critical section
2345                            ;   (ah) = 82h     free all critical sections (Leave-all)
2346                            ;   (ah) = 84h     entering idle loop (don't understand how this works)
2347                            ; --------------------------------------------------------------------------
2348
2349                            ;** Critical section definitions
2350                            ; --------------------------------------------------------------------------
2351                            ;   Although DOS is not designed to be reentrant there are some hacks
2352                            ;   which various programs use to make it so, in a limited fashion.
2353                            ;   Both WIN386 and some servers block copy a section of the DOS data
2354                            ;   area so that DOS can be reentered on behalf of another thread/program.
2355                            ;   DOS's global data structures, such as the memory arena, are not
2356                            ;   in this area, so critical section indicators are used to protect
```

```
;   those areas.  DOS flags a critical section by issuing an INT_IBM
;   (int 2Ah) at each critical section entry and exit.  Some clients
;   (such as WIN386) just don't "context switch" the DOS when one
;   of these is in effect, others, such as the IBM server, go ahead
;   and reenter the DOS and if they get an int 2A to reenter the same
;   critical section they then switch away from that second thread and
;   let the first one finish and exit the section.
; ----------------------------------------------------------------------------

; These below are subject to leave-all sections
critDisk    EQU     1                   ; Disk I/O critical section
critShare   EQU     1                   ; Sharer I/O critical section
critMem     EQU     1                   ; memory maintenance critical section
critSFT     EQU     1                   ; sft table allocation
critDevice  EQU     2                   ; Device I/O critical section
critNet     EQU     5                   ; network critical section
critIFS     EQU     6                   ; ifsfunc critical section
; These below are not subject to leave-all sections
critASSIGN  EQU     8                   ; Assign has munged a system call

;============================================================================
; MULT.INC (MSDOS 6.0, 1991)
;============================================================================
; 04/05/2019 - Retro DOS v4.0

;Break <Multiplex channels>

; ----------------------------------------------------------------------------
; 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.
;
;      ABh      Unspecified IBM use
;      ACh      Graphics
;      ADh      NLS (toronto)
;      AEh
;      AFh      Mode
;      B0h      GRAFTABL        GRAFTABL
;
;      D7h      Banyan VINES
; ----------------------------------------------------------------------------

;MUX 00-3F reserverd for IBM
;MUX 80-BF reserverd for IBM

;MUX 40-7F reserved for Microsoft

;MUX C0-FF users

MultSHARE   EQU     10h             ; sharer
    ;       1   MFT_enter
    ;       2   MFTClose
    ;       3   MFTclU
    ;       4   MFTCloseP
    ;       5   MFTCloN
    ;       6   set_block
    ;       7   clr_block
    ;       8   chk_block
```

```
2481                                    ;       9   MFT_get
2482                                    ;       10  ShSave
2483                                    ;       11  ShChk
2484                                    ;       12  ShCol
2485                                    ;       13  ShCloseFile
2486
2487                                MultNET     EQU     11h             ; Network support
2488                                MultIFS     EQU     11h                     ; Network support
2489                                    ;   1   IFS_RMDIR
2490                                    ;   2   IFS_SEQ_RMDIR
2491                                    ;   3   IFS_MKDIR
2492                                    ;   4   IFS_SEQ_MKDIR
2493                                    ;   5   IFS_CHDIR
2494                                    ;   6   IFS_CLOSE
2495                                    ;   7   IFS_COMMIT
2496                                    ;   8   IFS_READ
2497                                    ;   9   IFS_WRITE
2498                                    ;   10  IFS_LOCK
2499                                    ;   11  IFS_UNLOCK
2500                                    ;   12  IFS_DISK_INFO
2501                                    ;   13  IFS_SET_FILE_ATTRIBUTE
2502                                    ;   14  IFS_SEQ_SET_FILE_ATTRIBUTE
2503                                    ;   15  IFS_GET_FILE_INFO
2504                                    ;   16  IFS_SEQ_GET_FILE_INFO
2505                                    ;   17  IFS_RENAME
2506                                    ;   18  IFS_SEQ_RENAME
2507                                    ;   19  IFS_DELETE
2508                                    ;   20  IFS_SEQ_DELETE
2509                                    ;   21  IFS_OPEN
2510                                    ;   22  IFS_SEQ_OPEN
2511                                    ;   23  IFS_CREATE
2512                                    ;   24  IFS_SEQ_CREATE
2513                                    ;   25  IFS_SEQ_SEARCH_FIRST
2514                                    ;   26  IFS_SEQ_SEARCH_NEXT
2515                                    ;   27  IFS_SEARCH_FIRST
2516                                    ;   28  IFS_SEARCH_NEXT
2517                                    ;   29  IFS_ABORT
2518                                    ;   30  IFS_ASSOPER
2519                                    ;   31  Printer_SET_STRING
2520                                    ;   32  IFSFlushBuf
2521                                    ;   33  IFSBufWrite
2522                                    ;   34  IFSResetEnvironment
2523                                    ;   35  IFSSpoolCheck
2524                                    ;   36  IFSSpoolClose
2525                                    ;   37  IFSDeviceOper
2526                                    ;   38  IFSSpoolEchoCheck
2527                                    ;   39      - - -   Unused   - - -
2528                                    ;   40      - - -   Unused   - - -
2529                                    ;   41      - - -   Unused   - - -
2530                                    ;   42  SERVER_DOSCALL_CLOSEFILES_FOR_UID
2531                                    ;   43  DEVICE_IOCTL
2532                                    ;   44  IFS_UPDATE_CB
2533                                    ;   45  IFS_FILE_XATTRIBUTES
2534                                    ;   46  IFS_XOPEN
2535                                    ;   47  IFS_DEPENDENT_IOCTL
2536
2537                                MultDOS     EQU     12h             ; DOS call back
2538                                    ;       1   DOS_CLOSE
2539                                    ;       2   RECSET
2540                                    ;       3   Get DOSGROUP
2541                                    ;       4   PATHCHRCMP
2542                                    ;       5   OUT
2543                                    ;       6   NET_I24_ENTRY
2544                                    ;       7   PLACEBUF
2545                                    ;       8   FREE_SFT
2546                                    ;       9   BUFWRITE
2547                                    ;       10  SHARE_VIOLATION
2548                                    ;       11  SHARE_ERROR
2549                                    ;       12  SET_SFT_MODE
2550                                    ;       13  DATE16
2551                                    ;       14  SETVISIT
2552                                    ;       15  SCANPLACE
2553                                    ;       16  SKIPVISIT
2554                                    ;       17  StrCpy
2555                                    ;       18  StrLen
2556                                    ;       19  UCase
2557                                    ;       20  POINTCOMP
2558                                    ;       21  CHECKFLUSH
2559                                    ;       22  SFFromSFN
2560                                    ;       23  GetCDSFromDrv
2561                                    ;       24  Get_User_Stack
2562                                    ;       25  GetThisDrv
2563                                    ;       26  DriveFromText
2564                                    ;       27  SETYEAR
2565                                    ;       28  DSUM
2566                                    ;       29  DSLIDE
2567                                    ;       30  StrCmp
2568                                    ;       31  initcds
2569                                    ;       32  pjfnfromhandle
2570                                    ;       33  $NameTrans
2571                                    ;       34  CAL_LK
2572                                    ;       35  DEVNAME
2573                                    ;       36  Idle
2574                                    ;   37  DStrLen
2575                                    ;   38  NLS_OPEN      DOS 3.3
2576                                    ;   39  $CLOSE        DOS 3.3
2577                                    ;   40  NLS_LSEEK     DOS 3.3
2578                                    ;   41  $READ         DOS 3.3
2579                                    ;   42  FastInit      DOS 4.0
2580                                    ;   43  NLS_IOCTL     DOS 3.3
2581                                    ;   44  GetDevList    DOS 3.3
2582                                    ;   45  NLS_GETEXT    DOS 3.3
2583                                    ;   46  MSG_RETRIEVAL DOS 4.0
2584                                    ;   47  FAKE_VERSION  DOS 4.0
2585
2586                                NLSFUNC     EQU     14h             ; NLSFUNC CALL , DOS 3.3
2587                                    ;       0   NLSInstall
2588                                    ;       1   ChgCodePage
2589                                    ;       2   GetExtInfo
2590                                    ;       3   SetCodePage
2591                                    ;       4   GetCntry
2592
2593                                multANSI    EQU     1Ah                 ; ANSI multiplex number
2594                                    ;   0   INSTALL_CHECK              ; install check for ANSI
2595                                    ;   1   IOCTL_2F                  ; 2F interface to IOCTL
2596                                    ;   2   DA_INFO_2F                ; J.K. Information passing to ANSI.
2597
2598                                multMULT        EQU     4Ah
2599                                multMAGIC       EQU     256*multMULT + 11h
2600                                multMULTRPLTOM  EQU     06h
2601
2602                                    ;   0   swap disk function for single floppy drive m/cs
2603                                    ;       BIOS broadcasts with cx==0, and apps who handle
2604                                    ;       swap disk messaging set cx == -1. BIOS sets dl == requested
```

```
                                ;       drive
                                ;
                                ;   1   Get available HMA & pointer to it. Returns in BX & ES:DI
                                ;   2   Allocate HMA. BX == number of bytes in HMA to be allocated
                                ;       returns pointer in ES:DI
                                ;
                                ;   3-4 currently used by nobody
                                ;   5   Switcher API
                                ;   6   Top of Memory for RPL.
                                ;           BIOS issues INT 2f AX=4a06 & DX = Top of Mem and any RPL
                                ;           code present in TOM should respond with a new TOM in DX
                                ;           to protect itself from MSLOAD & SYSINIT tromping over it.
                                ;           SYSINIT builds an arena with owner type 8 & name 'RPL' to
                                ;           protect the RPL code from COMMAND.COM transient protion.
                                ;           It is the responsibility of RPL program to release the mem.
                                ;   7   Reserved for PROTMAN support.
                                ;  10   smartdrv 4.0
                                ;  11   dblspace api
                                ;  12   MRCI     api
                                ;  13   dblspace/mrci stealth packet api

                               MultAPM     EQU     4ch             ; Obselete ???
                                ;       00h     APM_VER_CHK
                                ;       01h     APM_SUS_SYS_REQ
                                ;       FFh     APM_SUS_RES_BATT_NOTIFY

                               MultPWR_BRDCST  EQU     53h     ; Used by POWER.EXE to broadcast      ; M036
                                                        ;  APM events                          ; M036
                               MultPWR_API     EQU     54h     ; Used for accessing POWER.EXE's API  ; M036

                               ;FASTOPEN is not chained through INT 2F   ; DOS 3.3 F.C.
                               ;     it calls Multdos 42 to set up an entry routine address
                                ;       0   Install status  (reserved)
                                ;       1   Lookup
                                ;       2   Insert
                                ;       3   Delete
                                ;       4   Purge         (reserved)

                               ;===========================================================================
                               ; FIND.INC (MSDOS 6.0, 1991)
                               ;===========================================================================
                               ; 17/05/2019 - Retro DOS v4.0
                               ; 09/07/2018 - Retro DOS v3.0 (MSDOS 3.3, 1987)

                               ;Break      <find first/next buffer>

                               struc find_buf
00000000 ??                    .drive:         resb 1              ; drive of search
00000001 <res Bh>              .name:          resb 11             ; formatted name
0000000C ??                    .sattr:         resb 1              ; attribute of search
0000000D ????                  .LastEnt:   resw 1          ; LastEnt
0000000F ????                  .DirStart:  resw 1          ; DirStart
00000011 ????????              .NETID:         resb 4 ; MSDOS 6.0        ; Reserved for NET
00000015 ??                    .attr:          resb 1              ; attribute found
00000016 ????                  .time:          resw 1              ; time
00000018 ????                  .date:          resw 1              ; date
0000001A ????                  .size_l:    resw 1          ; low(size)
0000001C ????                  .size_h:    resw 1          ; high(size)
0000001E <res Dh>              .pname:         resb 13             ; packed name
                               .size:
                               endstruc

                               ;===========================================================================
                               ; DOSCNTRY.INC (MSDOS 6.0, 1991)
                               ;===========================================================================
                               ; 29/04/2019 - Retro DOS v4.0
                               ; 09/07/2018 - Retro DOS v3.0 (MSDOS 3.3, 1987)

                               ;Equates for COUNTRY INFORMATION.
                               SetCountryInfo     EQU     1       ;country info
                               SetUcase    EQU     2       ;uppercase table
                               SetLcase    EQU     3       ;lowercase table (Reserved)
                               SetUcaseFile       EQU     4       ;uppercase file spec table
                               SetFileList EQU     5       ;valid file character list
                               SetCollate  EQU     6       ;collating sequence
                               SetDBCS     EQU     7       ;double byte character set
                               SetALL             EQU     -1      ;all the entries

                               ;DOS country and code page information table structure.
                               ;Internally, IBMDOS gives a pointer to this table.
                               ;IBMBIO, MODE and NLSFUNC modules communicate with IBMDOS through
                               ;this structure.

                               struc  DOS_CCDPG   ; DOS_country_cdpg_info
00000000 ??????????????????    .ccInfo_reserved:  resb 8  ;reserved for internal use
00000008 <res 40h>             .ccPath_CountrySys:        resb 64 ;path and filename for country info
00000048 ????                  .ccSysCodePage:            resw 1  ;system code page id
0000004A ????                  .ccNumber_of_entries:      resw 1  ; (default value = 6)
0000004C ??                    .ccSetUcase:               resb 1  ; (default value = SetUcase)
0000004D ????????              .ccUcase_ptr:              resd 1  ;pointer to Ucase table

00000051 ??                    .ccSetUcaseFile:   resb 1 ; (default value = SetUcaseFile)
00000052 ????????              .ccFileUcase_ptr:  resd 1 ;pointer to File Ucase table

00000056 ??                    .ccSetFileList:            resb 1  ; (default value = SetFileList)
00000057 ????????              .ccFileChar_ptr:   resd 1  ;pointer to File char list table

0000005B ??                    .ccSetCollate:             resb 1  ; (default value = SetCollate)
0000005C ????????              .ccCollate_ptr:            resd 1  ;pointer to collate table

                               ; MSDOS 6.0
00000060 ??                    .ccSetDBCS:        resb 1  ; (default value = SetDBCS)
00000061 ????????              .ccDBCS_ptr:               resd 1  ; pointer to DBCS table

00000065 ??                    .ccSetCountryInfo: resb 1   ; (default value = SetCountryInfo)
00000066 ????                  .ccCountryInfoLen: resw 1  ;length of country info
00000068 ????                  .ccDosCountry:             resw 1  ;system country code id
0000006A ????                  .ccDosCodePage:            resw 1  ;system code page id
0000006C ????                  .ccDFormat:        resw 1  ;date format
0000006E ??????????            .ccCurSymbol:              resb 5  ;5 byte of (currency symbol+0)
00000073 ????                  .cc1000Sep:        resb 2  ;2 byte of (1000 sep. + 0)
00000075 ????                  .ccDecSep:         resb 2  ;2 byte of (Decimal sep. + 0)
00000077 ????                  .ccDateSep:        resb 2  ;2 byte of (date sep. + 0)
00000079 ????                  .ccTimeSep:        resb 2  ;2 byte of (time sep. + 0)
0000007B ??                    .ccCFormat:        resb 1  ;currency format flags
0000007C ??                    .ccCSigDigits:             resb 1  ;# of digits in currency
0000007D ??                    .ccTFormat:        resb 1  ;time format
0000007E ????????              .ccMono_ptr:               resd 1  ;monocase routine entry point
00000082 ????                  .ccListSep:        resb 2  ;data list separator
00000084 <res Ah>              .ccReserved_area:  resw 5  ;reserved
                               .size:
                               endstruc

                               ;Ucase table
```

```
                                struc CC_UCASE_TAB
00000000 ????                   .ccUcase_leng:            resw 1  ; (default value = 128)
00000002 <res 80h>              .ccUcase_data:            resb 128
                                endstruc

                                ;File Ucase table
                                struc CC_FILE_UCASE_TAB
00000000 ????                   .ccFileucase_leng: resw 1  ; (default value = 128)
00000002 <res 80h>              .ccFileucase_data: resb 128
                                endstruc

                                ;File char list
                                struc CC_FILE_CHAR_TAB
00000000 ????                   .ccFilechar_leng:  resw 1
00000002 <res 2Eh>              .ccFilechar_data:  resb 46
                                endstruc

                                ;collate table
                                struc CC_COLLATE_TAB
00000000 ????                   .ccCollate_leng:   resw 1  ; (default value = 128)
00000002 <res 100h>             .ccCollate_data:   resb 256
                                endstruc

                                OLD_COUNTRY_SIZE  equ     (DOS_CCDPG.size - DOS_CCDPG.ccDFormat - 10)
                                NEW_COUNTRY_SIZE  equ     (DOS_CCDPG.size - DOS_CCDPG.ccDosCountry) ; 38

                                ; 06/08/2018
                                ; DOSCNTRY.INC (MSDOS 6.0, 1991)

                                ;CAPITALIZATION equates
                                CAP_ONE_CHAR       equ    20H
                                CAP_STRING  equ    21H
                                CAP_ASCIIZ  equ    22H
                                CHECK_YES_NO       equ    23H
                                UPPER_TABLE equ    80H

                                ;NLS_YES    equ    59H  ; 'Y'
                                ;NLS_yes2   equ    79H  ; 'y'
                                ;NLS_NO             equ    4EH  ; 'N'
                                ;NLS_no2    equ    6EH  ; 'n'

                                ;============================================================================
                                ; CURDIR.INC (MSDOS 6.0, 1991)
                                ;============================================================================
                                ; 25/04/2019 - Retro DOS v4.0
                                ; 09/07/2018 - Retro DOS v3.0 (CURDIR.INC, MSDOS 3.3, 1987)

                                ;BREAK <Current directory list structure>

                                ;----+----+----+----+----+----+----+----+----+----+----+----+----+----+----;
                                ;        C A V E A T   P R O G R A M M E R          ;
                                ;                                                   ;
                                ; CDS items are used bu the internal routines to store cluster numbers and ;
                                ; network identifiers for each logical name.  The ID field is used dually, ;
                                ; both as net ID and for a cluster number for local devices.  In the case  ;
                                ; of local devices, the cluster number will be -1 if there is a potential  ;
                                ; of the disk being changed or if the path must be recracked.            ;
                                ;
                                ;   Some pathnames have special preambles, such as
                                ;
                                ;           \\machine\sharename\...
                                ;   For these pathnames we can't allow ".." processing to back us
                                ;   up into the special front part of the name.  The CURDIR_END field
                                ;   holds the address of the seperator character which marks
                                ;   the split between the special preamble and the regular
                                ;   path list; ".." processing isn't allowed to back us up past
                                ;   (i.e., before) CURDIR_END
                                ;   For the root, it points at the leading /.  For net
                                ;   assignments it points at the end (nul) of the initial assignment:
                                ;   A:/    \\foo\bar          \\foo\bar\blech\bozo
                                ;     ^             ^                  ^

                                DIRSTRLEN   EQU    64+3           ; Max length in bytes of directory strings

                                TEMPLEN     EQU    DIRSTRLEN*2

                                struc curdir       ; curdir_list
00000000 <res 43h>              .text:             resb DIRSTRLEN        ; text of assignment and curdir
00000043 ????                   .flags:            resw 1                ; various flags
00000045 ????????               .devptr:   resd 1                 ; local pointer to DPB or net device
00000049 ????????               .ID:       resw 2                 ; cluster of current dir (net ID)
0000004D ????                   .user_word: resw 1
0000004F ????                   .end:              resw 1                ; index to ".." backup limit - see above
                                ; MSDOS 6.0
00000051 ??                     .type:             resb 1                ; IFS drive (2=ifs, 4=netuse)
00000052 ????????               .ifs_hdr:  resd 1                 ; Ptr to File System Header
00000056 ????                   .fsda:             resb 2                ; File System Dependent Data Area
                                .size:
                                endstruc

                                curdirLen   EQU curdir.size       ; 88    ; Needed for screwed up

                                %define curdir_netID curdir_ID  ; dword

                                ;** Flag values for CURDIR_FLAGS

                                curdir_isnet       EQU    1000000000000000B
                                CURDIR_isifs       EQU    1000000000000000B ; MSDOS 6.0
                                curdir_inuse       EQU    0100000000000000B
                                curdir_splice      EQU    0010000000000000B
                                curdir_local       EQU    0001000000000000B

                                ;                                                   ;
                                ;        C A V E A T   P R O G R A M M E R          ;
                                ;----+----+----+----+----+----+----+----+----+----+----+----+----+----+----;

                                ;============================================================================
                                ; CPMFCB.INC (MSDOS 3.3, 1987)
                                ;============================================================================
                                ; 09/07/2018 - Retro DOS v3.0

                                ;BREAK <File Control Block definition>

                                ;
                                ; Field definition for FCBs
                                ; The FCB has the following structure:
                                ;
                                ;   +---------------------------+
                                ;   |   Drive indicator(byte)   |
                                ;   +---------------------------+
                                ;   |    Filename (8 chars)     |
                                ;   +---------------------------+
                                ;   |    Extension (3 chars)    |
                                ;   +---------------------------+
```

```
2853                                    ;   |   Current Extent(word)    |
2854                                    ;   +---------------------------+
2855                                    ;   |    Record size (word)     |
2856                                    ;   +---------------------------+
2857                                    ;   |    File Size (2 words)    |
2858                                    ;   +---------------------------+
2859                                    ;   |      Date of write      |
2860                                    ;   +---------------------------+
2861                                    ;   |      Time of write      |
2862                                    ;   +---------------------------+
2863                                    ;---+----+----+----+----+----+----+----+----+----+----+----+----+----+----+----;
2864                                    ;       C A V E A T   P R O G R A M M E R          ;
2865                                    ;                                                  ;
2866                                    ;   +---------------------------+
2867                                    ;   |   8 bytes reserved      |
2868                                    ;   +---------------------------+
2869                                    ;                                                  ;
2870                                    ;       C A V E A T   P R O G R A M M E R          ;
2871                                    ;---+----+----+----+----+----+----+----+----+----+----+----+----+----+----+----;
2872                                    ;   |    next record number     |
2873                                    ;   +---------------------------+
2874                                    ;   |   random record number    |
2875                                    ;   +---------------------------+
2876                                    ;
2877
2878                                    struc       SYS_FCB
2879 00000000 ??                       .drive:     resb 1
2880 00000001 ????????????????         .name:      resb 8
2881 00000009 ??????                   .ext:       resb 3
2882 0000000C ????                     .EXTENT: resw 1
2883 0000000E ????                     .RECSIZ: resw 1                 ; Size of record (user settable)
2884 00000010 ????                     .FILSIZ: resw 1                 ; Size of file in bytes; used with the
2885                                                                ; following word
2886 00000012 ????                     .DRVBP:     resw 1              ; BP for SEARCH FIRST and SEARCH NEXT
2887 00000014 ????                     .FDATE:     resw 1              ; Date of last writing
2888 00000016 ????                     .FTIME:     resw 1              ; Time of last writing
2889                                    ;---+----+----+----+----+----+----+----+----+----+----+----+----+----+----+----;
2890                                    ;       C A V E A T   P R O G R A M M E R          ;
2891                                    ;                                                  ;
2892 00000018 ????????????????         .reserved: resb 8           ; RESERVED
2893                                    ;                                                  ;
2894                                    ;       C A V E A T   P R O G R A M M E R          ;
2895                                    ;---+----+----+----+----+----+----+----+----+----+----+----+----+----+----+----;
2896 00000020 ??                       .NR:resb 1                  ; Next record
2897 00000021 ????????                 .RR:resb 4                  ; Random record
2898                                    .size:
2899                                    endstruc
2900
2901                                    FILDIRENT EQU SYS_FCB.FILSIZ        ; Used only by SEARCH FIRST and SEARCH
2902                                                                ; NEXT
2903                                    ; 20/07/2018
2904                                    %define fcb_sfn    SYS_FCB.reserved ; byte
2905
2906                                    ; Note that fcb_net_handle, fcb_nsl_drive, fcb_nsld_drive and fcb_l_drive
2907                                    ; all must point to the same byte.  Otherwise, the FCBRegen will fail.
2908                                    ; NOTE about this byte (fcb_nsl_drive)
2909                                    ;   The high two bits of this byte are used as follows to indicate the FCB type
2910                                    ;   00 means a local file or device with sharing loaded
2911                                    ;   10 means a remote (network) file
2912                                    ;   01 means a local file with no sharing loaded
2913                                    ;   11 means a local device with no sharing loaded
2914
2915                                    ; 20/07/2018
2916
2917                                    ;
2918                                    ; Network FCB
2919                                    ;
2920
2921                                    %define fcb_net_drive     SYS_FCB.reserved+1  ; byte
2922                                    %define fcb_net_handle    SYS_FCB.reserved+2  ; word
2923                                    %define fcb_netID  SYS_FCB.reserved+4  ; dword
2924
2925                                    ;
2926                                    ; No sharing local file FCB
2927                                    ;
2928
2929                                    %define fcb_nsl_drive     SYS_FCB.reserved+1  ; byte
2930                                    %define fcb_nsl_bits      SYS_FCB.reserved+2  ; byte
2931                                    %define fcb_nsl_firclus SYS_FCB.reserved+3  ; word
2932                                    %define fcb_nsl_dirsec    SYS_FCB.reserved+5  ; word
2933                                    %define fcb_nsl_dirpos  SYS_FCB.reserved+7  ; byte
2934
2935                                    ;
2936                                    ; No sharing local device FCB
2937                                    ;
2938
2939                                    %define fcb_nsld_drive    SYS_FCB.reserved+1  ; byte
2940                                    %define fcb_nsld_drvptr SYS_FCB.reserved+2  ; dword
2941
2942                                    ;
2943                                    ; Sharing local FCB
2944                                    ;
2945
2946                                    %define fcb_l_drive       SYS_FCB.reserved+1  ; byte
2947                                    %define fcb_l_firclus     SYS_FCB.reserved+2  ; word
2948                                    %define fcb_l_mfs  SYS_FCB.reserved+4  ; word
2949                                    %define fcb_l_attr SYS_FCB.reserved+6  ; byte
2950
2951                                    ;
2952                                    ; Bogusness:  the four cases are:
2953                                    ;
2954                                    ;   local file         00
2955                                    ;   local device    40
2956                                    ;   local sharing   C0
2957                                    ;   network     80
2958                                    ;
2959                                    ; Since sharing and network collide, we cannot use a test instruction for
2960                                    ; deciding whether a network or a share check in involved
2961                                    ;
2962                                    FCBDEVICE   EQU 040h
2963                                    FCBNETWORK  EQU 080h
2964                                    FCBSHARE    EQU 0C0h
2965
2966                                    ; FCBSPECIAL must be able to mask off both net and share
2967                                    FCBSPECIAL  EQU 080h
2968                                    FCBMASK     EQU 0C0h
2969
2970                                    ;=============================================================================
2971                                    ; FASTOPEN.INC, MSDOS 6.0, 1991
2972                                    ;=============================================================================
2973                                    ; 11/07/2018 - Retro DOS v3.0
2974                                    ; 25/04/2019 - Retro DOS v4.0
2975                                    ; 21/02/2024 - Retro DOS v5.0
2976
```

```
                                struc FEI   ; FASTOPEN_EXTENDED_INFO
00000000 ??                     .dirpos:    resb 1
00000001 ????????              .dirsec:    resd 1 ; MSDOS 6.0
                                ;.dirsec:   resw 1 ; MSDOS 3.3
00000005 ????                   .clusnum:   resw 1
00000007 ????                               resw 1 ; PCDOS 7.1 ; 21/02/2024
00000009 ????                   .lastent:   resw 1  ; for search first ; MSDOS 6.0
0000000B ????                   .dirstart:  resw 1  ; for search first ; MSDOS 6.0
                                .size:
                                endstruc

                                ; 23/07/2018
                                ;FASTOPEN NAME CACHING Subfunctions
                                FONC_Look_up       equ     1
                                FONC_insert equ     2
                                FONC_delete equ     3
                                FONC_update equ     4
                                FONC_purge  equ     5       ;reserved for the future use.
                                FONC_Rename equ     6       ;AN001

                                ; 27/07/2018
                                ;FastOpen Data Structure
                                struc fastopen_entry       ;Fastopen Entry pointer in DOS
00000000 ????                   .entry_size:       resw 1  ; = 4   ; size of the following
00000002 ????????              .name_caching:     resd 1
                                ; MSDOS 6.0
                                ;.fatchain_caching: resd 1;reserved for future use
                                .size:
                                endstruc

                                ; 27/07/2018
                                ;Equates used in DOS.
                                FastOpen_Set              equ     00000001b
                                FastOpen_Reset            equ     11111110b
                                Lookup_Success            equ     00000010b
                                Lookup_Reset              equ     11111101b
                                Special_Fill_Set       equ     00000100b
                                Special_Fill_Reset     equ     11111011b
                                No_Lookup          equ     00001000b
                                Set_For_Search            equ     00010000b      ;DCR 167

                                ; 09/08/2018
                                ; (FASTXXXX.INC, MSDOS 6.0, 1991)
                                ; Fastxxx equates
                                FastOpen_ID    equ     1
                                FastSeek_ID    equ     2
                                Fast_yes       equ     10000000B ; 80h ; fastxxx flag

                                ;Structure definitions
                                ;
                                struc Fasttable_Entry      ; Fastxxx  Entry pointer in DOS
00000000 ????                   .Fast_Entry_Num: resw 1    ; number of entries
00000002 ????????              .FastOpen_Seek:     resd 1  ; fastopen & fastseek entry address
                                endstruc

                                ;============================================================================
                                ; LOCK.INC, MSDOS 6.0, 1991
                                ;============================================================================
                                ; 14/07/2018 - Retro DOS v3.0

                                ;** LOCK.INC - Definitions for Record Locking

                                ;** LOCK functions

                                LOCK_ALL        equ    0
                                UNLOCK_ALL      equ    1
                                LOCK_MUL_RANGE          equ    2
                                UNLOCK_MUL_RANGE    equ    3
                                LOCK_READ       equ    4
                                WRITE_UNLOCK            equ    5
                                LOCK_ADD        equ    6

                                ;** Structure for Lock buffer

                                struc LockBuf
00000000 ????????              .Lock_position:    resd 1         ; file position for LOCK
00000004 ????????              .Lock_length:      resd 1         ; number of bytes to LOCK
                                endstruc

                                ;============================================================================
                                ; DPL.ASM, MSDOS 6.0, 1991
                                ;============================================================================
                                ; 04/08/2018 - Retro DOS v3.0

                                ; (SRVCALL.ASM)

                                struc DPL
00000000 ????                   .AX:resw    1      ; AX register
00000002 ????                   .BX:resw    1      ; BX register
00000004 ????                   .CX:resw    1      ; CX register
00000006 ????                   .DX:resw    1      ; DX register
00000008 ????                   .SI:resw    1      ; SI register
0000000A ????                   .DI:resw    1      ; DI register
0000000C ????                   .DS:resw    1      ; DS register
0000000E ????                   .ES:resw    1      ; ES register
00000010 ????                   .rsrvd: resw       1      ; Reserved
00000012 ????                   .UID:       resw    1      ; User (Machine) ID (0 = local macine)
00000014 ????                   .PID:       resw    1      ; Process ID (0 = local user PID)
                                .size:
                                endstruc

                                ;----------------------------------------------------------------------------
                                ; DOSDATA
                                ;----------------------------------------------------------------------------
                                ;============================================================================
                                ; 24/04/2019 - Retro DOS v4.0

                                DosDataSg equ 3 ; DOS Data Segment address (dw in 'retrodos4.s')
                                           ; ((just after resident IO.SYS code&data))

                                ;============================================================================
                                ; WIN386.INC, MSDOS 6.0, 1991
                                ;============================================================================
                                ; 24/04/2019 - Retro DOS 4.0

                                ;
                                ;  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
                                
                                ; When Win386_Devcall is broadcast, BX is the Device ID. DOS must
                                ; answer call outs from the DOSMGR
                                
                                Win386_DOSMGR               equ     15H
                                
                                ; The following structures are used to communicate instance data to
                                ; Win386 from the DOS and the BIOS. See Win386 API documentation
                                ; (chapter 3, "Call Out Interfaces") for further description.
                                
                                struc Win386_SIS   ; Startup Info Structure
00000000 ????                    .Version:          resb    2       ; db 3, 0
00000002 ????????                .Next_Dev_Ptr:             resd    1       ; pointer to next SIS in list
00000006 ????????                .Virt_Dev_File_Ptr:        resd    1
0000000A ????????                .Reference_Data:  resd    1
0000000E ????????                .Instance_Data_Ptr:        resd    1       ; pointer to instance data array
                                endstruc
                                
                                size_of_Win386_SIS equ 18 ; 24/04/2019 - Retro DOS v4.0
                                
                                struc Win386_IIS   ; Instance Item Structure
00000000 ????????               .Ptr:                      resd    1       ; pointer to an instance item
00000004 ????                   .Size:                     resw    1       ; size of an instance item
                                endstruc
                                
                                size_of_Win386_IIS equ 6 ; 24/04/2019 - Retro DOS v4.0
                                
                                ;Win386 DOSMGR function return values to indicate operation done
                                
                                WIN_OP_DONE         equ     0B97Ch ;
                                DOSMGR_OP_DONE              equ     0A2ABh ;
                                
                                ;M021
                                ; WInoldap callout multiplex number
                                
                                WINOLDAP            equ     46h     ;
                                
                                ;=============================================================================
                                ;-----------------------------------------------------------------------------
                                ; DOSCODE
                                ;-----------------------------------------------------------------------------
                                ; 04/11/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
                                
                                ;=============================================================================
                                ; MSHEAD.ASM (MSDOS 6.0, 1991)
                                ;=============================================================================
                                ; 16/07/2018 - Retro DOS 3.0
                                ;-----------------------------------------------------------------------------
                                ; 24/04/2019 - Retro DOS 4.0
                                
                                ; MSDOS 6.0
                                ;-----------------------------------------------------------------------------
                                ; FILE : ORIGIN.INC
                                ;-----------------------------------------------------------------------------
                                ; This is included in origin.asm and mshead.asm. Contains the equate that
                                ; is used for ORGing the DOS code.
                                ;
                                ; Brief Description of the necessacity of this ORG:
                                ; ------------------------------------------------
                                ;
                                ; A special problem exits when running out of the HMA. The HMA starts at
                                ; address FFFF:10. There is no place in the HMA with an offset of zero.
                                ; This means programs running out off the HMA must use non-zero offset base
                                ; addresses. It also means that if we're running multiple programs from the
                                ; HMA, the base offset of each segment must atleast be as big as all of the
                                ; HMA segments that precede it.
                                ;
                                ; One solution to this problem to ORG each module at 64K minus its size.
                                ; For instance a code segment 1234h bytes in length would org'd at edcbh.
                                ; This gives max. flexibility regarding it's location in the HMA. By
                                ; selecting segment values between f124h and ffffh it could be located
                                ; anywhere in the HMA. The problem with this is that programs with such
                                ; high ORGs would not be able to run in low RAM.
                                ;
                                ; A compromise solution is to set the ORG address somewhere between 0010h
                                ; and ffffh - their size. In the particular case of the BIOS and the DOS
                                ; the following solution has been implemented:
                                ;
                                ; The Bios Code segment will have a very small offset and run at the very
                                ; front of the HMA, after the VDISK header. THE Dos Code segment will have
                                ; a base offset of (700+<min. size off RAM based BIOS>+<min. size of the DOS
                                ; DATA segment when DOS is running low>). This will reflect the lowest
                                ; possible physical address at which DOS code will run, while still providing
                                ; max. possible flexibility in HMA positioning. This offset MUST NOT be
                                ; smaller then that 20+size of Bios Code segment when running high. This is
                                ; mostly true.
                                ;
                                ; Also this ORG'd value must be communicated to the BIOS. This is done by
                                ; putting this value after the first jmp instruction in the DOS code in
                                ; mshead.asm.
                                ;
                                ; In order for the stripz utility to know how many zeroes to be stripped
                                ; out, this value is placed at the beginning of the binary in origin.asm.
                                ;
                                ; Revision History:
                                ;
                                ; Currently this is being done manually. Therefore any change in the DOS DATA
                                ; Size or the BIOS size should be reflected here. --- Feb 90
                                ;
                                ; BDSIZE.INC contains the equates for BIODATASIZE, BIOCODESIZ and DOSDATASIZ.
                                ; A utility called getsize will obtain the corresponding values from msdos
                                ; and msbio.map and update the values in BDSIZ.INC if they are different.
                                ; DOS should now be built using the batch file makedos.bat which invokes this
                                ; utility. The FORMAT of BDSIZE.INC should not be changed as getsize is
                                ; dependant on that.                                   --- Apr 3 '90
                                ;
                                ; For ROMDOS, however, there is no need to org the doscode to any location
                                ; other than zero.  Therefore the stripz utility will not need to be used,
                                ; so the offset will not need to be included at the beginning of the code
                                ; segment.  Also, the BIOS can just assume that the resident code begins
                                ; at offset zero within the segment.
                                ;
                                ;
                                ;-----------------------------------------------------------------------------
                                
                                BIODATASTART        EQU     0070Oh
                                ;include    bdsize.inc     ; this sets the values:
```

```
                                ;       BIODATASIZ
                                ;       BIOCODESIZ
                                ;       DOSDATASIZ

                            ; 05/12/2022
                            ;BIODATASIZ EQU 00910H     ; 0900h for MSDOS 6.21 IO.SYS
                                        ; 0900h for MSDOS 5.0 IO.SYS
                            ;BIOCODESIZ EQU 01A70H     ; 1A70h for MSDOS 6.21 IO.SYS
                                        ; 1A60h for MSDOS 5.0 IO.SYS
                            ;DOSDATASIZ EQU 01370H     ; 1370h for MSDOS 6.21 IO.SYS
                                        ; 1370h for MSDOS 5.0 IO.SYS
                            ;ifndef ROMDOS
                            ;
                            ;BYTSTART   EQU     BIODATASTART+BIODATASIZ+BIOCODESIZ+DOSDATASIZ
                            ;PARASTART  EQU     (BYTSTART + 0FH) AND (NOT 0FH)
                            ;
                            ;else
                            ;
                            ;BYTSTART   EQU     0
                            ;PARASTART  EQU     0
                            ;
                            ;endif ; ROMDOS

                            ; 24/04/2019 - Retro DOS v4.0 - Modification
                            ; ----------------------------------------------------------------
                            ;MSDAT001E equ 136Ah ; 4970 ; for MSDOS 6.21
                            ;MSDAT001E equ 1370h ; 4976 ; for Retro DOS v4.0 modif. 25/05/2019
                            ;DOSDATASIZE equ MSDAT001E
                            ; 05/12/2022
                            ;DOSDATASIZE equ $ ; 29/04/2019 ; -only- for RETRO DOS v4.0 :
                            ;_PARASTART_ equ DOSDATASIZE ; segment value will point to start of
                                                ; of DOSDATA (in low memory) while
                                                ; dos/kernel code starts just after
                                                ; this data block ((org = DOSDATASIZE))
                                                ; (in low memory or in HMA)
                            ; ----------------------------------------------------------------

                            ; 04/11/2022
                            ; ----------------------------------------------------------------
                            ; NOTE:
                            ; Microsoft dos programmers were calling 'IO.SYS' as dos 'BIOS'
                            ; (Also, they were calling 'ROMBIOS' as 'ROM' only!)
                            ; ----------------------------------------------------------------

                            ; ------------------------------------------------------------------------------
                            ; 06/12/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
                            ; ------------------------------------------------------------------------------
                            ; ------------------------------------------------------------------------------
                            ; 01/01/2024 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMDOS.COM)
                            ; ------------------------------------------------------------------------------

                            ;------------------------------------------------------------------------------
                            ; Start of (PCDOS 7.1) IBMDOS.COM
                            ;------------------------------------------------------------------------------

                            ;segment .code vstart=3DD0h ; 06/12/2022
                            ; 29/09/2023
                            ;segment .code vstart=3DE0h ; 19/09/2023 - Retro DOS v4.2 (Modified MSDOS 6.22)
                            ; ------------------------------------------------------------------------------
                            ; 01/01/2024
                            segment .code vstart=3F10h ; 01/01/2024 - Retro DOS v5.0 (Modified PCDOS 7.1)

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

                            ;[ORG 3DE0h]

                            ;[ORG _PARASTART_]     ; [org 136Ah]

                            ;[ORG 1370h] ; 25/05/2019 - Retro DOS v4.0

                            ; 01/01/2024 - Retro DOS v5.0
                            ;[ORG 3F10h]

                                ; 05/12/2022 - RetroDOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
                                ;PARASTART equ 3DD0h ; BIOSDATASTART+BIOSDATASIZE
                                                    ; +BIOSCODESIZE+DOSDATASIZE (rounded up)

                                ; 29/09/2023
                                ; 19/09/2023 - Retro DOS v4.2 (Modified MSDOS 6.22 MSDOS.SYS)
                                ;PARASTART equ 3DE0h  ; (MSDOS 6.22 MSDOS.SYS)

                                ; 01/01/2024 - Retro DOS v4.2 (Modified PCDOS 7.1 IBMDOS.COM)
                                PARASTART equ 3F10h   ; (PCDOS 7.1 IBMDOS.COM)

                                [ORG PARASTART]

                            _$STARTCODE:

                            ;PARASTART:
00000000 E9198B                     JMP     DOSINIT

                                ;dw     PARASTART      ; PARASTART = 3DE0h for MSDOS 6.0, 6.22
                                ; 04/11/2022           ; PARASTART = 3DD0h for MSDOS 5.0
00000003 [0000]                 dw      _$STARTCODE    ; PARASTART = 3F10h for PCDOS 7.1 ; 01/01/2024

                            BioDataSeg:
00000005 7000                   dw      0070h          ; Bios data segment fixed at 70h

                            ; DosDSeg is a data word in the DOSCODE segment that is loaded with
                            ; the segment address of DOSDATA. This is purely an optimization, that
                            ; allows getting the DOS data segment without going through the
                            ; BIOS data segment. It is used by the "getdseg" macro.

                            DosDSeg:
00000007 0000                   dw      0

                            ;==============================================================================
                            ; MSTABLE.ASM (MSDOS 6.0, 1991)
                            ;==============================================================================
                            ; 16/07/2018 - Retro DOS 3.0
                            ; 29/04/2019 - Retro DOS 4.0

                                ; (MSDOS version)
                                ; DOSCODE:3DE9h (MSDOS 6.21, MSDOS.SYS)
                                ;db     6
                                ;db     20
                                ; 04/11/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
                                ; DOSCODE:3DD9h (MSDOS 5.0, MSDOS.SYS)
                                ;db     5
                                ;db     0

                                ; Offset 0C78h in IBMDOS.COM (MSDOS 3.3, 1987)
                            MSVERS:                     ; MS-DOS version in hex for $GET_VERSION
00000009 07                 MSMAJOR: DB MAJOR_VERSION  ; DOS_MAJOR_VERSION
```

```
0000000A 0A                     MSMINOR: DB MINOR_VERSION  ; DOS_MINOR_VERSION

                                ;;hkn YRTAB & MONTAB moved to DOSDATA in ms_data.asm
                                ;   I_am    YRTAB,8,<200,166,200,165,200,165,200,165>    ; [SYSTEM]
                                ;   I_am    MONTAB,12,<31,28,31,30,31,30,31,31,30,31,30,31> ; [SYSTEM]

                                ; DOSTAB.ASM (MSDOS 6.0, 1991)
                                ; YRTAB & MONTAB moved from TABLE segment in ms_table.asm
                                ;
                                ;   I_am    YRTAB,8,<200,166,200,165,200,165,200,165>
                                ;   I_am    MONTAB,12,<31,28,31,30,31,30,31,31,30,31,30,31>

                                ; This is the error code mapping table for INT 21 errors. This table defines
                                ; those error codes which are "allowed" for each system call. If the error
                                ; code ABOUT to be returned is not "allowed" for the call, the correct action
                                ; is to return the "real" error via Extended error, and one of the allowed
                                ; errors on the actual call.
                                ;
                                ; The table is organized as follows:
                                ;
                                ;    Each entry in the table is of variable size, but the first
                                ;       two bytes are always:
                                ;
                                ;       Call#,Cnt of bytes following this byte
                                ;
                                ; EXAMPLE:
                                ;       Call 61 (OPEN)
                                ;
                                ;       DB      61,5,12,3,2,4,5
                                ;
                                ;       61 is the AH INT 21 call value for OPEN.
                                ;        5 indicates that there are 5 bytes after this byte (12,3,2,4,5).
                                ;       Next five bytes are those error codes which are "allowed" on OPEN.
                                ;       The order of these values is not important EXCEPT FOR THE LAST ONE (in
                                ;       this case 5).  The last value will be the one returned on the call if
                                ;       the "real" error is not one of the allowed ones.
                                ;
                                ; There are a number of calls (for instance all of the FCB calls) for which
                                ;   there is NO entry.  This means that NO error codes are returned on this
                                ;   call, so set up an Extended error and leave the current error code alone.
                                ;
                                ; The table is terminated by a call value of 0FFh

                                ;PUBLIC I21_MAP_E_TAB
                                    ; 10/08/2018

                                ; 29/04/2019
                                ; DOSCODE:3DE9h (MSDOS 6.21, MSDOS.SYS)
                                ; 04/11/2022
                                ; DOSCODE:3DDBh    (MSDOS 5.0 MSDOS.SYS)
                                ; 01/01/2024
                                ; DOSCODE:3F1Bh    (PCDOS 7.1 IBMDOS.COM)

                                I21_MAP_E_TAB:     ; LABEL BYTE
0000000B 38020102                   DB  INTERNATIONAL,2,error_invalid_function,error_file_not_found
0000000F 3903030205                 DB  MKDIR,3,error_path_not_found,error_file_not_found,error_access_denied
00000014 3A041003                   DB  RMDIR,4,error_current_directory,error_path_not_found
00000018 0205                       DB          error_file_not_found,error_access_denied
0000001A 3B020203                   DB  CHDIR,2,error_file_not_found,error_path_not_found
0000001E 3C040302                   DB  CREAT,4,error_path_not_found,error_file_not_found
00000022 04                         DB          error_too_many_open_files
00000023 05                         DB          error_access_denied
                                    ; MSDOS 6.0
00000024 3D0603020C                 DB      OPEN,6,error_path_not_found,error_file_not_found,error_invalid_access
00000029 04                         DB          error_too_many_open_files
0000002A 1A05                       DB          error_not_DOS_disk,error_access_denied
                                    ; MSDOS 3.3
                                    ;DB     OPEN,5,error_path_not_found,error_file_not_found,error_invalid_access
                                    ;DB            error_too_many_open_files,error_access_denied
0000002C 3E0106                     DB  CLOSE,1,error_invalid_handle
0000002F 3F020605                   DB  READ,2,error_invalid_handle,error_access_denied
00000033 40020605                   DB  WRITE,2,error_invalid_handle,error_access_denied
00000037 4103030205                 DB  UNLINK,3,error_path_not_found,error_file_not_found,error_access_denied
0000003C 42020601                   DB  LSEEK,2,error_invalid_handle,error_invalid_function
00000040 4304030201                 DB  CHMOD,4,error_path_not_found,error_file_not_found,error_invalid_function
00000045 05                         DB          error_access_denied
00000046 44050F0D01                 DB  IOCTL,5,error_invalid_drive,error_invalid_data,error_invalid_function
0000004B 0605                       DB          error_invalid_handle,error_access_denied
0000004D 45020604                   DB  XDUP,2,error_invalid_handle,error_too_many_open_files
00000051 46020604                   DB  XDUP2,2,error_invalid_handle,error_too_many_open_files
                                    ; MSDOS 6.0
00000055 47021A0F                   DB  CURRENT_DIR,2,error_not_DOS_disk,error_invalid_drive
                                    ; MSDOS 3.3
                                    ;DB  CURRENT_DIR,1,error_invalid_drive
00000059 48020708                   DB  ALLOC,2,error_arena_trashed,error_not_enough_memory
0000005D 49020709                   DB  DEALLOC,2,error_arena_trashed,error_invalid_block
00000061 4A03070908                 DB  SETBLOCK,3,error_arena_trashed,error_invalid_block,error_not_enough_memory
00000066 4B08030102                 DB  EXEC,8,error_path_not_found,error_invalid_function,error_file_not_found
0000006B 040B0A                     DB          error_too_many_open_files,error_bad_format,error_bad_environment
0000006E 0805                       DB          error_not_enough_memory,error_access_denied
00000070 4E03030212                 DB  FIND_FIRST,3,error_path_not_found,error_file_not_found,error_no_more_files
00000075 4F0112                     DB  FIND_NEXT,1,error_no_more_files
                                    ; MSDOS 6.0
00000078 5605110302                 DB  RENAME,5,error_not_same_device,error_path_not_found,error_file_not_found
0000007D 1005                       DB              error_current_directory,error_access_denied
                                    ; MSDOS 3.3
                                    ;DB  RENAME,4,error_not_same_device,error_path_not_found,error_file_not_found
                                    ;DB              error_access_denied
                                    ; MSDOS 6.0
0000007F 57040608                   DB  FILE_TIMES,4,error_invalid_handle,error_not_enough_memory
00000083 0D01                       DB              error_invalid_data,error_invalid_function
                                    ; MSDOS 3.3
                                    ;DB  FILE_TIMES,2,error_invalid_handle,error_invalid_function
00000085 580101                     DB  ALLOCOPER,1,error_invalid_function
00000088 5A040302                   DB  CREATETEMPFILE,4,error_path_not_found,error_file_not_found
0000008C 0405                       DB          error_too_many_open_files,error_access_denied
0000008E 5B055003                   DB  CREATENEWFILE,5,error_file_exists,error_path_not_found
00000092 020405                     DB          error_file_not_found,error_too_many_open_files,error_access_denied
00000095 5C040601                   DB  LOCKOPER,4,error_invalid_handle,error_invalid_function
00000099 2421                       DB          error_sharing_buffer_exceeded,error_lock_violation
0000009B 65020102                   DB  GETEXTCNTRY,2,error_invalid_function,error_file_not_found      ;DOS 3.3
0000009F 66020102                   DB  GETSETCDPG,2,error_invalid_function,error_file_not_found        ;DOS 3.3
000000A3 680106                     DB  COMMIT,1,error_invalid_handle                                   ;DOS 3.3
000000A6 67030408                   DB  EXTHANDLE,3,error_too_many_open_files,error_not_enough_memory
000000AA 01                         DB              error_invalid_function
                                    ; MSDOS 6.0
000000AB 6C0A                       DB      ExtOpen,10
000000AD 03020C                     DB        error_path_not_found,error_file_not_found,error_invalid_access
000000B0 045008                     DB          error_too_many_open_files,error_file_exists,error_not_enough_memory
000000B3 1A0D                       DB          error_not_DOS_disk,error_invalid_data
000000B5 0105                       DB          error_invalid_function,error_access_denied
000000B7 69040F0D                   DB      GetSetMediaID,4,error_invalid_drive,error_invalid_data
000000BB 0105                       DB          error_invalid_function,error_access_denied
                                ; 01/01/2024
```

```
                                    ; PCDOS 7.1
000000BD 700101                     db      ExtCountryInfo,1,error_invalid_function
000000C0 FF                         DB  0FFh

                                ;19/09/2023
                                ;22/12/2022
                                ;04/11/2022
                                ;29/04/2019 - Retro DOS v4.0
                                ;============================================================================
                                ;   Retro DOS v4.0
                                ;============================================================================
000000C1 00                         db      0
                                RETRODOSMSG:
000000C2 0D0A                       db      13,10
                                    ;;;;;db"Retro DOS v4.0 by Erdogan Tan [2019]"
                                    ;;;;db  "Retro DOS v4.0 by Erdogan Tan [2022]"
                                    ;;;db   "Retro DOS v4.1 by Erdogan Tan [2022]" ; 28/12/2022
                                    ;;db    "Retro DOS v4.2 by Erdogan Tan [2022]" ; 30/12/2022
                                    ;db     "Retro DOS v4.2 by Erdogan Tan [2023]"
000000C4 526574726F20444F53-        db      "Retro DOS v5.0 by Erdogan Tan [2024]" ; 01/01/2024
000000CD 2076352E3020627920-
000000D6 4572646F67616E2054-
000000DF 616E205B323032345D
000000E8 0D0A2400                   db      13,10,"$", 0

                                ;============================================================================
                                ; MSTABLE.ASM, MSDOS 6.0, 1991
                                ;============================================================================
                                ; 11/07/2018 - Retro DOS v3.0

                                    %define short_addr dw  ; 03/03/2018 - Retro DOS v2.0
                                align 2
                                    ; IBMDOS.COM (MSDOS 3.3) - Offset 0E00h

                                ; Standard Functions
                                ;DISPATCH  LABEL WORD
                                DISPATCH:
                                    ; 16/07/2018 - Retro DOS v3.0
                                    ; (MSDOS 3.3)

                                ; 29/04/2019
                                ; DOSCODE:3E9Eh (MSDOS 6.21, MSDOS.SYS)

                                ; 04/11/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
                                ; DOSCODE:3E8Eh (MSDOS 5.0, MSDOS.SYS)

                                ; 01/01/2024 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMDOS.COM)
                                ; DOSCODE:3FD2h (PCDOS 7.1, IBMDOS.COM)

000000EC [7172]                         short_addr  _$ABORT                         ;  0      0
000000EE [261C]                         short_addr  _$STD_CON_INPUT                 ;  1      1
000000F0 [2F1C]                         short_addr  _$STD_CON_OUTPUT                ;  2      2
000000F2 [E11C]                         short_addr  _$STD_AUX_INPUT                 ;  3      3
000000F4 [FC1C]                         short_addr  _$STD_AUX_OUTPUT                ;  4      4
000000F6 [021D]                         short_addr  _$STD_PRINTER_OUTPUT     ;  5       5
000000F8 [671B]                         short_addr  _$RAW_CON_IO             ;  6       6
000000FA [931B]                         short_addr  _$RAW_CON_INPUT                 ;  7      7
000000FC [4219]                         short_addr  _$STD_CON_INPUT_NO_ECHO         ;  8      8
000000FE [9A19]                         short_addr  _$STD_CON_STRING_OUTPUT         ;  9      9
00000100 [A619]                         short_addr  _$STD_CON_STRING_INPUT          ; 10      A
00000102 [161D]                         short_addr  _$STD_CON_INPUT_STATUS          ; 11      B
00000104 [1F1D]                         short_addr  _$STD_CON_INPUT_FLUSH           ; 12      C
00000106 [1E14]                         short_addr  _$DISK_RESET             ; 13       D
00000108 [570F]                         short_addr  _$SET_DEFAULT_DRIVE             ; 14      E
0000010A [1124]                         short_addr  _$FCB_OPEN                      ; 15      F
0000010C [BE1D]                         short_addr  _$FCB_CLOSE                     ; 16     10
0000010E [2825]                         short_addr  _$DIR_SEARCH_FIRST              ; 17     11
00000110 [6E25]                         short_addr  _$DIR_SEARCH_NEXT               ; 18     12
00000112 [641D]                         short_addr  _$FCB_DELETE             ; 19       13
00000114 [6C22]                         short_addr  _$FCB_SEQ_READ                  ; 20     14
00000116 [7022]                         short_addr  _$FCB_SEQ_WRITE                  ; 21     15
00000118 [1825]                         short_addr  _$FCB_CREATE             ; 22       16
0000011A [2A1E]                         short_addr  _$FCB_RENAME             ; 23       17
                                    ; 16/07/2018
                                        ;short_addr _CPMFUNC                        ; 24     18
0000011C [8A06]                         short_addr  NO_OP                    ; 24       18
0000011E [520F]                         short_addr  _$GET_DEFAULT_DRIVE             ; 25     19
00000120 [470F]                         short_addr  _$SET_DMA                       ; 26     1A

                                ;----+----+----+----+----+----+----+----+----+----+----+----+----+----+----;
                                ;           C  A  V  E  A  T     P  R  O  G  R  A  M  M  E  R            ;
                                ;                                                                        ;
00000122 [F512]                         short_addr  _$SLEAZEFUNC             ; 27       1B
00000124 [F712]                         short_addr  _$SLEAZEFUNCDL                  ; 28     1C
                                ;                                                                        ;
                                ;           C  A  V  E  A  T     P  R  O  G  R  A  M  M  E  R            ;
                                ;----+----+----+----+----+----+----+----+----+----+----+----+----+----+----;

                                        ;short_addr  _CPMFUNC                       ; 29     1D
                                        ;short_addr  _CPMFUNC                       ; 30     1E

                                ; 08/07/2018 - Retro DOS v3.0
                                ; MSDOS 6.0 - MSTABLE.ASM, 1991

00000126 [8A06]                     short_addr  NO_OP                       ; 29       1D
00000128 [8A06]                     short_addr  NO_OP                       ; 30       1E

                                ;----+----+----+----+----+----+----+----+----+----+----+----+----+----+----;
                                ;           C  A  V  E  A  T     P  R  O  G  R  A  M  M  E  R            ;
                                ;                                                                        ;
0000012A [4213]                         short_addr  _$GET_DEFAULT_DPB               ; 31     1F
                                ;                                                                        ;
                                ;           C  A  V  E  A  T     P  R  O  G  R  A  M  M  E  R            ;
                                ;----+----+----+----+----+----+----+----+----+----+----+----+----+----+----;
                                        ;short_addr _CPMFUNC                        ; 32     20

                                ; 08/07/2018 - Retro DOS v3.0
                                ; MSDOS 6.0 - MSTABLE.ASM, 1991

0000012C [8A06]                     short_addr  NO_OP                       ; 32       20

0000012E [7422]                         short_addr  _$FCB_RANDOM_READ               ; 33     21
00000130 [7822]                         short_addr  _$FCB_RANDOM_WRITE              ; 34     22
00000132 [761D]                         short_addr  _$GET_FCB_FILE_LENGTH           ; 35     23
00000134 [4C1D]                         short_addr  _$GET_FCB_POSITION              ; 36     24

                                ;MAXCALL = ($-DISPATCH)/2 - 1
                                MAXCALL EQU ($-DISPATCH)/2 - 1

                                ; Extended Functions
00000136 [780F]                         short_addr  _$SET_INTERRUPT_VECTOR          ; 37     25
                                ;----+----+----+----+----+----+----+----+----+----+----+----+----+----+----;
                                ;           C  A  V  E  A  T     P  R  O  G  R  A  M  M  E  R            ;
                                ;                                                                        ;
```

```
3594 00000138 [4F16]                    short_addr  _$CREATE_PROCESS_DATA_BLOCK     ; 38     26
3595                                    ;                                                       ;
3596                                    ;           C  A  V  E  A  T     P  R  O  G  R  A  M  M  E  R            ;
3597                                    ;----+----+----+----+----+----+----+----+----+----+----+----+----+----+----;
3598 0000013A [6822]                    short_addr  _$FCB_RANDOM_READ_BLOCK         ; 39     27
3599 0000013C [6422]                    short_addr  _$FCB_RANDOM_WRITE_BLOCK        ; 40     28
3600 0000013E [D212]                    short_addr  _$PARSE_FILE_DESCRIPTOR         ; 41     29
3601 00000140 [D80A]                    short_addr  _$GET_DATE                      ; 42     2A
3602 00000142 [F50A]                    short_addr  _$SET_DATE                      ; 43     2B
3603 00000144 [140B]                    short_addr  _$GET_TIME                      ; 44     2C
3604 00000146 [250B]                    short_addr  _$SET_TIME                      ; 45     2D
3605 00000148 [C30C]                    short_addr  _$SET_VERIFY_ON_WRITE           ; 46     2E
3606
3607                                    ; Extended functionality group
3608 0000014A [340F]                    short_addr  _$GET_DMA                       ; 47     2F
3609 0000014C [9C0C]                    short_addr  _$GET_VERSION                   ; 48     30
3610 0000014E [1572]                    short_addr  _$KEEP_PROCESS                  ; 49     31
3611                                    ;----+----+----+----+----+----+----+----+----+----+----+----+----+----+----;
3612                                    ;           C  A  V  E  A  T     P  R  O  G  R  A  M  M  E  R            ;
3613                                    ;                                                       ;
3614 00000150 [4413]                    short_addr  _$GET_DPB                       ; 50     32
3615                                    ;                                                       ;
3616                                    ;           C  A  V  E  A  T     P  R  O  G  R  A  M  M  E  R            ;
3617                                    ;----+----+----+----+----+----+----+----+----+----+----+----+----+----+----;
3618 00000152 [3D02]                    short_addr  _$SET_CTRL_C_TRAPPING           ; 51     33
3619 00000154 [2C13]                    short_addr  _$GET_INDOS_FLAG                ; 52     34
3620 00000156 [690F]                    short_addr  _$GET_INTERRUPT_VECTOR          ; 53     35
3621 00000158 [030F]                    short_addr  _$GET_DRIVE_FREESPACE           ; 54     36
3622 0000015A [A50F]                    short_addr  _$CHAR_OPER                     ; 55     37
3623 0000015C [CA0C]                    short_addr  _$INTERNATIONAL                 ; 56     38
3624                                    ; XENIX CALLS
3625                                    ;   Directory Group
3626 0000015E [1C28]                        short_addr  _$MKDIR                     ; 57     39
3627 00000160 [3D27]                        short_addr  _$RMDIR                     ; 58     3A
3628 00000162 [8C27]                        short_addr  _$CHDIR                     ; 59     3B
3629                                    ;   File Group
3630 00000164 [A180]                        short_addr  _$CREAT                     ; 60     3C
3631 00000166 [C97F]                        short_addr  _$OPEN                ; 61     3D
3632 00000168 [6D77]                        short_addr  _$CLOSE                     ; 62     3E
3633 0000016A [7478]                        short_addr  _$READ                ; 63     3F
3634 0000016C [D178]                        short_addr  _$WRITE                     ; 64     40
3635 0000016E [1381]                        short_addr  _$UNLINK                    ; 65     41
3636 00000170 [D678]                        short_addr  _$LSEEK                     ; 66     42
3637 00000172 [AE80]                        short_addr  _$CHMOD                     ; 67     43
3638 00000174 [8428]                        short_addr  _$IOCTL                     ; 68     44
3639 00000176 [3A79]                        short_addr  _$DUP                 ; 69     45
3640 00000178 [5879]                        short_addr  _$DUP2                ; 70     46
3641 0000017A [D926]                        short_addr  _$CURRENT_DIR         ; 71     47
3642                                    ;   Memory Group
3643 0000017C [0E73]                        short_addr  _$ALLOC                     ; 72     48
3644 0000017E [8374]                        short_addr  _$DEALLOC                   ; 73     49
3645 00000180 [5F74]                        short_addr  _$SETBLOCK                  ; 74     4A
3646                                    ;   Process Group
3647 00000182 [E86B]                        short_addr  _$EXEC                ; 75     4B
3648 00000184 [4D72]                        short_addr  _$EXIT                ; 76     4C
3649 00000186 [DE6B]                        short_addr  _$WAIT                ; 77     4D
3650 00000188 [2326]                        short_addr  _$FIND_FIRST          ; 78     4E
3651                                    ;   Special Group
3652 0000018A [7726]                        short_addr  _$FIND_NEXT                 ; 79     4F
3653                                    ; SPECIAL SYSTEM GROUP
3654                                    ;----+----+----+----+----+----+----+----+----+----+----+----+----+----+----;
3655                                    ;           C  A  V  E  A  T     P  R  O  G  R  A  M  M  E  R            ;
3656                                    ;                                                       ;
3657 0000018C [A202]                    short_addr  _$SET_CURRENT_PDB           ; 80     50
3658 0000018E [AE02]                    short_addr  _$GET_CURRENT_PDB           ; 81     51
3659 00000190 [3813]                    short_addr  _$GET_IN_VARS               ; 82     52
3660 00000192 [6514]                    short_addr  _$SETDPB                    ; 83     53
3661                                    ;                                                       ;
3662                                    ;           C  A  V  E  A  T     P  R  O  G  R  A  M  M  E  R            ;
3663                                    ;----+----+----+----+----+----+----+----+----+----+----+----+----+----+----;
3664 00000194 [BE0C]                    short_addr  _$GET_VERIFY_ON_WRITE       ; 84     54
3665                                    ;----+----+----+----+----+----+----+----+----+----+----+----+----+----+----;
3666                                    ;           C  A  V  E  A  T     P  R  O  G  R  A  M  M  E  R            ;
3667                                    ;                                                       ;
3668 00000196 [3E16]                    short_addr  _$DUP_PDB                   ; 85     55
3669                                    ;                                                       ;
3670                                    ;           C  A  V  E  A  T     P  R  O  G  R  A  M  M  E  R            ;
3671                                    ;----+----+----+----+----+----+----+----+----+----+----+----+----+----+----;
3672 00000198 [3981]                    short_addr  _$RENAME                    ; 86     56
3673 0000019A [6679]                    short_addr  _$FILE_TIMES                ; 87     57
3674 0000019C [B874]                    short_addr  _$ALLOCOPER                 ; 88     58
3675
3676                                    ; 08/07/2018 - Retro DOS v3.0
3677                                    ; ---------------------------------------------------------------------------;
3678                                    ; MSDOS 6.0 - MSTABLE.ASM, 1991
3679
3680                                    ; Network extention system calls
3681 0000019E [B90F]                    short_addr  _$GetExtendedError          ; 89     59
3682 000001A0 [C581]                    short_addr  _$CreateTempFile            ; 90     5A
3683 000001A2 [AD81]                    short_addr  _$CreateNewFile             ; 91     5B
3684 000001A4 [8883]                    short_addr  _$LockOper                  ; 92     5C
3685 000001A6 [8375]                    short_addr  _$ServerCall                ; 93     5D
3686 000001A8 [227B]                    short_addr  _$UserOper                  ; 94     5E
3687 000001AA [877A]                    short_addr  _$AssignOper                ; 95     5F
3688 000001AC [647F]                    short_addr  _$NameTrans                 ; 96     60
3689 000001AE [8A06]                short_addr  NO_OP                       ; 97     61
3690 000001B0 [AE02]                    short_addr  _$GET_CURRENT_PDB           ; 98     62
3691                                ; the next call is reserved for hangool sys call
3692                                    ; 29/04/2019 - Retro DOS v4.0 (MSDOS 6.0)
3693 000001B2 [D90F]                    short_addr  _$ECS_Call                  ; 99     63
3694                                    ;short_addr  NO_OP  ;  MSDOS 3.3        ; 99     63
3695                                    ;----+----+----+----+----+----+----+----+----+----+----+----+----+----+----;
3696                                    ;           C  A  V  E  A  T     P  R  O  G  R  A  M  M  E  R            ;
3697                                    ;                                                       ;
3698 000001B4 [BA02]                    short_addr  _$SET_PRINTER_FLAG          ; 100    64
3699                                    ;                                                       ;
3700                                    ;           C  A  V  E  A  T     P  R  O  G  R  A  M  M  E  R            ;
3701                                    ;----+----+----+----+----+----+----+----+----+----+----+----+----+----+----;
3702 000001B6 [870D]                    short_addr  _$GetExtCntry               ; 101    65
3703 000001B8 [B10E]                    short_addr  _$GetSetCdPg                ; 102    66
3704 000001BA [B877]                    short_addr  _$ExtHandle                 ; 103    67
3705 000001BC [A077]                    short_addr  _$COMMIT                    ; 104    68
3706
3707                                ; 08/07/2018
3708                                ; Above system calls are valid for Retro DOS v3.0 (MSDOS 3.3)
3709                                ; Following system calls are valid for Retro DOS v4.0 (MSDOS 6.0)
3710
3711                                ; 29/04/2019 - Retro DOS v4.0 (MSDOS 6.0)
3712 000001BE [4117]                    short_addr  _$GSetMediaID                  ; 105    69   ;AN000;
3713 000001C0 [A077]                    short_addr  _$COMMIT                       ; 106    6A   ;AN000;
3714 000001C2 [8A06]                    short_addr  NO_OP                          ; 107    6B
3715                                                                               ; IFS_IOCTL no longer
3716                                                                               ; supported
3717 000001C4 [7082]                    short_addr  _$Extended_Open                ; 108    6C   ;AN000;
```

```
                                ;----+----+----+----+----+----+----+----+----+----+----+----+----+----+----+----;
                                ;           C  A  V  E  A  T     P  R  O  G  R  A  M  M  E  R                  ;
                                ;                                                                             ;
                                ;ifdef ROMEXEC
                                ;       short_addr  $ROM_FIND_FIRST                 ; 109    6D
                                ;       short_addr  $ROM_FIND_NEXT                  ; 110    6E
                                ;   short_addr  $ROM_EXCLUDE                ; 111    6F          ; M078
                                ;endif
                                ;                                                                             ;
                                ;           C  A  V  E  A  T     P  R  O  G  R  A  M  M  E  R                  ;
                                ;----+----+----+----+----+----+----+----+----+----+----+----+----+----+----+----;

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

                                ; 01/01/2024 - Retro DOS v5.0 (PCDOS 7.1 IBMDOS.COM)

000001C6 [8A06]                     short_addr NO_OP      ; 6Dh, OS/2 "DosMkDir2" - ROM DOS: Find first ROM program
000001C8 [8A06]                     short_addr NO_OP      ; 6Eh, OS/2 "DosEnumAttrib" - ROM DOS: Find next ROM program
000001CA [8A06]                     short_addr NO_OP      ; 6Fh, OS/2 "DosQMaxEASize" - ROM DOS: Get/set searched ROM area
000001CC [4D0D]                     short_addr _$ExtCountryInfo ; 70h, MSDOS 7 (WIN 95) - Get/set extended country information
                                                          ; GET/SET INTERNATIONALIZATION INFORMATION
000001CE [F40F]                     short_addr _$LONGNAME ; 71h, MSDOS 7 (WIN 95) LONG FILENAME FUNCTIONS
000001D0 [F40F]                     short_addr _$LONGNAME ; 72h, MSDOS 7 (WIN 95) LFN-FindClose
000001D2 [0110]                     short_addr _$FAT32EXT ; 73h, MSDOS 7 - FAT32 extended drive functions

                                ;MAXCOM  = ($-DISPATCH)/2 - 1

                                MAXCOM  EQU ($-DISPATCH)/2 - 1

                                ; 08/07/2018 - Retro DOS v3.0
                                ; MSDOS 6.0 - MSTABLE.ASM, 1991

                                ;   If Installed

                                align 2

                                ;PUBLIC FOO

                                FOO:; LABEL WORD
000001D4 [4707]                         short_addr Leave2F

000001D6 [D801]                 DTab:       DW DOSTable

                                    ;PUBLIC FOO,DTAB

                                    ; IBMDOS.COM (MSDOS 3.3) - Offset 0ED6h

                                ; 29/04/2019
                                ; DOSCODE:3F7Ch (MSDOS 6.21, MSDOS.SYS)

                                ; 04/11/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
                                ; DOSCODE:3F6Ch (MSDOS 5.0, MSDOS.SYS)

                                ; 01/01/2024 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMDOS.COM)
                                ; DOSCODE:40BEh (PCDOS 7.1, IBMDOS.COM)

                                DOSTable:  ; LABEL  WORD
000001D8 32                             DB  (DOSTableEnd-DOSTable-1)/2 ; db 50 ; 01/01/2024
000001D9 [B909]                         short_addr  DOSInstall          ;   0 install check
000001DB [2837]                         short_addr  DOS_CLOSE           ;   1   DOS_CLOSE
000001DD [9A0F]                         short_addr  RECSET              ;   2   RECSET
000001DF [B309]                         short_addr  DosGetGroup         ;   3   Get DOSGROUP
000001E1 [EA5D]                         short_addr  PATHCHRCMP          ;   4   PATHCHRCMP
000001E3 [311C]                         short_addr  OUTT                ;   5   OUT
000001E5 [FA60]                         short_addr  NET_I24_ENTRY       ;   6   NET_I24_ENTRY
000001E7 [D268]                         short_addr  PLACEBUF            ;   7   PLACEBUF
000001E9 [8C38]                         short_addr  FREE_SFT            ;   8   FREE_SFT
000001EB [1C6B]                         short_addr  BUFWRITE            ;   9   BUFWRITE
000001ED [4C84]                         short_addr  SHARE_VIOLATION     ;   10  SHARE_VIOLATION
000001EF [FD32]                         short_addr  SHARE_ERROR         ;   11  SHARE_ERROR
000001F1 [E732]                         short_addr  SET_SFT_MODE        ;   12  SET_SFT_MODE
000001F3 [600B]                         short_addr  DATE16              ;   13  DATE16
000001F5 [E417]                         short_addr  Idle          ;   14      empty slot
000001F7 [CB68]                         short_addr  SCANPLACE           ;   15  SCANPLACE
000001F9 [E417]                         short_addr  Idle          ;   16      empty slot
000001FB [B117]                         short_addr  StrCpy              ;   17  StrCpy
000001FD [C917]                         short_addr  StrLen              ;   18  StrLen
000001FF [975D]                         short_addr  UCase         ;   19  UCase
00000201 [0F69]                         short_addr  POINTCOMP           ;   20  POINTCOMP
00000203 [E76A]                         short_addr  CHECKFLUSH          ;   21  CHECKFLUSH
00000205 [E976]                         short_addr  SFFromSFN           ;   22  SFFromSFN
00000207 [DC7B]                         short_addr  GetCDSFromDrv       ;   23  GetCDSFromDrv
00000209 [7704]                         short_addr  Get_User_Stack      ;   24  Get_User_Stack
0000020B [8D7B]                         short_addr  GETTHISDRV          ;   25  GetThisDrv
0000020D [897F]                         short_addr  DriveFromText       ;   26  DriveFromText
0000020F [030C]                         short_addr  SETYEAR             ;   27  SETYEAR
00000211 [920C]                         short_addr  DSUM                ;   28  DSUM
00000213 [F70B]                         short_addr  DSLIDE              ;   29  DSLIDE
00000215 [8F17]                         short_addr  StrCmp              ;   30  StrCmp
00000217 [D57A]                         short_addr  InitCDS             ;   31  initcds
00000219 [B676]                         short_addr  pJFNFromHandle      ;   32  pJfnFromHandle
0000021B [647F]                         short_addr  _$NameTrans          ;   33  $NameTrans
0000021D [B106]                         short_addr  CAL_LK              ;   34  CAL_LK
0000021F [D44D]                         short_addr  DEVNAME             ;   35  DEVNAME
00000221 [E417]                         short_addr  Idle                ;   36  Idle
00000223 [D717]                         short_addr  DStrLen             ;   37  DStrLen
00000225 [8618]                         short_addr  NLS_OPEN            ;   38  NLS_OPEN      DOS 3.3
00000227 [6D77]                         short_addr  _$CLOSE              ;   39  $CLOSE        DOS 3.3
00000229 [8C18]                         short_addr  NLS_LSEEK           ;   40  NLS_LSEEK     DOS 3.3
0000022B [7478]                         short_addr  _$READ        ;   41  $READ         DOS 3.3
0000022D [4818]                         short_addr  FastInit            ;   42  FastInit      DOS 3.4  ;AN000;
0000022F [C918]                         short_addr  NLS_IOCTL           ;   43  NLS_IOCTL     DOS 3.3
00000231 [B818]                         short_addr  GetDevList          ;   44  GetDevList    DOS 3.3
00000233 [DD18]                         short_addr  NLS_GETEXT          ;   45  NLS_GETEXT    DOS 3.3

                                    ; 29/04/2019 - Retro DOS v4.0
00000235 [E118]                     short_addr  MSG_RETRIEVAL    ;   46  MSG_RETRIEVAL DOS 4.0  ;AN000;

00000237 [8A06]                     short_addr  NO_OP            ;   M006: 47  no longer supported
                                ;***        short_addr  Fake_Version     ;   47  Fake_Version  DOS 4.0  ;AN006;

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

                                    ; 01/01/2024 - Retro DOS v5.0 (PCDOS 7.1)
00000239 [A04A]                     short_addr  int_2Fh_1230h    ;   48
                                                          ; FIND SFT ENTRY IN INTERNAL FILE TABLES
0000023B [6C09]                     short_addr  int_2Fh_1231h    ;   49
                                                          ; SET/CLEAR REPORT WINDOWS TO DOS PROGRAMS FLAG

                                DOSTableEnd:  ; LABEL BYTE

                                    ;ENDIF
```

```
                        ; ----------------------------------------------------------------------------
                        ; BREAK   <Copyright notice and version>
                        ; ----------------------------------------------------------------------------

                        ;CODSTRT EQU       $

                        ; 08/07/2018 - Retro DOS v3.0 by Erdogan Tan
                        ; (MSTABLE.ASM, MSDOS 6.0, 1991)

                        ; NOTE WARNING: This declaration of HEADER must be THE LAST thing in this
                        ;       module. The reason is so that the data alignments are the same in
                        ;       IBM-DOS and MS-DOS up through header.

                            ;PUBLIC HEADER

                        HEADER:     ; LABEL BYTE
                                ;IF DEBUG
                                ;DB 13,10,"Debugging DOS version "
                                ;DB MAJOR_VERSION + "0"
                                ;DB "."
                                ;DB (MINOR_VERSION / 10) + "0"
                                ;DB (MINOR_VERSION MOD 10) + "0"
                                ;ENDIF

                        ; 05/12/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
                        ; (MSDOS 5.0 MSDOS.SYS compatibility)
                        %if 0
                                ;IF NOT IBM
                                DB  13,10,"MS-DOS version "
                                DB  MAJOR_VERSION + "0"
                                DB  "."
                                DB  (MINOR_VERSION / 10) + "0"
                                ;DB (MINOR_VERSION MOD 10) + "0"
                                DB  (MINOR_VERSION % 10) + "0"

                                ;IF HIGHMEM
                                ;DB "H"
                                ;ENDIF

                                ;DB     13,10,"Copyright 1981,82,83,84,88 Microsoft Corp.",13,10,"$"
                                ; 30/04/2019 - Retro DOS v4.0
                                DB      13,10,"Copyright 1981-1993 Microsoft Corp.",13,10,"$"

                                ;ENDIF

                        %endif

                        ;IF DEBUG
                        ;   DB      13,10,"$"
                        ;ENDIF

                        ;include copyrigh.inc

                        ; DOSCODE:3FDDh (MSDOS 6.21, MSDOS.SYS)

                            ;DB     "MS DOS Version 6 (C)Copyright 1981-1993 Microsoft Corp "
                            ;DB     "Licensed Material - Property of Microsoft "
                            ;DB     "All rights reserved "

                        ; 05/12/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
                        ; DOSCODE:3FCDh (MSDOS 5.0, MSDOS.SYS)

                        ; 28/12/2022 - Retro DOS v4.1
                        %if 0
                            ; 05/12/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
                        ms_copyright:
                            db      'MS DOS Version 5.00 (C)Copyright 1981-1991 Microsoft Corp '
                            db      'Licensed Material - Property of Microsoft '
                            db      'All rights reserved '

                        %endif
                            ;; 28/12/2022 - Retro DOS v4.1
                        ;ms_copyright:
                            ;db     13,10,"MS DOS Version 5.0"
                            ;db     13,10,"Copyright 1981-1991 Microsoft Corp.",13,10,"$",0

                        ;   ; 21/09/2023 - Retro DOS v4.2 MSDOS.SYS
                        ;   ; (MSDOS 6.22 MSDOS.SYS - DOSCODE:3FDDh (File offset: 509))
                        ;ms_copyright:
                        ;   db 'MS DOS Version 6 (C)Copyright 1981-1994 Microsoft Corp '
                        ;   db 'Licensed Material - Property of Microsoft All rights reserved '

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

                            ; 20/09/2023 - Retro DOS v4.2
                        ;ms_copyright:
                            ;db     13,10,"MS DOS Version 6.22"
                            ;db     13,10,"Copyright 1981-1994 Microsoft Corp.",13,10,"$",0

                        ;============================================================================
                        ; MSCODE.ASM
                        ;============================================================================

                        ; Retro DOS v2.0 (NASM 2.11) source code modifications by Erdogan Tan
                        ; 03/03/2018

                        ;
                        ; MSCODE.ASM -- MSDOS code
                        ;

                        ;INCLUDE DOSSEG.ASM
                        ;INCLUDE STDSW.ASM

                        ;CODE    SEGMENT BYTE PUBLIC  'CODE'
                        ;ASSUME  CS:DOSGROUP,DS:NOTHING,ES:NOTHING,SS:NOTHING

                        ;.xcref
                        ;INCLUDE DOSSYM.ASM
                        ;INCLUDE DEVSYM.ASM
                        ;.cref
                        ;.list

                        ;IFNDEF  KANJI
                        ;KANJI   EQU     0       ; FALSE
                        ;ENDIF

                        ;IFNDEF  IBM
                        ;IBM     EQU     0
                        ;ENDIF

                        ;IFNDEF  HIGHMEM
                        ;HIGHMEM  EQU     0
                        ;ENDIF
```

```
                                ;i_need  USER_SP,WORD
                                ;i_need  USER_SS,WORD
                                ;i_need  SAVEDS,WORD
                                ;i_need  SAVEBX,WORD
                                ;i_need  INDOS,BYTE
                                ;i_need  NSP,WORD
                                ;i_need  NSS,WORD
                                ;i_need  CURRENTPDB,WORD
                                ;i_need  AUXSTACK,BYTE
                                ;i_need  CONSWAP,BYTE
                                ;i_need  IDLEINT,BYTE
                                ;i_need  NOSETDIR,BYTE
                                ;i_need  ERRORMODE,BYTE
                                ;i_need  IOSTACK,BYTE
                                ;i_need  WPERR,BYTE
                                ;i_need  DSKSTACK,BYTE
                                ;i_need  CNTCFLAG,BYTE
                                ;i_need  LEAVEADDR,WORD
                                ;i_need  NULLDEVPT,DWORD

                                ;IF NOT IBM
                                ;i_need  OEM_HANDLER,DWORD
                                ;ENDIF

                                ;EXTRN   DSKSTATCHK:NEAR,GETBP:NEAR,DSKREAD:NEAR,DSKWRITE:NEAR

                        ;==========================================================================
                        ; MSDISP.ASM, MSDOS 6.0, 1991
                        ;==========================================================================
                        ; 11/07/2018 - Retro DOS v3.0
                        ; 01/05/2019 - Retro DOS v4.0

                        ; DosCode SEGMENT

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

                        ; 04/11/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
                        ; DOSCODE:4045h (MSDOS 5.0, MSDOS.SYS)

                        ; 01/01/2024 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMDOS.COM)
                        ; DOSCODE:4123h (PCDOS 7.1, IBMDOS.COM)

                        _$SET_CTRL_C_TRAPPING:
                            ; 01/05/2019 - Retro DOS v4.0

0000023D 3C07                   cmp     al,7    ; 01/01/2024 - Retro DOS v5.0
                                ;cmp    AL,6                    ; Is this a valid subfunction?
0000023F 7603                   jbe     short scct_1            ; If yes continue processing

00000241 B0FF                   mov     AL,0FFh                 ; Else set AL to -1 and
00000243 CF                     iret
                        scct_1:
00000244 1E                     push    DS

                                ;getdseg <DS>                   ; DS -> DosData, ASSUME DS:DosSeg
00000245 2E8E1E[0700]           mov     ds,[cs:DosDSeg]

0000024A 50                     push    AX                      ; DL only register that can change
0000024B 56                     push    SI

0000024C BE[3703]               mov     SI,CNTCFLAG             ; DS:SI --> Ctrl C Status byte
0000024F 30E4                   xor     AH,AH                   ; Clear high byte of AX
00000251 09C0                   or      AX,AX                   ; Check for subfunction 0
00000253 7504                   jnz     short scct_2            ; If not 0 jmp to next check

00000255 8A14                   mov     DL,[SI]                 ; Else move current ctrl C status
00000257 EB32                   jmp     SHORT scct_9s           ; into DL and jmp to exit
                        scct_2:
00000259 48                     dec     AX                      ; Now dec AX and see if it was 1
0000025A 7507                   jnz     short scct_3            ; If not 0 it wasn't 1 so do next chk

0000025C 80E201                 and     DL,1                    ; Else mask off bit 0 of DL and
0000025F 8814                   mov     [SI],DL                 ; save it as new Ctrl C status
00000261 EB28                   jmp     SHORT scct_9s           ; Jmp to exit
                        scct_3:
00000263 48                     dec     AX                      ; Dec AX again to see if it was 2
00000264 7507                   jnz     short scct_4            ; If not 0 wasn't 2 so go to next chk

00000266 80E201                 and     DL,1                    ; Else mask off bit 0 of DL and
00000269 8614                   xchg    [SI],DL                 ; Exchange DL with old status byte
0000026B EB1E                   jmp     SHORT scct_9s           ; Jump to exit (returning old status)
                        scct_4:
0000026D 3C03                   cmp     al,3 ; 01/01/2024
                                ;cmp    AX,3                    ; Test for 5 after it was dec twice
0000026F 7506                   jne     short scct_5            ; If not equal then not get boot drv
00000271 8A16[6900]             mov     DL,[BOOTDRIVE]          ; Else return boot drive in DL
00000275 EB14                   jmp     SHORT scct_9s           ; Jump to exit (returning boot drive)
                        scct_5:
                                ; 01/01/2024 - Retro DOS v5.0
00000277 7212                   jb      short scct_9s ; PCDOS 7.1
                                ;
00000279 3C04                   cmp     al,4 ; 01/01/2024
                                ;cmp    AX,4                    ; Test for 6 after it was dec twice
                                ;jne    short scct_9s           ; If not equal then not get version
0000027B 7512                   jne     short scct_6 ; 01/01/2024 ; PCDOS 7.1

                                ;mov    BX,(Minor_Version SHL 8) + Major_Version

                                ;;mov   bx,1406h ; 6.20         ; DOSCODE:4092h (MSDOS 6.21, MSDOS.SYS)
                                ;;mov   bx,1606h ; 6.22         ; DOSCODE:4092h (MSDOS 6.22, MSDOS.SYS)

                                ;mov    bx,0A07h ; 7.10         ; DOSCODE:4163h (PCDOS 7.1, IMBDOS.COM)
0000027D BB070A                 mov     bx,(MINOR_VERSION<<8)+MAJOR_VERSION  ; true dos version

                                ;mov    dl,0                    ; revision 0
                                ;mov    DL,DOSREVNM ; 0

                                ;xor    dh,dh                   ; assume vanilla DOS
                                ; 01/01/2024
00000280 BA0000                 mov     dx,0
00000283 3836[660D]             cmp     byte [DosHasHMA],dh ; 0
                                ;cmp    byte [DosHasHMA],0      ; is DOS in HMA?  (M021)
                                ;;je    short @F
                                ;je     short scct_6
00000287 7402                   je      short scct_9s ; 01/01/2024
                                ; 01/01/2024
00000289 B610                   mov     dh,10h                  ; version flags bit 4
                                ;or     DH,DOSINHMA ; 10h       ; 'DOS in HMA' status
                        ;@@:
                        ;scct_6:
                        ;ifdef ROMDOS
                        ;   or      DH,DOSINROM ; 08h
                        ;endif ; ROMDOS
```

```
                                    ; 01/01/2024 (PCDOS 7.1)
                                    ;jmp    short short scct_9s

                                    ; 01/01/2024
                                ;scct_6:
                                    ; ....

                                scct_9s:
0000028B 5E                         pop     SI
0000028C 58                         pop     AX
0000028D 1F                         pop     DS
                                scct_9f:
0000028E CF                         iret

                                    ; 01/01/2024 - Retro DOS v5.0 (PCDOS 7.1 IBMDOS.COM)
                                scct_6:
0000028F 8026[8600]DF               and     byte [DOS_FLAG],0DFh  ; clear bit 5 of DOS flag
00000294 80FA01                     cmp     dl, 1
00000297 75F2                       jne     short scct_9s
00000299 800E[8600]20               or      byte [DOS_FLAG],20h   ; set bit 5 of DOS flag
0000029E EBEB                       jmp     short scct_9s

                                SetCtrlShortEntry:              ; This allows a conditional entry
                                                                ; from main dispatch code
000002A0 EB9B                       jmp     SHORT _$SET_CTRL_C_TRAPPING

                                ; ==========================================================================
                                ;
                                ; The following two routines are dispatched to directly with ints disabled
                                ; immediately after the int 21h entry.   no DIS state is set.
                                ;
                                ; $Set_current_PDB takes BX and sets it to be the current process
                                ;   *** THIS FUNCTION CALL IS SUBJECT TO CHANGE!!! ***
                                ;
                                ; ==========================================================================

                                _$SET_CURRENT_PDB:
000002A2 1E                         push    DS
                                    ;getdseg <DS>               ; DS -> DosData, ASSUME DS:DosSeg
000002A3 2E8E1E[0700]               mov     ds,[cs:DosDSeg]
000002A8 891E[3003]                 mov     [CurrentPDB],BX             ; Set new PSP segment from caller's BX
000002AC 1F                         pop     DS
000002AD CF                         iret

                                ; ==========================================================================
                                ;
                                ; $get_current_PDB returns in BX the current process
                                ;   *** THIS FUNCTION CALL IS SUBJECT TO CHANGE!!! ***
                                ;
                                ; ==========================================================================

                                _$GET_CURRENT_PDB:
000002AE 1E                         push    DS
                                    ;getdseg <DS>               ; DS -> DosData, ASSUME DS:DosSeg
000002AF 2E8E1E[0700]               mov     ds,[cs:DosDSeg]
000002B4 8B1E[3003]                 mov     BX,[CurrentPDB]             ; Return current PSP segment in BX
000002B8 1F                         pop     DS
000002B9 CF                         iret

                                ; ==========================================================================
                                ;
                                ; Sets the Printer Flag to whatever is in AL.
                                ; NOTE: THIS PROCEDURE IS SUBJECT TO CHANGE!!!
                                ;
                                ; ==========================================================================

                                _$SET_PRINTER_FLAG:
000002BA 1E                         push    ds
                                    ;getdseg <DS>               ; DS -> DosData, ASSUME DS:DosSeg
000002BB 2E8E1E[0700]               mov     ds,[cs:DosDSeg]
000002C0 A2[A00A]                   mov     [PRINTER_FLAG],AL     ; Set printer flag from caller's AL
000002C3 1F                         pop     ds
000002C4 CF                         iret

                                ; 01/05/2019 - Retro DOS v4.0
                                ; 08/07/2018 - Retro DOS v3.0
                                ; (MSDISP.ASM, MSDOS 6.0, 1991)

                                ; ---------------------------------------------------------------------------
                                ; BREAK   <System call entry points and dispatcher>
                                ; ---------------------------------------------------------------------------

                                ; DOSCODE:40CCh (MSDOS 6.21, MSDOS.SYS)

                                ; ==========================================================================
                                ;
                                ; The Quit entry point is where all INT 20h's come from. These are old- style
                                ; exit system calls. The CS of the caller indicates which Process is dying.
                                ; The error code is presumed to be 0. We simulate an ABORT system call.
                                ;
                                ; ==========================================================================

                                SYSTEM_CALL:    ; PROC NEAR

                                ; 04/11/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
                                ; DOSCODE:40BFh (MSDOS 5.0, MSDOS.SYS)

                                ;entry     QUIT
                                QUIT:                           ; INT 20H entry point
                                    ;MOV    AH,0
000002C5 30E4                       xor     ah,ah ; 08/07/2018
000002C7 EB36                       JMP     SHORT SAVREGS

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

                                    ; The system call in AH is out of the range that we know how
                                    ; to handle. We arbitrarily set the contents of AL to 0 and
                                    ; IRET. Note that we CANNOT set the carry flag to indicate an
                                    ; error as this may break some programs compatability.

                                BADCALL:
                                        ;MOV       AL,0
000002C9 30C0                       xor     al,al ; 08/07/2018
                                IRETT:      ; 06/05/2019
                                _IRET:
000002CB CF                             IRET

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

                                ; 01/05/2019 - Retro DOS v4.0
                                ; DOSCODE:40D3h (MSDOS 6.21 MSDOS.SYS)
                                ; 04/11/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
                                ; DOSCODE:40C6h (MSDOS 5.0 MSDOS.SYS)
```

```
                                    ; An alternative method of entering the system is to perform a
                                    ; CALL 5 in the program segment prefix with the contents of CL
                                    ; indicating what system call the user would like. A subset of
                                    ; the possible system calls is allowed here only the
                                    ; CPM-compatible calls may get dispatched.

                                            ; System call entry point and dispatcher
                                CALL_ENTRY:
000002CC 1E                         push    DS
                                    ;getdseg <DS>                  ; DS -> DosData, ASSUME DS:DosSeg
000002CD 2E8E1E[0700]               mov     ds,[cs:DosDSeg]
000002D2 8F06[EC05]                 pop     word [SAVEDS]          ; Save original DS

000002D6 58                             POP     AX                      ; IP from the long call at 5
000002D7 58                             POP     AX                      ; Segment from the long call at 5
000002D8 8F06[8405]                 POP     WORD [USER_SP]         ; IP from the CALL 5

                                            ; Re-order the stack to simulate an interrupt 21.

000002DC 9C                         PUSHF                          ; Start re-ordering the stack
000002DD FA                         CLI
000002DE 50                             PUSH    AX                      ; Save segment
000002DF FF36[8405]                     PUSH       WORD [USER_SP]        ; Stack now ordered as if INT had been used
                                    ; 04/11/2022
                                    ; DOSCODE:40EAh (MSDOS 6.21 MSDOS.SYS)
                                    ; DOSCODE:40DDh (MSDOS 5.0 MSDOS.SYS)
000002E3 FF36[EC05]                 push    word [SAVEDS]
000002E7 1F                         pop     ds
                                    ;
                                    ;cmp    cl,36
000002E8 80F924                         CMP     CL,MAXCALL             ; This entry point doesn't get as many calls
000002EB 77DC                           JA      SHORT BADCALL
000002ED 88CC                           MOV     AH,CL
                                    ; 08/07/2018
000002EF EB0E                       jmp     short SAVREGS

                                ; ----------------------------------------------------------------------------
                                ; 01/05/2019 - Retro DOS v4.0
                                ; 01/01/2024 - Retro DOS v5.0

                                    ; This is the normal INT 21 entry point. We first perform a
                                    ; quick test to see if we need to perform expensive DOS-entry
                                    ; functions. Certain system calls are done without interrupts
                                    ; being enabled.

                                    ;entry  COMMAND                ; Interrupt call entry point (int 21h)

                                ; DOSCODE:40F8h (MSDOS 6.21, MSDOS.SYS)
                                ; 04/11/2022
                                ; DOSCODE:40EBh (MSDOS 5.0, MSDOS.SYS)
                                ; 01/01/2024
                                ; DOSCODE:41D7h (PCDOS 7.1, IBMDOS.COM)

                                COMMAND:
                                    ; 22/12/2022
000002F1 FA                         cli

                                    ; 01/05/2019 - Retro DOS v4.0
                                    ; 08/07/2018 - Retro DOS v3.0

                                ; 22/12/2022
                                ; 04/11/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
                                ; 01/01/2024 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMDOS.COM)

                                    ;IF     NOT IBM
000002F2 80FCF8                     CMP     AH,SET_OEM_HANDLER
000002F5 7203                       JB      SHORT NOTOEM
000002F7 E98701                     JMP     _$SET_OEM_HANDLER

                                NOTOEM:
                                    ;ENDIF

                                ; DOSCODE:40F8h (MSDOS 6.21, MSDOS.SYS)
                                ; DOSCODE:40EBh (MSDOS 5.0, MSDOS.SYS)
                                ; 01/01/2024 - Retro DOS v5.0
                                ; DOSCODE:41D7h (PCDOS 7.1, IBMDOS.COM)

                                    ; 22/12/2022
                                    ;cli    ; 08/07/2018

                                    ; 01/01/2024
                                _COMMAND: ; MSDOS 3.3 (IBMDOS)
                                    ;;cmp   ah,6Ch  ; MSDOS 6.21
                                    ;cmp    ah,73h  ; PCDOS 7.1 ; Max int 21h function call number
                                    ; 04/11/2022
000002FA 80FC73                     CMP     AH,MAXCOM ; 6Ch for MSDOS 6.0 (6.21,6.22) & MSDOS 5.0
                                                      ; 73h for PCDOS 7.1
                                    ;JBE    SHORT SAVREGS
000002FD 77CA                           JA  SHORT BADCALL ; 08/07/2018

                                    ; 31/05/2019

                                    ; The following set of calls are issued by the server at
                                    ; *arbitrary* times and, therefore, must be executed on
                                    ; the user's entry stack and executed with interrupts off.

                                SAVREGS:
                                    ; 01/05/2019 - Retro DOS v4.0
                                    ; 10/08/2018
                                    ; 08/07/2018 - Retro DOS v3.0
000002FF 80FC33                     cmp     ah,33h                 ; Check Minimum special case #
                                    ;;je    _$SET_CTRL_C_TRAPPING
                                    ;je     short SetCtrlShortEntry ; If equal jmp directly to function
00000302 7218                       jb      short SaveAllRegs      ; Not special case so continue
                                    ; 04/11/2022
00000304 749A                       je      short SetCtrlShortEntry ; If equal jmp directly to function
00000306 80FC64                     cmp     ah,64h                 ; Check Max case number
00000309 7711                       ja      short SaveAllRegs      ; Not special case so continue
0000030B 74AD                       je      short _$SET_PRINTER_FLAG ; If equal jmp directly to function
0000030D 80FC51                     cmp     ah,51h                 ; Is this a Get PSP call (51h)?
00000310 749C                       je      short _$GET_CURRENT_PDB       ; Yes, jmp directly to function
                                    ; 25/06/2024
00000312 80FC50                     cmp      ah,50h                ; Is this a Set PSP call (50h) ?
00000315 748B                       je      short _$SET_CURRENT_PDB       ; Yes, jmp directly to function
00000317 80FC62                     cmp     ah,62h                 ; Is this a Get PSP call (62h)?
0000031A 7492                       je      short _$GET_CURRENT_PDB       ; Yes, jmp directly to function

                                SaveAllRegs:
                                    ; 01/05/2019 - Retro DOS v4.0
                                    ; 01/01/2024 - Retro DOS v5.0

0000031C 06                             push       ES
0000031D 1E                         push    DS
```

```
                                        push    BP
                                        push    DI
                                        push    SI
                                        push    DX
                                        push    CX
                                        push    BX
                                        push    AX

                                        mov     AX,DS
                                        ;getdseg <DS>                   ; DS -> DosData, ASSUME DS:DosSeg
                                        mov     ds,[cs:DosDSeg]
                                        mov     [SAVEDS],AX             ; save caller's DS
                                        mov     [SAVEBX],BX

                                            ;INC     BYTE [INDOS]               ; Flag that we're in the DOS

                                        ; 08/07/2018 - Retro DOS v3.0
                                        ;xor     ax,ax
                                        ;mov     [USER_ID],ax
                                        ;mov     ax,[CurrentPDB]
                                        ;mov     [PROC_ID],ax

                                        ; 01/05/2019

                                        ; Note: Nsp and Nss have to be unconditionally initialized here
                                        ; even if InDOS is zero. Programs like CROSSTALK 3.7 depend on
                                        ; this!!!

                                        MOV     AX,[USER_SP]
                                            MOV     [NSP],AX
                                            MOV     AX,[USER_SS]
                                            MOV     [NSS],AX

                                        xor     AX,AX ; 0
                                        mov     [FSHARING],AL           ; allow redirection

                                        test    byte [IsWin386],1       ; WIN386 patch. Do not update USER_ID
                                        jnz     short set_indos_flag  ; if win386 present
                                        mov     [USER_ID],AX
                                    set_indos_flag:
                                        INC     BYTE [INDOS]            ; Flag that we're in the DOS

                                        ; 01/01/2024 (PCDOS 7.1 IBMDOS.COM)
                                        inc     byte [INDOS_FLAG]       ; duplicated INDOS flag (what for ?)

                                            MOV     [USER_SP],SP
                                            MOV     [USER_SS],SS

                                        mov     AX,[CurrentPDB]
                                        mov     [PROC_ID],AX
                                        mov     DS,AX
                                        pop     AX
                                        push    AX

                                        ; save user stack in his area for later returns (possibly from EXEC)

                                            MOV     [PDB.USER_STACK],SP       ; mov [2Eh], sp
                                            MOV     [PDB.USER_STACK+2],SS     ; mov [30h], ss

                                        ; 18/07/2018
                                        ;mov    byte [CS:FSHARING], 0

                                        ;MOV     BX,CS                  ; no holes here.
                                        ;MOV     SS,BX

                                        ;getdseg <ss>                   ; ss -> dosdat, already flag is CLI
                                        mov     ss,[cs:DosDSeg]
                                                                        ;entry  REDISP
                                    REDISP:
                                            MOV     SP,AUXSTACK                ; Enough stack for interrupts
                                            STI                               ; stack is in our space now...

                                        mov     bx,ss
                                        mov     ds,bx

                                        xchg    ax,bx

                                        xor     ax,ax ; 0

                                        ; 04/11/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
                                        ; MSDOS 5.0 MSDOS.SYS - DOSCODE:416Eh  (from org 3DD0h)
                                        ; MSDOS 6.21 MSDOS.SYS - DOSCODE:417Bh (from org 3DE0h)

                                        ; (Note: ss: segment prefix was not needed here! ds=ss ! -04/11/2022-)

                                        ;mov    [ss:EXTOPEN_ON],al ; 0; Clear extended open flag
                                        ;;and   word [ss:DOS34_FLAG],EXEC_AWARE_REDIR
                                        ;and    word [ss:DOS34_FLAG],800h ; clear all bits except bit 11
                                        ;mov    [ss:CONSWAP],al  ; 0  ; random clean up of possibly mis-set flags
                                        ;mov    [ss:NoSetDir],al ; 0  ; set directories on search
                                        ;mov    [ss:FAILERR],al ; 0   ; FAIL not in progress
                                        ;inc    ax
                                        ;;inc   AL                      ; AL = 1
                                        ;mov    [ss:IDLEINT],al                ; presume that we can issue INT 28h

                                        ; 15/12/2022
                                        mov     [EXTOPEN_ON],al ; 0   ; Clear extended open flag
                                        ;and    word [DOS34_FLAG],EXEC_AWARE_REDIR
                                        and     word [DOS34_FLAG],800h; clear all bits except bit 11
                                        mov     [CONSWAP],al  ; 0     ; random clean up of possibly mis-set flags
                                        ;mov    byte [IDLEINT],1
                                        mov     [NoSetDir],al ; 0     ; set directories on search
                                        mov     [FAILERR],al ; 0      ; FAIL not in progress
                                        inc     ax
                                        ;inc    al                      ; AL = 1
                                        mov     [IDLEINT],al            ; presume that we can issue INT 28h

                                        XCHG    AX,BX                   ; Restore AX and BX = 1

                                        MOV     BL,AH
                                            SHL     BX,1                       ; 2 bytes per call in table

                                        CLD
                                                ; Since the DOS maintains mucho state information across system
                                                ; calls, we must be very careful about which stack we use.
                                                ; First, all abort operations must be on the disk stack. This
                                                ; is due to the fact that we may be hitting the disk (close
                                                ; operations, flushing) and may need to report an INT 24.

                                        OR      AH,AH
                                            JZ      SHORT DSKROUT              ; ABORT

                                            ;CMP       AH,12
                                            ;JBE       SHORT IOROUT           ; Character I/O
```

```
                                    ;CMP       AH,GET_CURRENT_PDB      ; INT 24h needs GET,SET PDB
                                    ;JZ SHORT IOROUT
                                    ;CMP       AH,SET_CURRENT_PDB
                                    ;JNZ       SHORT DSKROUT

                                        ; Second, PRINT and PSPRINT and the server issue
                                        ; GetExtendedError calls at INT 28 and INT 24 time.
                                        ; This call MUST, therefore, use the AUXSTACK.

                                ; 10/08/2018
000003A0 80FC59                 cmp     ah,GETEXTENDEDERROR ; 59h
000003A3 7439                   je      short DISPCALL

                                ; 01/05/2019

                                        ; Old 1-12 system calls may be either on the IOSTACK (normal
                                        ; operation) or on the AUXSTACK (at INT 24 time).

000003A5 80FC0C                 cmp     ah,12 ; STD_CON_INPUT_FLUSH ; 0Ch
000003A8 770C                   ja      short DSKROUT

                               IOROUT:
                                    ; 04/11/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
                                    ; (ss: prefix was not needed here! ds=ss)
                                    ;cmp   byte [ss:ERRORMODE],0 ; Are we in an INT 24h?
                                    ; 15/12/2022
000003AA 803E[2003]00               cmp     BYTE [ERRORMODE],0    ; Are we in an INT 24h?
000003AF 752D                           JNZ     SHORT DISPCALL          ; Stay on AUXSTACK if INT 24h
000003B1 BC[A00A]                       MOV     SP,IOSTACK
000003B4 EB28                           JMP     SHORT DISPCALL

                                        ; We are on a system call that is classified as "the rest".
                                        ; We place ourselves onto the DSKSTACK and away we go.
                                        ; We know at this point:
                                        ; *  An INT 24 cannot be in progress. Therefore we reset
                                        ;    ErrorMode and WpErr
                                        ; *  That there can be no critical sections in effect.
                                        ;    We signal the server to remove all the resources.

                               DSKROUT:
                                    ; 01/01/2024 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMDOS.COM)
                                    ; 15/12/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
                                    ; 08/07/2018 - Retro DOS v3.0
000003B6 A3[3A03]                   mov     [USER_IN_AX],ax             ; Remember what user is doing
                                    ; 01/01/2024
                                    ;mov    byte [EXTERR_LOCUS],1 ; errLOC_Unk (Default)
                                    ;MOV    BYTE [WPERR],-1             ; error mode, so good place to
                                                                        ; make sure flags are reset
000003B9 C706[2203]FF01             mov     word [WPERR],1FFh

000003BF C606[2003]00               MOV     BYTE [ERRORMODE],0    ; Cannot make non 1-12 calls in


                                    ; 04/11/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
                                    ; (ss: prefix was not needed here! ds=ss)

                                    ;mov    [ss:USER_IN_AX],ax    ; Remember what user is doing
                                    ;mov    byte [ss:EXTERR_LOCUS],1 ; errLOC_Unk (Default)
                                    ;mov    byte [ss:ERRORMODE],0 ; Cannot make non 1-12 calls in
                                    ;mov    byte [ss:WPERR],-1    ; error mode, so good place to
                                                                        ; make sure flags are reset
000003C4 50                         push    ax
000003C5 B482                       mov     ah,82h              ; Release all resource information
000003C7 CD2A                       int     2Ah                 ; Microsoft Networks
                                                                ; END DOS CRITICAL SECTIONS 0 THROUGH 7
000003C9 58                         pop     ax

                                        ; Since we are going to be running on the DSKStack and since
                                        ; INT 28 people will use the DSKStack, we must turn OFF the
                                        ; generation of INT 28's.

                                    ; 15/12/2022
                                    ;mov     byte [ss:IDLEINT],0
                                    ;
                                        ;mov       sp,DSKSTACK
                                    ;test  byte [ss:CNTCFLAG],-1  ; 0FFh
                                        ;jz short DISPCALL

000003CA C606[5803]00               mov     byte [IDLEINT],0

000003CF BC[2009]                   MOV     SP,DSKSTACK
000003D2 F606[3703]FF               TEST    BYTE [CNTCFLAG],-1
000003D7 7405                       JZ      SHORT DISPCALL

                                        PUSH    AX
                                        ;invoke    DSKSTATCHK
000003DA E8265A                         CALL       DSKSTATCHK
000003DD 58                             POP     AX
                               DISPCALL:
                                    ; 01/05/2019 - Retro DOS v4.0
000003DE 2E8B9F[EC00]               mov     bx,[CS:BX+DISPATCH]

                                    ; 15/12/2022
000003E3 871E[EA05]                 xchg    bx,[SAVEBX]
000003E7 8E1E[EC05]                 MOV     DS,[SAVEDS]

                                    ; 04/11/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
                                    ; (ss: prefix was not needed here! ds=ss)
                                    ;xchg   bx,[ss:SAVEBX]
                                    ;mov    ds,[ss:SAVEDS]

000003EB 36FF16[EA05]               call    word [SS:SAVEBX] ; near call

                                    ; The EXEXA20OFF bit of DOS_FLAG will now be unconditionally cleared
                                    ; here. Please see under M003, M009 and M068 tags in dossym.inc
                                    ; for explanation. Also NOTE that a call to ExecReady (ax=4b05) will
                                    ; return to LeaveDos and hence will not clear this bit. This is
                                    ; because this bit is used to indicate to the next int 21 call that
                                    ; the previous int 21 was an exec.
                                    ;
                                    ; So do not add any code between the call above and the label
                                    ; LeaveDOS if it needs to be executed even for ax=4b05

                                    ;;and   byte [ss:DOS_FLAG],~EXECA20OFF
                                    ;and    byte [ss:DOS_FLAG],0FBh ; clear bit 2

                                    ; 01/01/2024 - Retro DOS v5.0 (PCDOS 7.1 IBMDOS.COM)
000003F0 368026[8600]DB             and     byte [ss:DOS_FLAG],0DBh ; clear bit 2 and bit 5

                               ; 04/11/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
                               ; DOSCODE:41F7h

                               ; 01/01/2024 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMDOS.COM)
                               ; DOSCODE:42D4h
```

```
                                ;entry LEAVE
                                ;;;_LEAVE:                          ; Exit from a system call
                                LeaveDOS: ; 18/07/2018
                                ;ASSUME     SS:NOTHING                      ; User routines may misbehave
000003F6 FA                         CLI

                                    ; 01/05/2019
                                    ;getdseg <DS>                   ; DS -> DosData, ASSUME DS:DosSeg
000003F7 2E8E1E[0700]               mov     ds,[cs:DosDSeg]
000003FC 803E[8500]00               cmp     byte [A20OFF_COUNT],0 ; M068: Q: is count 0
00000401 752A                       jne     short disa20            ; M068: N: dec count and turn a20 off

                                LeaveA20On:
00000403 FE0E[2103]                     DEC     BYTE [INDOS]

                                    ; 01/01/2024 (PCDOS 7.1 IBMDOS.COM)
00000407 FE0E[B812]                 dec     byte [INDOS_FLAG]       ; duplicated INDOS flag (what for ?)

                                        ; 04/11/2022
0000040B 8E16[8605]                 mov     ss,[USER_SS]
0000040F 8B26[8405]                 MOV      SP,[USER_SP]
                                    ;MOV    SS,[USER_SS]
00000413 89E5                       MOV      BP,SP
                                    ;MOV    [BP.user_AX],AL
                                        ; 04/11/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
                                    ;;mov   [bp+0],al ; MSDOS 5.0 MSDOS.SYS - DOSCODE:4212h
                                    ;MOV    [BP+user_env.user_AX],AL  ; user_env.user_AX = 0

                                    ; 15/12/2022
00000415 884600                     MOV     [BP],AL ; mov [bp+0],al

                                    ;MOV    AX,[NSP]
                                        ;MOV       [USER_SP],AX
                                        ;MOV       AX,[NSS]
                                        ;MOV       [USER_SS],AX
                                    ; 01/01/2024
00000418 C406[F005]                 les     ax,[NSS]
0000041C A3[8605]                   mov     [USER_SS],ax
0000041F 8C06[8405]                 mov     [USER_SP],es

00000423 58                         pop     AX
00000424 5B                         pop     BX
00000425 59                         pop     CX
00000426 5A                         pop     DX
00000427 5E                         pop     SI
00000428 5F                         pop     DI
00000429 5D                         pop     BP
0000042A 1F                         pop     DS
0000042B 07                         pop     ES

0000042C CF                             IRET

                                disa20:                                 ; M068 - Start
0000042D 8B1E[6300]                 mov     bx,[A20OFF_PSP]             ; bx = PSP for which a20 to be off'd
00000431 3B1E[3003]                 cmp     bx,[CurrentPDB]             ; Q: do the PSP's match
00000435 75CC                       jne     short LeaveA20On        ; N: don't clear bit and don't turn
                                                                    ;    a20 off
                                                                    ; Y: turn a20 off and dec a20off_count
00000437 FE0E[8500]                 dec     byte [A20OFF_COUNT]     ; M068 - End
                                                                    ; Start - M004
0000043B 1E                         push    ds                      ; segment of stub
0000043C BB[4210]                   mov     bx,disa20_iret          ; offset in stub
0000043F 53                         push    bx
00000440 CB                         retf                            ; go to stub
                                                                    ; End - M004
                                ;SYSTEM_CALL ENDP

                                ; DOSCODE:424Ch (MSDOS 6.21, MSDOS.SYS)
                                ; 04/11/2022
                                ; DOSCODE:423Fh (MSDOS 5.0, MSDOS.SYS)

                                ; ============================================================================
                                ;
                                ; Restore_World restores all registers ('cept SS:SP, CS:IP, flags) from
                                ; the stack prior to giving the user control
                                ;
                                ; ============================================================================

                                ; 01/05/2019 - Retro DOS v4.0

                                        ;procedure restore_world,NEAR
                                restore_world:
                                    ;getdseg <es>           ; es -> dosdata
00000441 2E8E06[0700]               mov     es,[cs:DosDSeg]

00000446 268F06[EE05]                   POP WORD [ES:RESTORE_TMP]

                                        POP     AX
                                        POP     BX
                                        POP     CX
                                        POP     DX
                                        POP     SI
                                        POP     DI
                                        POP     BP
                                        POP     DS

00000453 26FF26[EE05]                       jmp     word [ES:RESTORE_TMP]

                                ;restore_world     ENDP

                                ; 01/05/2019 - Retro DOS v4.0 (MSDOS 6.0, MSDISP.ASM, 1991)

                                ; DOSCODE:4263h (MSDOS 6.21, MSDOS.SYS)
                                ; 04/11/2022
                                ; DOSCODE:4256h (MSDOS 5.0, MSDOS.SYS)

                                ; ============================================================================
                                ;
                                ; Save_World saves complete registers on the stack
                                ;
                                ; ============================================================================

                                        ;procedure save_world,NEAR
                                save_world:
                                    ;getdseg <es>           ; es -> dosdata
00000458 2E8E06[0700]               mov     es,[cs:DosDSeg]

0000045D 268F06[EE05]                   POP WORD [ES:RESTORE_TMP]

                                    ; 12/05/2019

00000462 1E                         PUSH    DS
00000463 55                             PUSH    BP
```

```
4710 00000464 57                            PUSH    DI
4711 00000465 56                            PUSH    SI
4712 00000466 52                            PUSH    DX
4713 00000467 51                            PUSH    CX
4714 00000468 53                            PUSH    BX
4715 00000469 50                            PUSH    AX
4716
4717 0000046A 26FF36[EE05]                  push    word [ES:RESTORE_TMP]
4718
4719 0000046F 55                            push    BP
4720 00000470 89E5                          mov     BP,SP
4721 00000472 8E4614                        mov     ES,[BP+20]      ; es was pushed before call
4722 00000475 5D                            pop     BP
4723
4724 00000476 C3                            retn
4725
4726                                    ;save_world ENDP
4727
4728                                    ; 01/05/2019
4729
4730                                    ; DOSCODE:4282h (MSDOS 6.21, MSDOS.SYS)
4731                                    ; 04/11/2022
4732                                    ; DOSCODE:4275h (MSDOS 5.0, MSDOS.SYS)
4733
4734                                    ; ============================================================================
4735                                    ;
4736                                    ; Get_User_Stack returns the user's stack (and hence registers) in DS:SI
4737                                    ;
4738                                    ; ============================================================================
4739
4740                                            ;procedure get_user_stack,NEAR
4741                                    Get_User_Stack:
4742                                            ;getdseg <DS>                   ; DS -> DosData, ASSUME DS:DosSeg
4743 00000477 2E8E1E[0700]                  mov     ds,[cs:DosDSeg]
4744 0000047C C536[8405]                        lds si,[USER_SP]
4745 00000480 C3                            retn
4746
4747                                    ;get_user_stack  ENDP
4748
4749                                    ; 22/12/2022
4750                                    ; 04/11/2022 - Retro DOS v4.0 (Modified MSDOS 5.0, MSDOS.SYS)
4751                                    ; 01/01/2024 - Retro DOS v5.0 (Modified PCDOS 7.1, IBMDOS.COM)
4752                                    ;%if 0
4753
4754                                    ; ----------------------------------------------------------------------------
4755                                    ;
4756                                    ; Set_OEM_Handler -- Set OEM sys call address and handle OEM Calls
4757                                    ; Inputs:
4758                                    ;   User registers, User Stack, INTS disabled
4759                                    ;   If CALL F8, DS:DX is new handler address
4760                                    ; Function:
4761                                    ;   Process OEM INT 21 extensions
4762                                    ; Outputs:
4763                                    ;   Jumps to OEM_HANDLER if appropriate
4764                                    ;
4765                                    ; ----------------------------------------------------------------------------
4766
4767                                    ;IF NOT IBM
4768
4769                                    _$SET_OEM_HANDLER:
4770                                        ; 01/05/2019 - Retro DOS v4.0
4771
4772                                        ;(cmp   ah,SET OEM HANDLER  ; 0F8h)
4773                                        ;(jb    short NOTOOEM)
4774
4775 00000481 06                            push    es ; *
4776                                        ;getdseg <es>                   ; es -> dosdata
4777 00000482 2E8E06[0700]                  mov     es,[cs:DosDSeg]
4778
4779 00000487 750C                          jne     short check_trueversion_request ; check Retro DOS true version
4780                                                                                ; (message) request
4781                                        ; AH = 0F8h = SET OEM HANDLER
4782
4783 00000489 268916[1400]                  MOV      [es:OEM_HANDLER],DX   ; Set Handler
4784 0000048E 268C1E[1600]                  MOV      [es:OEM_HANDLER+2],DS
4785
4786 00000493 07                            pop     es ; *
4787
4788 00000494 CF                            IRET                            ; Quick return, Have altered no registers
4789
4790                                    check_trueversion_request:
4791                                        ; 18/07/2019 - Retro DOS v3.0
4792
4793                                        ; Retro DOS v2.0 - 20/04/2018
4794 00000495 83F8FF                        CMP     AX,0FFFFh
4795                                        ; 18/07/2018
4796 00000498 7520                          jne     short DO_OEM_FUNC ; 01/05/2019
4797
4798                                        ; 01/05/2019
4799 0000049A 07                            pop     es ; *
4800
4801 0000049B B40E                          mov     ah,0Eh
4802
4803                                        ; Retro DOS v4.0 feature only!
4804 0000049D 81FBA101                      cmp     bx,417  ; Signature to bypass
4805                                                        ; Retro DOS true version message
4806 000004A1 7414                          je      short true_version_iret
4807
4808 000004A3 56                            push    si
4809 000004A4 53                            push    bx
4810
4811 000004A5 BE[C200]                      mov     si,RETRODOSMSG
4812                                    wrdosmsg:
4813                                        ;movb   ah,0Eh
4814 000004A8 BB0700                        mov     bx,7
4815                                    wrdosmsg_nxt:
4816 000004AB 2EAC                          cs      lodsb
4817 000004AD 3C24                          cmp     al,'$'
4818 000004AF 7404                          je      short wrdosmsg_ok
4819 000004B1 CD10                          int     10h
4820 000004B3 EBF6                          jmp     short wrdosmsg_nxt
4821
4822                                    wrdosmsg_ok:
4823 000004B5 5B                            pop     bx
4824 000004B6 5E                            pop     si
4825
4826                                    true_version_iret:
4827                                        ; ah = 0Eh
4828                                        ;mov    al,40h ; Retro DOS v4.0
4829                                        ;
4830                                        ;mov    al,41h ; Retro DOS v4.1
4831                                        ; 30/12/2022
4832                                        ;mov    al,42h ; Retro DOS v4.2
4833                                        ; 01/01/2024
```

```
                                    mov     al,50h ; Retro DOS v5.0
000004B9 CF                         iret

                                    ; If above F8 try to jump to handler

                                DO_OEM_FUNC:
                                    ; 01/05/2019
000004BA 26833E[1400]FF             cmp     word [es:OEM_HANDLER],-1
000004C0 7504                       JNE     short OEM_JMP
000004C2 07                         pop     es ; *
000004C3 E903FE                     JMP     BADCALL                 ; Handler not initialized
                                OEM_JMP:
000004C6 06                         push    es
000004C7 1F                         pop     ds ; DOSDATA segment !
000004C8 07                         pop     es ; *

                                    ; 22/12/2022
000004C9 FB                         sti     ; (enable interrupts before jumping to private handler)

000004CA FF2E[1400]                 JMP     FAR [OEM_HANDLER]

                                ;       ENDIF

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

                                ;%endif

                                ;============================================================================
                                ; MCODE.ASM, MSDOS 6.0, 1991
                                ;============================================================================
                                ; 17/07/2018 - Retro DOS v3.0

                                ;   TITLE   MISC DOS ROUTINES - Int 25 and 26 handlers and other
                                ;   NAME    IBMCODE

                                ;BREAK <NullDev -- Driver for null device>

                                ; ROMDOS note:
                                ;   NUL device driver used to be here, but it was removed and placed in
                                ;   DOSDATA, because the entry points have to be in the segment as the
                                ;   header, which is also in DOSDATA.

                                ;BREAK <AbsDRD, AbsDWRT -- INT int_disk_read, int_disk_write handlers>

                                ;----------------------------------------------------------------------------
                                ; 04/11/2022 - Retro DOS v4.0 (Modified MSDOS 5.0, MSDOS.SYS)
                                ;----------------------------------------------------------------------------
                                ; DOSCODE:428Ch (MSDOS 6.21 MSDOS.SYS)
                                ; DOSCODE:427Fh (MSDOS 5.0 MSDOS.SYS)

                                ; 01/01/2024 - Retro DOS v5.0
                                ; DOSCODE:435Fh (PCDOS 7.1 IBMDOS.COM)

                                ;Public MSC001S,MSC001E
                                ;MSC001S label byte
                                    ;IF      IBM
                                ; Codes returned by BIOS
                                ERRIN:
000004CE 02                         DB      2                       ; NO RESPONSE
000004CF 06                         DB      6                       ; SEEK FAILURE
000004D0 0C                         DB      12                      ; GENERAL ERROR
000004D1 04                         DB      4                       ; BAD CRC
000004D2 08                         DB      8                       ; SECTOR NOT FOUND
000004D3 00                         DB      0                       ; WRITE ATTEMPT ON WRITE-PROTECT DISK
                                ERROUT:
                                ; DISK ERRORS RETURNED FROM INT 25 and 26
000004D4 80                         DB      80H                     ; NO RESPONSE
000004D5 40                         DB      40H                     ; Seek failure
000004D6 02                         DB      2                       ; Address Mark not found
000004D7 10                         DB      10H                     ; BAD CRC
000004D8 04                         DB      4                       ; SECTOR NOT FOUND
000004D9 03                         DB      3                       ; WRITE ATTEMPT TO WRITE-PROTECT DISK

                                NUMERR      EQU     $-ERROUT
                                    ;ENDIF
                                ;MSC001E label byte
                                ;----------------------------------------------------------------------------

                                ;============================================================================
                                ; MSCODE.ASM - MSDOS 6.0 - 1991
                                ;============================================================================
                                ; 18/07/2018 - Retro DOS v3.0
                                ; 15/05/2019 - Retro DOS v4.0

                                ; 02/01/2024 - Retro DOS v5.0
                                ; PCDOS 7.1 IBMDOS.COM - DOSCODE:436Bh

                                ;BREAK <AbsDRD, AbsDWRT -- INT int_disk_read, int_disk_write handlers>>

                                ;   AbsSetup - setup for abs disk functions
                                ;----------------------------------------------------------------------------

                                AbsSetup:
                                    ; 02/01/2024 (PCDOS 7.1, Retro DOS 5.0)
                                    ;;;
                                    ;
000004DA 1E                         push    ds ; *
000004DB 16                         push    ss
000004DC 1F                         pop     ds
                                    ;
000004DD 8826[DB0A]                 mov     [absdrw_extd],ah
                                    ;mov    [ss:absdrw_extd],ah    ; Extended ABS Disk Read/Write flag
                                                                   ; (AH=1 for INT 21h ax=7305h function)
000004E1 08E4                       or      ah,ah
000004E3 7508                       jnz     short AbsSetup1             ; INT 21h AX=7305h

                                    ; INT 25h
000004E5 FE06[B812]                 inc     byte [INDOS_FLAG]
                                    ;inc    byte [ss:INDOS_FLAG]  ; Windows DOSBOX's INDOS flag ?
                                    ;;;
000004E9 FE06[2103]                 inc     byte [INDOS]
                                    ;INC    byte [SS:INDOS]             ; SS override
                                AbsSetup1:
000004ED FB                         STI
000004EE FC                         CLD
                                    ; 02/01/2024
                                    ;PUSH   DS
                                    ;push   ss
                                    ;pop    ds
000004EF E83C01                     CALL    GETBP
                                    ; 02/01/2024
000004F2 1F                         pop     ds ; *
000004F3 7229                       JC      short errdriv           ; PM. error drive ;AN000;
```

```
                                    ; 02/01/2024
                                    ;;mov   word [es:bp+1Fh]
                                    ;MOV    WORD [ES:BP+DPB.FREE_CNT],-1 ; do not trust user at all.
                                ;errdriv:
                                    ;POP    DS
                                    ;jnc    short AbsSetup2
                                ;AbsSetup_retn:
                                    ;retn

                                AbsSetup2:
                                    ; 15/05/2019 - Retro DOS v4.0
                                    ; MSDOS 6.0
                                                                    ; SS override
000004F5 36C706[0706]0000           MOV     word [SS:HIGH_SECTOR],0 ;>32mb       from API              ;AN000;
000004FC E8BD04                     CALL    RW32_CONVERT            ;>32mb convert 32bit format to 16bit ;AN000;
                                    ;jc     short AbsSetup_retn
                                    ;call   SET_RQ_SC_PARMS         ;LB. set up SC parms          ;AN000;
                                    ; 02/01/2024 - Retro DOS v5.0
000004FF 721D                       jc      short errdriv
                                    ;call   null_sub ; retn         ; PCDOS 7.1 IBMDOS.COM

                                    ; MSDOS 3.3 (& MSDOS 6.0)
00000501 1E                         PUSH    DS
00000502 56                         PUSH    SI
00000503 50                         PUSH    AX

00000504 16                         push    ss
00000505 1F                         pop     ds

00000506 BE[BE03]                   MOV     SI,OPENBUF
00000509 8804                       MOV     [SI],AL
0000050B 800441                     ADD     BYTE [SI],"A"
0000050E C744013A00                 MOV     WORD [SI+1],003AH ; ":",0
00000513 B80003                     MOV     AX,0300H
00000516 F8                         CLC
00000517 CD2A                       INT     int_IBM ; int 2Ah     ; Will set carry if shared

                                                    ; 04/11/2022
                                                    ; (INT 2Ah - AX = 0300h)
                                                    ; Microsoft Networks - CHECK DIRECT I/O
                                                    ; DS:SI -> ASCIIZ disk device name (may be full path or
                                                    ;    only drive specifier--must include the colon)
                                                    ; Return: CF clear if absolute disk access allowed

00000519 58                             POP     AX
0000051A 5E                             POP     SI
0000051B 1F                             POP     DS
0000051C 730E                           jnc     short AbsSetup_retn

                                        ; 02/01/2024
                                    errdriv:
                                        ;mov    word [ss:EXTERR],32h
0000051E 36C706[2403]3200               MOV     word [ss:EXTERR],error_not_supported
                                        ; 02/01/2024 - Retro DOS v5.0
00000525 36C706[BF0D]0702               mov     word [ss:AbsDskErr],207h ; PCDOS 7.1 IBMDOS.COM

                                        ; 02/01/2024
                                    AbsSetup_retn:
0000052C C3                             retn

                                    ;--------------------------------------------------------------------------
                                    ;
                                    ; Procedure Name : ABSDRD
                                    ;
                                    ; Interrupt 25 handler. Performs absolute disk read.
                                    ; Inputs:   AL - 0-based drive number
                                    ;           DS:BX point to destination buffer
                                    ;           CX number of logical sectors to read
                                    ;           DX starting logical sector number (0-based)
                                    ; Outputs:  Original flags still on stack
                                    ;           Carry set
                                    ;               AH error from BIOS
                                    ;               AL same as low byte of DI from INT 24
                                    ;
                                    ;--------------------------------------------------------------------------
                                            ;procedure   ABSDRD,FAR
                                    ABSDRD:
                                        ; 15/05/2019 - Retro DOS v4.0
                                        ; MSDOS 6.21 (DOSCODE:42E5h)
                                        ; 04/11/2022
                                        ; MSDOS 5.0 (DOSCODE:42D8h)

                                        ; 02/01/2024 - Retro DOS v5.0
                                        ; PCDOS 7.1 IBMDOS.COM - DOSCODE:43C4h

                                        ;;;
0000052D 30E4                           xor     ah,ah  ; ah=0         ; Interrupt 25h handler (ah=0)
0000052F 31F6                           xor     si,si  ; si=0         ; clear read/write mode flags
                                                                      ; (used with INT 21h ax=7305h)
                                    FAT32_ABSDRD:       ; ah=1 si=0 cf=0 (jump from '_$FAT32EXT')
                                        ; 25/06/2024
                                        ;cli ; *
                                        ;clc ; not necessary
                                                                    ; INT 21h
                                                                    ;       AX = 7305h
                                                                    ; FAT32 - EXTENDED ABSOLUTE DISK READ/WRITE
                                                                    ;       CX = FFFFh
                                                                    ;       DL = drive number (01h=A:, etc.)
                                                                    ;       SI = read/write mode flags
                                                                    ;       DS:BX -> disk I/O packet
                                                                    ;
                                                                    ; Extended Absolute Disk Read/Write mode flags:
                                                                    ;
                                                                    ;       Bit(s)  Description
                                                                    ;       0       direction (0=read, 1=write)
                                                                    ;       12-1    reserved (0)
                                                                    ;       14-13   write type (should be 00 on reads).
                                                                    ;               00 unknown data.
                                                                    ;               01 FAT data.
                                                                    ;               10 directory data.
                                                                    ;               11 file data
                                                                    ;       15      reserved (0)
                                                                    ;
                                                                    ; Format of disk read/write packet:
                                                                    ;
                                                                    ;       Offset Size     Description
                                                                    ;       00h    DWORD    sector number
                                                                    ;       04h    WORD     number of sectors to r/w
                                                                    ;       06h    DWORD    transfer address
                                                                    ;
                                                                    ; ref: Ralf Brown's Interrupt List
                                        ;;;
                                    absdrd_1:
                                        ; MSDOS 6.0
```

```
                                    ; 25/06/2024 - Retro DOS v5.0
00000531 FA                         CLI ; *
                                
                                ;   set up ds to point to DOSDATA
                                
00000532 50                         push    ax                  ; preserve AX value
00000533 8CD8                       mov     ax,ds               ; store DS value in AX
                                    ;getdseg <ds>
00000535 2E8E1E[0700]               mov     ds,[cs:DosDSeg]
0000053A A3[630D]                   mov     [TEMPSEG],ax        ; store DS value in TEMPSEG
0000053D 58                         pop     ax                  ; restore AX value
                                
                                    ; M072:
                                    ; We shall save es on the user stack here. We need to use ES in
                                    ; order to access the DOSDATA variables AbsRdWr_SS/SP at exit
                                    ; time in order to restore the user stack.
                                
0000053E 06                         push    es  ; ****          ; M072
                                
                                    ; 25/06/2024
                                    ; 02/01/2024 - Retrodos v5.0 (PCDOS 7.1 IBMDOS.COM)
                                    ;;;
0000053F 7305                       jnc     short absdrd_2      ; (not jumped from ABSDWRT) absolute disk read
                                
                                    ;(jumped from ABSDRWT)
00000541 08E4                       or      ah,ah
00000543 F9                         stc                         ; absolute disk write
00000544 EB02                       jmp     short absdrd_3
                                absdrd_2:
00000546 08E4                       or      ah,ah
                                absdrd_3:
00000548 7510                       jnz     short absdrd_4      ; EXTENDED ABSOLUTE DISK READ/WRITE
                                    ;;;
                                
0000054A 8C16[1B06]                 MOV     [AbsRdWr_SS],SS             ; M013
0000054E 8926[1D06]                 MOV     [AbsRdWr_SP],SP             ; M013
                                
                                ;   set up ss to point to DOSDATA
                                ;
                                ; NOTE! Due to an obscure bug in the 80286, you cannot use the ROMDOS
                                ; version of the getdseg macro with the SS register! An interrupt will
                                ; sneak through.
                                
                                ;ifndef ROMDOS
                                    ;getdseg <ss>               ; cli in entry of routine
                                
00000552 2E8E16[0700]               mov     ss,[cs:DosDSeg]
                                
                                ;else
                                ;   mov     ds, cs:[BioDataSeg]
                                ;   assume  ds:bdata
                                ;
                                ;   mov     ss, ds:[DosDataSg]
                                ;   assume  ss:DOSDATA
                                ;
                                ;endif ; ROMDOS
                                                                    ; 02/01/2024
00000557 BC[2009]                   MOV     SP,DSKSTACK             ;"@#IBM:12.01.2003.build_1.32#@ IBMDOS.COM(USA)"
                                                                    ; (PCDOS 7.1 IBMDOS.COM)
                                absdrd_4:
0000055A 8E1E[630D]                 mov     ds,[TEMPSEG]            ; restore DS value
                                
0000055E 06                         push    es ; *** (MSDOS 6.21)
0000055F E8F6FE                     call    save_world              ; save all regs
                                
00000562 06                         PUSH    ES ; **
                                
                                    ; 02/01/2024 - Retrodos v5.0
                                    ;;;
00000563 7303                       jnc     short absdrd_5          ; absolute disk read
00000565 E9A400                     jmp     absdrwt_3               ; (jumping back to) absolute disk write
                                absdrd_5:
                                    ;;;
                                
00000568 E86FFF                     CALL    AbsSetup
0000056B 723D                       JC      short ILEAVE
                                
                                    ; Here is a gross temporary fix to get around a serious design flaw in
                                    ;  the secondary cache. The secondary cache does not check for media
                                    ;  changed (it should). Hence, you can change disks, do an absolute
                                    ;  read, and get data from the previous disk. To get around this,
                                    ;  we just won't use the secondary cache for absolute disk reads.
                                    ;                                                       -mw 8/5/88
                                
                                    ;EnterCrit critDisk
0000056D E87213                     call    ECritDisk
00000570 36C606[7511]FF             MOV     byte [ss:CurSC_DRIVE],-1 ; invalidate SC  ;AN000;
                                    ;LeaveCrit critDisk
00000576 E89613                     call    LCritDisk
                                
                                        ;invoke    DSKREAD
00000579 E8C73A                     CALL    DSKREAD
0000057C 7513                           jnz short ERR_LEAVE             ;Jump if read unsuccessful.
                                
0000057E 89F9                           mov cx,di
00000580 368C1E[0E06]                   mov [ss:TEMP_VAR2],ds
00000585 36891E[0C06]                   mov [ss:TEMP_VAR],bx
                                
                                ;       CX = # of contiguous sectors read. (These constitute a block of
                                ;            sectors, also termed an "Extent".)
                                ;       [HIGH_SECTOR]:DX = physical sector # of first sector in extent.
                                ;       [TEMP_VAR2]:[TEMP_VAR] = Transfer address (destination data address).
                                ;       ES:BP -> Drive Parameter Block (DPB).
                                ;
                                ;   The Buffer Queue must now be scanned: the contents of any dirty
                                ;   buffers must be "read" into the transfer memory block, so that the
                                ;       transfer memory reflects the most recent data.
                                
                                    ;invoke DskRdBufScan            ;This trashes DS, but don't care.
0000058A E8483D                         call    DskRdBufScan
0000058D EB1B                       jmp     short ILEAVE
                                
                                TLEAVE:
0000058F 7419                       JZ      short ILEAVE
                                
                                ERR_LEAVE:                          ; M039
                                    ; 15/07/2018 - Retro DOS v3.0
                                        ;IF IBM
                                ; Translate the error code to ancient 1.1 codes
                                        PUSH    ES ; *
                                        PUSH    CS
                                        POP ES
                                        XOR AH,AH                   ; Nul error code
                                    ;mov    cx,6
```

```
5206 00000596 B90600                        MOV CX,NUMERR             ; Number of possible error conditions
5207 00000599 BF[CE04]                      MOV DI,ERRIN              ; Point to error conditions
5208 0000059C F2AE                          REPNE       SCASB
5209 0000059E 7504                          JNZ SHORT LEAVECODE               ; Not found
5210                                    ;mov    ah,[ES:DI+5]
5211 000005A0 268A6505                      MOV AH,[ES:DI+NUMERR-1]   ; Get translation
5212                                LEAVECODE:
5213 000005A4 07                            POP ES ; *
5214                                    ; 15/05/2019 - Retro DOS v4.0
5215 000005A5 36A3[BF0D]                mov     [ss:AbsDskErr],ax
5216                                        ;ENDIF
5217
5218 000005A9 F9                            STC
5219                                ILEAVE:
5220                                    ; 15/05/2019
5221 000005AA 07                            POP ES ; **
5222 000005AB E893FE                    call    restore_world
5223 000005AE 07                            pop es ; *** (MSDOS 6.21)
5224
5225                                    ; 02/01/2024 - Retro DOS v5.0 (PCDOS 7.1 IBMDOS.COM)
5226                                    ;;;
5227 000005AF 9C                        pushf
5228 000005B0 36803E[DB0A]00            cmp     byte [ss:absdrw_extd],0
5229                                                ; FAT32- EXTENDED ABSOLUTE DISK READ/WRITE flag
5230 000005B6 751F                      jnz     short ILEAVE_EXTD     ; INT 21h AX=7305h
5231                                    ; INT 25h
5232 000005B8 9D                        popf
5233                                    ;;;
5234
5235 000005B9 FA                        CLI
5236 000005BA 36A1[BF0D]                mov     ax,[ss:AbsDskErr]     ; restore error
5237 000005BE 36FE0E[2103]              DEC     BYTE [SS:INDOS]
5238                                    ;
5239                                    ; 02/01/2024 (PCDOS 7.1 IBMDOS.COM)
5240 000005C3 36FE0E[B812]              dec     byte [ss:INDOS_FLAG]  ; Windows DOSBOX's INDOS flag ?
5241                                    ;
5242 000005C8 16                            push       ss                     ; M072 - Start
5243 000005C9 07                            pop     es                        ; es - dosdata
5244 000005CA 268E16[1B06]                      mov ss,[es:AbsRdWr_SS]    ; M013
5245 000005CF 268B26[1D06]              mov     sp,[es:AbsRdWr_SP]    ; M013
5246 000005D4 07                        pop     es  ; ****                ; Note es was saved on user
5247                                                                      ; stack at entry
5248                                                                      ; M072 - End
5249 000005D5 FB                            STI
5250 000005D6 CB                        RETF   ; ! FAR return !
5251
5252                                    ; 02/01/2024 - Retro DOS v5.0
5253                                    ; (PCDOS 7.1 IBMDOS.COM)
5254                                    ;;;
5255                                ILEAVE_EXTD:       ; return from INT 21h AX=7305h
5256 000005D7 9D                        popf
5257 000005D8 36A1[BF0D]                mov     ax,[ss:AbsDskErr]     ; restore error
5258 000005DC 07                        pop     es ; ****
5259 000005DD FB                        sti
5260 000005DE C3                        retn
5261                                    ;;;
5262
5263                                ;ABSDRD     ENDP
5264
5265                                ;--------------------------------------------------------------------------
5266                                ;
5267                                ; Procedure Name : ABSDWRT
5268                                ;
5269                                ; Interrupt 26 handler. Performs absolute disk write.
5270                                ; Inputs:   AL - 0-based drive number
5271                                ;           DS:BX point to source buffer
5272                                ;           CX number of logical sectors to write
5273                                ;           DX starting logical sector number (0-based)
5274                                ; Outputs:  Original flags still on stack
5275                                ;           Carry set
5276                                ;               AH error from BIOS
5277                                ;               AL same as low byte of DI from INT 24
5278                                ;
5279                                ;--------------------------------------------------------------------------
5280                                        ;procedure   ABSDWRT,FAR
5281                                ABSDWRT:
5282                                    ; 15/05/2019 - Retro DOS v4.0
5283                                    ; MSDOS 6.21 (DOSCODE:436Ch)
5284                                    ; 04/11/2022
5285                                    ; MSDOS 5.0 (DOSCODE:435Fh)
5286
5287                                    ; 03/01/2024 - Retro DOS v5.0
5288                                    ; PCDOS 7.1 IBMDOS.COM - DOSCODE:4477h
5289
5290                                    ;;;
5291 000005DF 30E4                      xor     ah,ah   ; ah=0            ; Interrupt 26h handler (ah=0)
5292 000005E1 BE0100                    mov     si,1                      ; set direction flag/bit to write
5293                                                                      ; (used with INT 21h ax=7305h)
5294                                FAT32_ABSDWRT:     ; ah=1 si=1 cf=0 (jump from '_$FAT32EXT')
5295
5296                                                                      ; INT 21h
5297                                                                      ;       AX = 7305h
5298                                                                      ; FAT32 - EXTENDED ABSOLUTE DISK READ/WRITE
5299                                                                      ;       CX = FFFFh
5300                                                                      ;       DL = drive number (01h=A:, etc.)
5301                                                                      ;       SI = read/write mode flags
5302                                                                      ;       DS:BX -> disk I/O packet
5303                                                                      ;
5304                                                                      ; Extended Absolute Disk Read/Write mode flags:
5305                                                                      ;
5306                                                                      ;       Bit(s)  Description
5307                                                                      ;       0       direction (0=read, 1=write)
5308                                                                      ;       12-1    reserved (0)
5309                                                                      ;       14-13   write type (should be 00 on reads).
5310                                                                      ;               00 unknown data.
5311                                                                      ;               01 FAT data.
5312                                                                      ;               10 directory data.
5313                                                                      ;               11 file data
5314                                                                      ;       15      reserved (0)
5315                                                                      ;
5316                                                                      ; Format of disk read/write packet:
5317                                                                      ;
5318                                                                      ;       Offset Size     Description
5319                                                                      ;       00h    DWORD    sector number
5320                                                                      ;       04h    WORD     number of sectors to r/w
5321                                                                      ;       06h    DWORD    transfer address
5322                                                                      ;
5323                                                                      ; ref: Ralf Brown's Interrupt List
5324 000005E4 3C02                          cmp     al,2
5325 000005E6 7220                          jb      short absdrwt_2       ; floppy disk
5326                                        ; hard disk
5327 000005E8 53                            push    bx
5328 000005E9 1E                            push    ds
5329 000005EA 2E8E1E[0700]                  mov     ds,[cs:DosDSeg]
```

```
                                    xor     bh,bh
000005F1 88C3                       mov     bl,al                   ;
                                                                    ; NOTE: PCDOS 7.1 kernel does not set
                                                                    ; DOS_FLAG bit 6 or drive_flags bit 7
                                                                    ; (It appears that these bits are set
                                                                    ; by Windows or a system utility or
                                                                    ; driver that knows the addresses of
                                                                    ; these FLAGS in the DOSDATA segment.)
                                                                    ; Erdogan Tan - 03/01/2024
                                                                    ;
000005F3 F687[0813]80               test    byte [drive_flags+bx],80h
                                                                    ; test bit 7 (locked bit) ; 29/01/2024
000005F8 7505                       jnz     short absdwrt_1             ; locked (logical drive) -allowed to abs write-
                                                                    ; NOTE: lock/unlock are MSDOS/PCDOS 7 extd functions
000005FA F606[8600]40               test    byte [DOS_FLAG],40h     ; test bit 6 (large disk support -windows- bit?)
                                                                    ; (Windows OS running bit ?) ; 23/02/2024
                                                                    ; NOTE: Retro DOS v5 kernel must set this bit.
                                    absdwrt_1:
000005FF 1F                         pop     ds
00000600 5B                         pop     bx
00000601 7505                       jnz     short absdrwt_2         ; allowed
00000603 F9                         stc
00000604 E817FF                     call    errdriv         ; error
00000607 CB                         retf

                                    ;absdrwt_2:
                                    ;;;

                                    ; 03/01/2024
                                    %if 0
                                    CLI

                                    ;   set up ds to point to DOSDATA

                                    push    ax
                                    mov     ax,ds
                                    ;getdseg <ds>
                                    mov     ds,[cs:DosDSeg]
                                    mov     [TEMPSEG],ax
                                    pop     ax

                                    ; M072:
                                    ; We shall save es on the user stack here. We need to use ES in
                                    ; order to access the DOSDATA variables AbsRdWr_SS/SP at exit
                                    ; time in order to restore the user stack.

                                    push    es ; ****               ; M072

                                    MOV     [AbsRdWr_SS],SS                 ; M013
                                    MOV     [AbsRdWr_SP],SP                 ; M013

                                    ; set up ss to point to DOSDATA
                                    ;
                                    ; NOTE! Due to an obscure bug in the 80286, you cannot use the
                                    ; ROMDOS version of the getdseg macro with the SS register!
                                    ; An interrupt will sneak through.

                                    ;ifndef ROMDOS
                                    ;getdseg <ss>                   ; cli in entry of routine
                                    mov      ss,[cs:DosDSeg]
                                    ;else
                                    ;   mov     ds, cs:[BioDataSeg]
                                    ;   assume  ds:bdata
                                    ;
                                    ;   mov     ss, ds:[DosDataSg]
                                    ;   assume  ss:DOSDATA
                                    ;
                                    ;endif ; ROMDOS

                                    MOV     SP,DSKSTACK
                                            ; we are now switched to DOS's disk stack

                                    mov     ds,[TEMPSEG]            ; restore user's ds

                                    push    es ; *** (MSDOS 6.21)

                                    call    save_world              ; save all regs

                                    PUSH    ES ; **
                                    %endif
                                    ; 03/01/2024 - Retro DOS v5.0 (PCDOS 7.1 IBMDOS.COM)
                                    ;;;
                                    absdrwt_2:
                                    ; 25/06/2024
                                    ;cli
00000608 F9                         stc                     ; writable disk
                                                            ; ('jumped from ABSDWRT' sign for common r/w code)
00000609 E925FF                     jmp     absdrd_1        ; jump to ABSDRD (common r/w) code
                                    ;;;

                                    absdrwt_3:
0000060C E8CBFE                     CALL    AbsSetup
0000060F 7299                       JC      short ILEAVE

                                    ; 03/01/2024 - Retro DOS v5.0 (PCDOS 7.1 IBMDOS.COM)
00000611 E89E5F                     call    chk_set_first_access

                                    ;EnterCrit critDisk
00000614 E8CB12                     call    ECritDisk
00000617 36C606[7511]FF             MOV     byte [ss:CurSC_DRIVE],-1 ; invalidate SC ;AN000;
0000061D E8DB03                     CALL    Fastxxx_Purge           ; purge fatopen ;AN000;
                                    ;LeaveCrit critDisk
00000620 E8EC12                     call    LCritDisk

                                    ;M039
                                    ;       DS:BX = transfer address (source data address).
                                    ;       CX = # of contiguous sectors to write. (These constitute a block of
                                    ;        sectors, also termed an "Extent".)
                                    ;       [HIGH_SECTOR]:DX = physical sector # of first sector in extent.
                                    ;       ES:BP -> Drive Parameter Block (DPB).
                                    ;       [CURSC_DRIVE] = -1 (invalid drive).
                                    ;
                                    ;       Free any buffered sectors which are in Extent; they are being over-
                                    ;       written. Note that all the above registers are preserved for
                                    ;       DSKWRITE.

00000623 1E                             push       ds
                                    ;invoke DskWrtBufPurge          ; This trashes DS.
00000624 E89740                     call    DskWrtBufPurge
00000627 1F                             pop ds
                                    ;M039
                                    ;invoke DSKWRITE
00000628 E83C3A                     call    DSKWRITE
0000062B E961FF                     JMP     TLEAVE
```

```
                                ;ABSDWRT ENDP
                                ;--------------------------------------------------------------------------
                                ;
                                ; Procedure Name : GETBP
                                ;
                                ; Inputs:
                                ;   AL = Logical unit number (A = 0)
                                ; Function:
                                ;   Find Drive Parameter Block
                                ; Outputs:
                                ;   ES:BP points to DPB
                                ;   [THISDPB] = ES:BP
                                ;   Carry set if unit number bad or unit is a NET device.
                                ;           Later case sets extended error error_I24_not_supported
                                ; No other registers altered
                                ;
                                ;--------------------------------------------------------------------------

                                    ; 07/12/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
                                    ; 04/01/2024 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMDOS.COM)
                                    ; PCDOS 76.1 IBMDOS.COM - DOSCODE:44C7h
                                GETBP:
                                    ; 15/05/2019 - Retro DOS v4.0
                                    ; 11/07/2018 - Retro DOS v3.0
0000062E 50                         PUSH    AX
0000062F 0401                       ADD     AL,1            ; No increment; need carry flag
00000631 7210                       JC      SHORT SKIPGET
00000633 E85775                     CALL    GETTHISDRV
                                    ; MSDOS 6.0
00000636 730B                       JNC     SHORT SKIPGET         ;PM. good drive              ;AN000;

                                    ; 23/03/2024 - Retro DOS v5.0
                                    ;XOR    AH,AH                 ;DCR. ax= error code  ;AN000;
                                    ;CMP    AX,error_not_DOS_disk ;DCR. is unknown media ? ;AN000;
                                    ;JZ     SHORT SKIPGET         ;DCR. yes, let it go  ;AN000;
                                    ;STC                          ;DCR.                 ;AN000;
00000638 B400                       mov     ah,0

0000063A A3[2403]                   MOV     [EXTERR],AX    ;PM. invalid drive or Non DOS drive ;AN000;
0000063D C706[BF0D]0102             MOV     WORD [AbsDskErr],201h
                                SKIPGET:
00000643 58                         POP     AX
00000644 7212                       JC      SHORT GETBP_RETN ; 15/12/2022
                                    ; 07/12/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
                                    ;jnc    short getbp_t
                                    ;retn
                                getbp_t:
00000646 C42E[A205]                 LES     BP,[THISCDS]
                                    ; 15/12/2022
0000064A 26F6464480                 test    byte [es:bp+curdir.flags+1],curdir_isnet>>8
                                    ; 07/12/2022
                                    ;TEST   WORD [ES:BP+43H],8000H
                                    ;TEST   WORD [ES:BP+curdir.flags],curdir_isnet ; Clears carry
0000064F 7408                       JZ      SHORT GETBP_CDS
                                GETBP_err:  ; 04/01/2024
00000651 C706[2403]3200             MOV     WORD [EXTERR],error_not_supported  ; 32h
00000657 F9                         STC
                                GETBP_RETN:
00000658 C3                         RETN

                                GETBP_CDS:
                                    ;LES    BP,[ES:BP+45H]
00000659 26C46E45                   LES     BP,[ES:BP+curdir.devptr]
                                    ; 04/01/2024 (PCDOS 7.1 IBMDOS.COM)
                                    ;;;
0000065D 50                         push    ax
                                    ; 25/06/2024
0000065E 8CC0                       mov     ax,es
00000660 09E8                       or      ax,bp
00000662 58                         pop     ax
00000663 74EC                       jz      short GETBP_err ; zero address, error
                                    ;;;
                                ;--------------------------------------------------------------------------
                                GOTDPB:
                                    ; Load THISDPB from ES:BP
00000665 892E[8A05]                 MOV     [THISDPB],BP
00000669 8C06[8C05]                 MOV     [THISDPB+2],ES
0000066D C3                         RETN

                                ;BREAK <SYS_RET_OK SYS_RET_ERR CAL_LK ETAB_LK set system call returns>

                                ;--------------------------------------------------------------------------
                                ;
                                ; Procedure Name : SYS_RETURN
                                ;
                                ; These are the general system call exit mechanisms. All internal system
                                ; calls will transfer (jump) to one of these at the end. Their sole purpose
                                ; is to set the user's flags and set his AX register for return.
                                ;
                                ;--------------------------------------------------------------------------

                                        ;procedure   SYS_RETURN,NEAR
                                SYS_RETURN:
                                        ;entry     SYS_RET_OK
                                SYS_RET_OK:
0000066E E806FE                     call    Get_User_Stack
                                            ; turn off user's carry flag
                                SYS_RET_OK_clc: ; 25/06/2019
                                            ;;and      word [SI+16h],0FFFEh
                                    ;and    word [SI+user_env.user_F],~f_Carry
                                            ; 25/06/2019
00000671 806416FE                   and     byte [SI+user_env.user_F],~f_Carry ; 0FEh
                                    ; 04/01/2024
                                    ;JMP    SHORT DO_RET
                                DO_RET:
                                            ;MOV       [SI+user_env.user_AX],AX ; Really only sets AH
00000675 8904                       MOV     [SI],AX
00000677 C3                         RETN

                                            ;entry   SYS_RET_ERR
                                SYS_RET_ERR:
00000678 30E4                       XOR     AH,AH           ; hack to allow for smaller error rets
0000067A E86B00                     call    ETAB_LK         ; Make sure code is OK, EXTERR gets set
0000067D E81900                     CALL    ErrorMap

                                    ;entry  From_GetSet
                                From_GetSet:
00000680 E8F4FD                         call    Get_User_Stack
                                             ; signal carry to user
                                    ;;or    word [SI+16h],1
                                    ;OR     word [SI+user_env.user_F],f_Carry
                                    ; 25/06/2019
```

```
00000683 804C1601                   or      byte [SI+user_env.user_F],f_Carry
00000687 F9                         STC                     ; also, signal internal error
                                    ; 04/01/2024
00000688 EBEB                       jmp     short DO_RET
                                ;DO_RET:
                                    ;;MOV   [SI+user_env.user_AX],AX ; Really only sets AH
                                    ;MOV    [SI],AX
                                    ;RETN

                                    ;entry  FCB_RET_OK
                                FCB_RET_OK:
                                    ;entry  NO_OP           ; obsolete system calls dispatch to here
                                NO_OP:
0000068A 30C0                       XOR     AL,AL
0000068C C3                         retn

                                    ;entry  FCB_RET_ERR
                                FCB_RET_ERR:
0000068D 30E4                       XOR     AH,AH
0000068F 36A3[2403]                 mov     [ss:EXTERR],AX
00000693 E80300                     CALL    ErrorMap
00000696 B0FF                       MOV     AL,-1
00000698 C3                         retn

                                    ;entry  ErrorMap
                                ErrorMap:
00000699 56                         PUSH    SI
                                                            ; ERR_TABLE_21 is now in DOSDATA
0000069A BE[DB0D]                   MOV     SI,ERR_TABLE_21
                                                            ; SS override for FAILERR and EXTERR
0000069D 36803E[4A03]00             CMP     byte [SS:FAILERR],0 ; Check for SPECIAL case.
000006A3 7407                       JZ      short EXTENDED_NORMAL ; All is OK.
                                                             ; Ooops, this is the REAL reason
                                    ;mov    word [SS:EXTERR],53h
000006A5 36C706[2403]5300           MOV     word [SS:EXTERR],error_FAIL_I24
                                EXTENDED_NORMAL:
000006AC E80200                     call    CAL_LK          ; Set CLASS,ACTION,LOCUS for EXTERR
000006AF 5E                         POP     SI
000006B0 C3                         retn

                                    ;EndProc SYS_RETURN

                                ;--------------------------------------------------------------------------
                                ;
                                ; Procedure Name : CAL_LK
                                ;
                                ; Inputs:
                                ;   SI is OFFSET in DOSDATA of CLASS,ACTION,LOCUS Table to use
                                ;           (DS NEED not be DOSDATA)
                                ;   [EXTERR] is set with error
                                ; Function:
                                ;   Look up and set CLASS ACTION and LOCUS values for GetExtendedError
                                ; Outputs:
                                ;   [EXTERR_CLASS] set
                                ;   [EXTERR_ACTION] set
                                ;   [EXTERR_LOCUS] set  (EXCEPT on certain errors as determined by table)
                                ; Destroys SI, FLAGS
                                ;
                                ;--------------------------------------------------------------------------

                                    ;procedure CAL_LK,NEAR
                                CAL_LK:
000006B1 1E                         PUSH    DS
000006B2 50                         PUSH    AX
000006B3 53                         PUSH    BX

                                ;M048       Context DS              ; DS:SI -> Table
                                ;
                                ; Since this function can be called thru int 2f we shall not assume that SS
                                ; is DOSDATA

                                    ;getdseg  <ds> ; M048: DS:SI -> Table
                                    ; 15/05/2019 - Retro DOS v4.0
000006B4 2E8E1E[0700]               mov     ds,[cs:DosDSeg]

                                    ; 18/07/2018
                                    ;push   ss
                                    ;pop    ds

000006B9 8B1E[2403]                 MOV     BX,[EXTERR]     ; Get error in BL
                                TABLK1:
000006BD AC                         LODSB

000006BE 3CFF                       CMP     AL,0FFH
000006C0 7409                       JZ      short GOT_VALS ; End of table
000006C2 38D8                       CMP     AL,BL
000006C4 7405                       JZ      short GOT_VALS ; Got entry
000006C6 83C603                     ADD     SI,3            ; Next table entry
                                    ; 15/08/2018
000006C9 EBF2                       JMP     short TABLK1

                                GOT_VALS:
000006CB AD                         LODSW                   ; AL is CLASS, AH is ACTION

000006CC 80FCFF                     CMP     AH,0FFH
000006CF 7404                       JZ      short NO_SET_ACT
000006D1 8826[2603]                 MOV     [EXTERR_ACTION],AH ; Set ACTION
                                NO_SET_ACT:
000006D5 3CFF                       CMP     AL,0FFH
000006D7 7403                       JZ      short NO_SET_CLS
000006D9 A2[2703]                   MOV     [EXTERR_CLASS],AL ; Set CLASS
                                NO_SET_CLS:
000006DC AC                         LODSB                   ; Get LOCUS

000006DD 3CFF                       CMP     AL,0FFH
000006DF 7403                       JZ      short NO_SET_LOC
000006E1 A2[2303]                   MOV     [EXTERR_LOCUS],AL
                                NO_SET_LOC:
000006E4 5B                         POP     BX
000006E5 58                         POP     AX
000006E6 1F                         POP     DS
000006E7 C3                         retn

                                    ;EndProc CAL_LK

                                ;--------------------------------------------------------------------------
                                ;
                                ; Procedure Name : ETAB_LK
                                ;
                                ; Inputs:
                                ;   AX is error code
                                ;   [USER_IN_AX] has AH value of system call involved
                                ; Function:
                                ;   Make sure error code is appropriate to this call.
```

```
                                ; Outputs:
                                ;   AX MAY be mapped error code
                                ;   [EXTERR] = Input AX
                                ; Destroys ONLY AX and FLAGS
                                ;
                                ;--------------------------------------------------------------------------

                                    ;procedure ETAB_LK,NEAR

                                ETAB_LK: ; 10/08/2018 - Retro DOS v3.0
000006E8 1E                         PUSH    DS
000006E9 56                         PUSH    SI
000006EA 51                         PUSH    CX
000006EB 53                         PUSH    BX

                                    ;Context DS                 ; SS is DOSDATA

000006EC 16                         push    ss
000006ED 1F                         pop     ds

000006EE A3[2403]                   MOV     [EXTERR],AX         ; Set EXTERR with "real" error

                                                                ; I21_MAP_E_TAB is now in DOSCODE
000006F1 BE[0B00]                   MOV     SI,I21_MAP_E_TAB
000006F4 88C7                       MOV     BH,AL               ; Real code to BH
000006F6 8A1E[3B03]                 MOV     BL,[USER_IN_AX+1]   ; Sys call to BL
                                TABLK2:
                                    ; 15/05/2019 - Retro DOS v4.0
000006FA 2E                         cs
000006FB AD                         lodsw  ; MSDOS 6.0 (MSDOS 6.21 - MSDOS.SYS, DOSCODE:447Dh)

                                    ; 18/07/2018 - Retro DOS v3.0
                                    ;lodsw          ; IBMDOS.COM (MSDOS 3.3) - Offset 16F7h

000006FC 3CFF                       CMP     AL,0FFH             ; End of table?
000006FE 740C                       JZ      short NOT_IN_TABLE  ; Yes
00000700 38D8                       CMP     AL,BL               ; Found call?
00000702 740C                       JZ      short GOT_CALL      ; Yes
00000704 86E0                       XCHG    AH,AL               ; Count to AL
00000706 30E4                       XOR     AH,AH               ; Make word for add
00000708 01C6                       ADD     SI,AX               ; Next table entry
0000070A EBEE                       JMP     short TABLK2

                                NOT_IN_TABLE:
0000070C 88F8                       MOV     AL,BH               ; Restore original code
0000070E EB0C                       JMP     SHORT NO_MAP

                                GOT_CALL:
00000710 88E1                       MOV     CL,AH
00000712 30ED                       XOR     CH,CH               ; Count of valid err codes to CX
                                CHECK_CODE:
                                    ; 15/05/2019 - Retro DOS v4.0
00000714 2E                         cs
00000715 AC                         lodsb  ; MSDOS 6.0 (MSDOS 6.21 - MSDOS.SYS, DOSCODE:4497h)

                                    ; 18/07/2018
                                    ;lodsb          ; IBMDOS.COM (MSDOS 3.3) - Offset 1710h

00000716 38F8                       CMP     AL,BH               ; Code OK?
00000718 7402                       JZ      short NO_MAP        ; Yes
0000071A E2F8                       LOOP    CHECK_CODE
                                NO_MAP:
0000071C 30E4                       XOR     AH,AH               ; AX is now valid code
0000071E 5B                         POP     BX
0000071F 59                         POP     CX
00000720 5E                         POP     SI
00000721 1F                         POP     DS
00000722 C3                         retn

                                    ;EndProc ETAB_LK

                                ; 18/07/2018 - Retro DOS v3.0
                                ;--------------------------------------------------------------------------
                                ; BREAK <DOS 2F Handler and default NET 2F handler>

                                ;IF installed ; (*)

                                ;--------------------------------------------------------------------------
                                ;
                                ; Procedure Name : SetBad
                                ;
                                ; SetBad sets up info for bad functions
                                ;
                                ;--------------------------------------------------------------------------

                                SetBad:
                                    ;mov    ax,1
00000723 B80100                     MOV     AX,error_invalid_function ; ALL NET REQUESTS get inv func

                                    ; MSDOS 3.3
                                    ;;mov   byte [cs:EXTERR_LOCUS],1
                                    ;MOV    byte [CS:EXTERR_LOCUS],errLOC_Unk

                                ;   set up ds to point to DOSDATA

                                    ; 15/05/2019 - Retro DOS v4.0
                                    ; MSDOS 6.0
00000726 1E                         push    ds

                                    ;getdseg <ds>
00000727 2E8E1E[0700]               mov     ds,[cs:DosDSeg]

0000072C C606[2303]01               MOV     byte [EXTERR_LOCUS],errLOC_Unk ; 1

00000731 1F                         pop     ds              ;hkn; restore ds

00000732 F9                         STC
00000733 C3                         retn

                                ;--------------------------------------------------------------------------
                                ;
                                ; Procedure Name : BadCall
                                ;
                                ; BadCall is the initial routine for bad function calls
                                ;
                                ;--------------------------------------------------------------------------

                                BadCall:
00000734 E8ECFF                     call    SetBad
00000737 CB                         retf

                                ;--------------------------------------------------------------------------
                                ;
                                ; OKCall always sets carry to off.
```

```
                                ;
                                ;--------------------------------------------------------------------------
                                OKCall:
00000738 F8                         CLC
00000739 CB                         retf

                                ;--------------------------------------------------------------------------
                                ;
                                ; Procedure Name : INT2F
                                ;
                                ; INT 2F handler works as follows:
                                ;   PUSH    AX
                                ;   MOV     AX,multiplex:function
                                ;   INT     2F
                                ;   POP     ...
                                ; The handler itself needs to make the AX available for the various routines.
                                ;
                                ;--------------------------------------------------------------------------

                                ; 15/05/2019 - Retro DOS v4.0

                                ;KERNEL_SEGMENT equ 70h
                                ; 04/11/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
                                DOSBIODATASEG equ 70h

                                ; retrodos4.s - offset in BIOSDATA
                                bios_i2f equ 5

                                ;PUBLIC     Int2F
                                ;INT2F      PROC    FAR

                                ; 15/05/2019
                                ; DOSCODE:44BDh (MSDOS 6.21, MSDOS.SYS)

                                ; 04/11/2022
                                ; DOSCODE:44B0h (MSDOS 5.0, MSDOS.SYS)
                                ; 15/12/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
                                ; 18/07/2018 - Retro DOS v3.0
                                ; 05/01/2024 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMDOS.COM)
                                ; PCDOS 7.1 IBMDOS.COM - DOSCODE:45DAh
                                INT2F:
                                ; Offset 172Fh in IBMDOS.COM (MSDOS 3.3), 1987
                                INT2FNT:
                                ;ASSUME CS:DOSCODE,DS:NOTHING,ES:NOTHING,SS:NOTHING
0000073A FB                     STI
                                ;cmp    ah,11h
0000073B 80FC11                 CMP     AH,MultNET
0000073E 750A                   JNZ     short INT2FSHR
                                TestInstall:
00000740 08C0                   OR      AL,AL
00000742 7403                   JZ      short Leave2F
                                BadFunc:
00000744 E8DCFF                 CALL    SetBad

                                ;entry  Leave2F
                                Leave2F:
00000747 CA0200                 RETF    2                       ; long return + clear flags off stack

                                INT2FSHR:
                                ;cmp    ah,10h
0000074A 80FC10                 CMP     AH,MultSHARE            ; is this a share request
0000074D 74F1                   JZ      short TestInstall       ; yes, check for installation
                                INT2FNLS:
                                ;cmp    ah,14h
0000074F 80FC14                 CMP     AH,NLSFUNC              ; is this a DOS 3.3 NLSFUNC request
00000752 74EC                   JZ      short TestInstall       ; yes check for installation
                                INT2FDOS:
                                ;ASSUME CS:DOSCODE,DS:NOTHING,ES:NOTHING,SS:NOTHING

                                ; 18/07/2018
                                ; MSDOS 3.3
                                ;;cmp   ah,12h
                                ;CMP    AH,MultDOS
                                ;jz     short DispatchDOS
                                ;iret

                                ; 15/05/2019
                                ; MSDOS 6.0
                                ;cmp    ah,12h  ; 07/12/2022
00000754 80FC12                 CMP     AH,MultDOS
00000757 7503                   JNZ     short check_win             ;check if win386 broadcast
00000759 E93F02                 jmp     DispatchDOS

                                ; .... win386 ....

                                check_win:
                                ;cmp    ah,16h
0000075C 80FC16                 cmp     ah,MultWin386           ; Is this a broadcast from Win386?
0000075F 7408                   je      short Win386_Msg

                                ; M044
                                ; Check if the callout is from Winoldap indicating swapping out or in
                                ; of Windows. If so, do special action of going and saving last para
                                ; of the Windows memory arena which Winoldap does not save due to a
                                ; bug

00000761 80FC46                 cmp     ah,WINOLDAP ; 46h       ; from Winoldap?
                                ;jne    short next_i2f          ; no, chain on
                                ; 15/12/2022
                                ;jmp    winold_swap             ; yes, do desired action
00000764 7460                   je      short winold_swap
00000766 E92301                 jmp     next_i2f

                                ; 15/12/2022
                                ;next_i2f:
                                ;   ;;;jmp  bios_i2f
                                ;   ;;jmp   far ptr 70h:5 ; MSDOS 6.21 (MSDOS.SYS, DOSCODE:44F1h)
                                ;   ;jmp    KERNEL_SEGMENT:bios_i2f
                                ;   ; 04/11/2022
                                ;   jmp     DOSBIODATASEG:bios_i2f

                                ;   IRET                            ; This assume that we are at the head
                                                                    ; of the list
                                ;INT2F      ENDP

                                ; 15/05/2019 - Retro DOS v4.0

                                ; We have received a message from Win386. There are three possible
                                ; messages we could get from Win386:
                                ;
                                ; Init              - for this, we set the IsWin386 flag and return a pointer
                                ;             to the Win386 startup info structure.
```

```
                                ; Exit            - for this, we clear the IsWin386 flag.
                                ; DOSMGR query    - for this, we need to indicate that instance data
                                ;                   has already been handled. this is indicated by setting
                                ;                   CX to a non-zero value.

                                Win386_Msg:
00000769 1E                         push    ds

                                    ;getdseg <DS>               ; ds is DOSDATA
0000076A 2E8E1E[0700]               mov     ds,[cs:DosDSeg]

                                    ; For WIN386 2.xx instance data

0000076F 3C03                       cmp     al,3                ; win386 2.xx instance data call?
00000771 7503                       jne     short Win386_Msg_exit
00000773 E94801                     jmp     OldWin386Init       ; yes, return instance data
                                Win386_Msg_exit:
00000776 3C06                       cmp     al,Win386_Exit  ; 6 ; is it an exit call?
00000778 7503                       jne     short Win386_Msg_devcall
0000077A E94A01                     jmp     Win386_Leaving
                                Win386_Msg_devcall:
0000077D 3C07                       cmp     al,Win386_Devcall ; 7 ; is it call from DOSMGR?
0000077F 7503                       jne     short Win386_Msg_init
00000781 E97E01                     jmp     Win386_Query
                                Win386_Msg_init:
00000784 3C05                       cmp     al,Win386_Init ; 5  ; is it an init call?
00000786 7403                       je      short Win386_Starting
00000788 E90001                     jmp     win_nexti2f         ; no, return

                                Win386_Starting:
                                    ; 05/01/2024 - Retro DOS v5.0
                                    ; PCDOS 7.1 IBMDOS.COM - DOSCODE:4630h
                                    ;;;
0000078B 50                         push    ax
0000078C 51                         push    cx
0000078D 57                         push    di
0000078E 06                         push    es
0000078F 1E                         push    ds
00000790 07                         pop     es
00000791 BF[FB01]                   mov     di,INBUF
00000794 B90500                     mov     cx,5
00000797 FC                         cld
                                Win386_s_floop:
00000798 B8434F                     mov     ax,'CO' ; 4F43h
0000079B AB                         stosw
0000079C B84E20                     mov     ax,'N ' ; 204Eh
0000079F AB                         stosw
000007A0 83C737                     add     di,55   ; (write 5 times 'CON ' and skip 55 bytes between them)
                                                    ; what for ?
000007A3 E2F3                       loop    Win386_s_floop
000007A5 07                         pop     es
000007A6 5F                         pop     di
000007A7 59                         pop     cx
000007A8 58                         pop     ax
                                    ;;;

                                    ; 17/12/2022
000007A9 F6C201                     test    dl,1
                                    ;test   dx,1                ; is this really win386?
000007AC 7403                       jz      short Win386_vchk   ; YES! go and handle it
000007AE E9DA00                     jmp     win_nexti2f         ; NO! It's win 286 dos extender! M002
                                Win386_vchk:
                                    ; M018 -- start of block changes
                                    ; The VxD needs to be loaded only for Win 3.0. If version is greater
                                    ; than 030Ah, we skip the VxD presence check

                                ;M067 -- Begin changes
                                ; If Win 3.0 is run, the VxD ptr has been initialized. If Win 3.1 is now
                                ;run, it tries to unnecesarily load the VxD even though it is not needed.
                                ;So, we null out the VxD ptr before the check.

                                    ;mov    word [Win386_Info+6],0
000007B1 C706[E70E]0000             mov     word [Win386_Info+Win386_SIS.Virt_Dev_File_Ptr],0
                                    ;mov    word [Win386_Info+8],0
000007B7 C706[E90E]0000             mov     word [Win386_Info+Win386_SIS.Virt_Dev_File_Ptr+2],0

                                ;M067 -- End changes

                                ;ifdef JAPAN
                                ;   cmp     di,0300h            ; version >= 300 i.e 3.10 ;M037
                                ;else
                                    ;cmp    di,030Ah            ; version >= 30a i.e 3.10 ;M037
                                    ; 05/01/2024 - PCDOS 7.1 - Retro DOS v5.0
000007BD 81FF0004                   cmp     di,0400h            ; version >= 400 ; 05/01/2024
                                ;endif
                                    ;jae    noVxD31             ; yes, VxD not needed  ;M037
000007C1 724E                       jb      short Win386_vxd
000007C3 E9DD00                     jmp     noVxD31

                                    ; 15/12/2022
                                winold_swap:
000007C6 1E                         push    ds
000007C7 06                         push    es
000007C8 56                         push    si
000007C9 57                         push    di
000007CA 51                         push    cx

                                    ;getdseg <ds>               ;ds = DOSDATA
000007CB 2E8E1E[0700]               mov     ds,[cs:DosDSeg]

000007D0 3C01                       cmp     al,1                ;swap Windows out call
000007D2 751B                       jne     short swapin        ;no, check if Swap in call
000007D4 E88801                     call    getwinlast
000007D7 1E                         push    ds
000007D8 07                         pop     es
000007D9 8EDE                       mov     ds,si               ;ds = memory arena of Windows
000007DB 31F6                       xor     si,si
                                    ;mov    di,6 ; 05/01/2024
000007DD BF[0600]                   mov     di,WinoldPatch1 ; 6
000007E0 B90800                     mov     cx,8
000007E3 FC                         cld
                                    ;push   cx
000007E4 F3A4                       rep     movsb               ;save first 8 bytes
                                    ;pop    cx
                                    ; 25/06/2024
000007E6 B108                       mov     cl,8
                                    ;mov    di,1176h ; 05/01/2024
000007E8 BF[7611]                   mov     di,WinoldPatch2 ; 1176h
000007EB F3A4                       rep     movsb               ;save next 8 bytes
000007ED EB1B                       jmp     short winold_done
                                swapin:
000007EF 3C02                       cmp     al,2                ;swap Windows in call?
000007F1 7517                       jne     short winold_done   ;no, something else, pass it on
000007F3 E86901                     call    getwinlast
```

```
000007F6 8EC6                   mov     es,si
000007F8 31FF                   xor     di,di
000007FA BE[0600]               mov     si,WinoldPatch1
000007FD B90800                 mov     cx,8
00000800 FC                     cld
                                ;push   cx
00000801 F3A4                   rep     movsb                   ;restore first 8 bytes
                                ;pop    cx
                                ; 25/06/2024
00000803 B108                   mov     cl,8
00000805 BE[7611]               mov     si,WinoldPatch2
00000808 F3A4                   rep     movsb                   ;restore next 8 bytes
                        winold_done:
0000080A 59                     pop     cx
0000080B 5F                     pop     di
0000080C 5E                     pop     si
0000080D 07                     pop     es
0000080E 1F                     pop     ds
0000080F EB7B                   jmp     short next_i2f          ;chain on
                                ; 15/12/2022
                                ;jmp    next_i2f

                        Win386_vxd:
00000811 50                     push    ax
00000812 53                     push    bx
00000813 51                     push    cx
00000814 52                     push    dx
00000815 56                     push    si
00000816 57                     push    di                      ; save regs !!dont change order!!

00000817 8B1E[8C00]             mov     bx,[UMB_HEAD]           ; M062 - Start
0000081B 83FBFF                 cmp     bx,0FFFFh               ; Q: have umbs been initialized
0000081E 741F                   je      short Vxd31             ; N: continue
                                                                ; Y: save arena associated with
                                                                ;    umb_head

00000820 C606[DA0D]01           mov     byte [UmbSaveFlag],1    ; indicate that we're saving
                                                                ; umb_arena
00000825 1E                     push    ds
00000826 06                     push    es

                                ;mov    ax,ds
                                ;mov    es,ax                   ; es - > dosdata
                                ; 05/01/2024
00000827 1E                     push    ds
00000828 07                     pop     es

00000829 8EDB                   mov     ds,bx
0000082B 31F6                   xor     si,si                   ; ds:si -> umb_head

                                ; 05/01/2024 PCDOS 7.1
                                ;;;
                                ;clc    ; not necessary (XOR already clears CF)

                        restore_ubmhead:            ; !! PCDOS 7.1 bug !!
                                                    ; jump from 'Win386_Leaving' here was/is wrong
                                                    ; (DI and SI would be reversed for 'Win386_Leaving')
                                                    ; Erdogan Tan - 05/01/2024
                                ;;;

0000082D FC                     cld

0000082E BF[2F11]               mov     di,UmbSave1
00000831 B90B00                 mov     cx,11
00000834 F3A4                   rep     movsb

00000836 BF[D50D]               mov     di,UmbSave2
                                ;mov    cx,5
                                ; 18/12/2022
00000839 B105                   mov     cl,5
0000083B F3A4                   rep     movsb

                                ; 05/01/2024 PCDOS 7.1
                                ;;;
                                ;jnb    short restore_ubmhead_c ; (not jumped from 'Win386_Leaving')
                                ;jmp    restore_ubmhead_ok ; (jumped from 'Win386_Leaving' just after 'stc')
                        ;restore_ubmhead_c:
                                ;;;

0000083D 07                     pop     es
0000083E 1F                     pop     ds                      ; M062 - End

                        Vxd31:
                                ;test   byte [DOS_FLAG],2
0000083F F606[8600]02           test    byte [DOS_FLAG],SUPPRESS_WINA20     ; M066
00000844 7408                   jz      short Dont_Supress              ; M066
00000846 5F                     pop     di                              ; M066
00000847 5E                     pop     si                              ; M066
00000848 5A                     pop     dx                              ; M066
00000849 59                     pop     cx                              ; M066
0000084A 5B                     pop     bx                              ; M066
0000084B 58                     pop     ax                              ; M066
0000084C EB55                   jmp     short noVxD31                   ; M066

                                ; We check here if the VxD is available in the root of the boot drive.
                                ; We do an extended open to suppress any error messages

                        Dont_Supress:
0000084E A0[6900]               mov     al,[BOOTDRIVE]
00000851 0440                   add     al,'A' - 1              ; get drive letter
00000853 A2[F812]               mov     [VxDpath],al            ; path is root of bootdrive
                                ;mov    ah,ExtOpen  ;6Ch        ; extended open
                                ;mov    al,0                    ; no extended attributes
                                ; 18/12/2022
00000856 B8006C                 mov     ax,ExtOpen<<8 ; 6C00h
00000859 BB8020                 mov     bx,2080h                ; read access, compatibility mode
                                                                ; no inherit, suppress crit err
0000085C B90700                 mov     cx,7                    ; hidden,system,read-only attr
                                ;05/01/2024
                                ;inc    dx                      ; dx bit 0 = 1 ; fail if file does not exist
0000085F BA0100                 mov     dx,1                    ; fail if file does not exist

00000862 BE[F812]               mov     si,VxDpath ; "c:\\wina20.386"
                                                                ; path of VxD file
00000865 BFFFFF                 mov     di,0FFFFh               ; no extended attributes

00000868 CD21                   int     21h                     ; do extended open

0000086A 5F                     pop     di
0000086B 5E                     pop     si
0000086C 5A                     pop     dx
0000086D 59                     pop     cx

0000086E 7321                   jnc     short VxDthere          ; we found the VxD, go ahead
```

```
                                    ; We could not find the VxD. Cannot let windows load. Return cx != 0
                                    ; to indicate error to Windows after displaying message to user that
                                    ; VxD needs to be present to run Windows in enhanced mode.

00000870 52                         push    dx
00000871 1E                         push    ds
00000872 56                         push    si
00000873 BE[300A]                   mov     si,NoVxDErrMsg
00000876 0E                         push    cs
00000877 1F                         pop     ds
00000878 B96300                     mov     cx,VxDMesLen ; 99     ;
0000087B B402                       mov     ah,2                  ; write char to console
0000087D FC                         cld
                                vxdlp:
0000087E AC                         lodsb
0000087F 86D0                       xchg    dl,al                 ; get char in dl
00000881 CD21                       int     21h
00000883 E2F9                       loop    vxdlp

00000885 5E                         pop     si
00000886 1F                         pop     ds
00000887 5A                         pop     dx
00000888 5B                         pop     bx
00000889 58                         pop     ax                    ;all registers restored
0000088A 41                         inc     cx                    ;cx != 0 to indicate error
                                    ; 15/12/22022
                                    ;jmp    win_nexti2f           ;chain on
                                    ;jmp    short win_nexti2f

                                    ; 15/12/2022
                                win_nexti2f:
0000088B 1F                         pop     ds
                                    ;jmp    short next_i2f        ; go to BIOS i2f handler
                                    ; 15/12/2022
                                next_i2f:
                                    ;;;jmp  bios_i2f
                                    ;;jmp   far ptr 70h:5 ; MSDOS 6.21 (MSDOS.SYS, DOSCODE:44F1h)
                                    ;jmp    KERNEL_SEGMENT:bios_i2f
                                    ; 04/11/2022
0000088C EA05007000                 jmp     DOSBIODATASEG:bios_i2f

                                VxDthere:
00000891 89C3                       mov     bx,ax
00000893 B43E                       mov     ah,CLOSE ; 3Eh
00000895 CD21                       int     21h                   ;close the file

                                    ; Update the VxD ptr in the instance data structure with path to VxD

                                    ; 07/12/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
                                    ;mov    bx,Win386_Info
                                    ;mov    word [bx+Win386_SIS.Virt_Dev_File_Ptr],VxDpath
                                    ;mov    word [bx+Win386_SIS.Virt_Dev_File_Ptr+2],ds
                                    ; 15/12/2022
00000897 C706[E70E][F812]           mov     word [Win386_Info+Win386_SIS.Virt_Dev_File_Ptr],VxDpath
0000089D 8C1E[E90E]                 mov     word [Win386_Info+Win386_SIS.Virt_Dev_File_Ptr+2],ds

000008A1 5B                         pop     bx
000008A2 58                         pop     ax
                                noVxD31:
                                    ; M018; End of block changes

000008A3 800E[5B0F]01               or      byte [IsWin386],1     ; Indicate WIN386 present
000008A8 800E[650D]01               or      byte [redir_patch],1  ; Enable critical sections; M002

                                    ; M002;
                                    ; Save the previous es:bx (instance data ptr) into our instance table

000008AD 52                         push    dx                    ; M002
000008AE 89DA                       mov     dx,bx                 ; M002
                                                                  ; point ES:BX to Win386_Info ; M002
000008B0 BB[E10E]                   mov     bx,Win386_Info
000008B3 895702                     mov     [bx+2],dx             ; M002
000008B6 8C4704                     mov     [bx+4],es             ; M002
000008B9 5A                         pop     dx                    ; M002
000008BA 1E                         push    ds                    ; M002
000008BB 07                         pop     es                    ; M002
                                    ;jmp    win_nexti2f           ; M002
                                    ; 15/12/2022
000008BC EBCD                       jmp     short win_nexti2f

                                    ; 15/12/2022
                                    ; Code to return Win386 2.xx instance table
                                OldWin386Init:
000008BE 58                         pop     ax                    ; discard ds pushed on stack
000008BF BE[FC10]                   mov     si,OldInstanceJunk
                                                                  ; ds:si = instance table
000008C2 B84852                     mov     ax,5248h ; 'HR'               ; indicate instance data present
                                    ;jmp    next_i2f
                                    ; 15/12/2022
000008C5 EBC5                       jmp     short next_i2f

                                Win386_Leaving:
                                    ; 15/12/2022
000008C7 F6C201                     test    dl,1
                                    ;test   dx,1                  ; is this really win386?
                                    ;jz     short Win386_Leaving_c
                                    ;jmp    win_nexti2f           ; NO! It's win 286 dos extender! M002
                                    ; 15/12/2022
000008CA 75BF                       jnz     short win_nexti2f

                                Win386_Leaving_c:
                                                                  ; M062 - Start
000008CC 803E[DA0D]01               cmp     byte [UmbSaveFlag],1  ; Q: was umb_arena saved at win start
                                                                  ;    up.
000008D1 7523                       jne     short noumb           ; N: not saved
000008D3 C606[DA0D]00               mov     byte [UmbSaveFlag],0  ; Y: clear UmbSaveFlag and restore
                                                                  ;    previously saved umb_head

                                    ; 05/01/2024 - PCDOS 7.1 (has BUG here!)
                                    ; PCDOS 7.1 IBMDOS.COM - DOSCODE:472Bh
                                    ;;;
                                    ;push   ax ; (not necessary)
                                    ;push   es
                                    ;push   cx
                                    ;push   si
                                    ;push   di
                                    ;mov    es,[UMB_HEAD]
                                    ;xor    di,di
                                    ;stc
                                    ;jmp    restore_ubmhead       ; !! PCDOS 7.1 bug !! IBMDOS.COM - DOSCODE:4737h
                                    ;                      ; (jumped code does not restore umbhead,
                                    ;                      ; MSDOS 6.22 "Win386_Leaving" code was/is correct,
                                    ;                      ; modified code -in PCDOS 7.1 IBMDOS.COM- is wrong)
```

```
                                    ;                       ; Erdogan Tan - 05/01/2024
                                    ;
                                    ;;;

                                    ; 05/01/2024 - Retro DOS v5.0 (Modified PCDOS 7.1)
                                    ;push   ax ; (not necessary)
000008D8 06                         push    es
000008D9 51                         push    cx
000008DA 56                         push    si
000008DB 57                         push    di

                                    ;mov    ax,[UMB_HEAD]
                                    ;mov    es,ax
                                    ; 05/01/2024
000008DC 8E06[8C00]                 mov     es,[UMB_HEAD]
000008E0 31FF                       xor     di,di                   ; es:di -> umb_head

000008E2 FC                         cld

000008E3 BE[2F11]                   mov     si,UmbSave1
000008E6 B90B00                     mov     cx,11
000008E9 F3A4                       rep     movsb
000008EB BE[D50D]                   mov     si,UmbSave2
                                    ;mov    cx,5
                                    ; 18/12/2022
000008EE B105                       mov     cl,5
000008F0 F3A4                       rep     movsb

                                restore_ubmhead_ok:         ; 05/01/2024 - PCDOS 7.1 IBMDOS.COM - DOSCODE:473Ah
000008F2 5F                         pop     di
000008F3 5E                         pop     si
000008F4 59                         pop     cx
000008F5 07                         pop     es
                                    ; 05/01/2024
                                    ;pop    ax
                                noumb:                                      ; M062 - End
                                    ;and    byte [IsWin386],0       ; Win386 is gone
                                    ; 26/06/2024
000008F6 8026[5B0F]FE               and     byte [IsWin386],0FEh ; ~1
000008FB 8026[650D]00               and     byte [redir_patch],0  ; Disable critical sections ; M002
00000900 EB89                       jmp     short win_nexti2f

                                ;   ; 15/12/2022
                                ;   ; Code to return Win386 2.xx instance table
                                ;OldWin386Init:
                                ;   pop     ax                      ; discard ds pushed on stack
                                ;   mov     si,OldInstanceJunk
                                ;                                   ; ds:si = instance table
                                ;   mov     ax,5248h ; 'RH'                 ; indicate instance data present
                                ;   ;jmp    next_i2f
                                ;   ; 15/12/2022
                                ;   jmp     short _next_i2f

                                Win386_Query:
00000902 83FB15                     cmp     bx,Win386_DOSMGR ; 15h; is this from DOSMGR?
00000905 7584                       jne     short win_nexti2f       ; no, ignore it & chain to next
00000907 09C9                       or      cx,cx                   ; is it an instance query?
00000909 7508                       jnz     short dosmgr_func       ; no, some DOSMGR query
0000090B 41                         inc     cx                      ; indicate that data is instanced
                                    ;
                                    ; M001; We were previously returning a null ptr in es:bx. This will not work.
                                    ; M001; WIN386 needs a ptr to a table in es:bx with the following offsets:
                                    ; M001;
                                    ; M001; OFFSETS STRUC
                                    ; M001;     Major_version  db      ?
                                    ; M001;     Minor_version  db      ?
                                    ; M001;     SaveDS         dw      ?
                                    ; M001;     SaveBX         dw      ?
                                    ; M001;     Indos          dw      ?
                                    ; M001;     User_id        dw      ?
                                    ; M001;     CritPatch      dw      ?
                                    ; M001; OFFSETS    ENDS
                                    ; M001;
                                    ; M001; User_Id is the only variable really important for proper functioning
                                    ; M001; of Win386. The other variables are used at init time to patch stuff
                                    ; M001; out. In DOS 5.0, we do the patching ourselves. But we still need to
                                    ; M001; pass this table because Win386 depends on this table to get the
                                    ; M001; User_Id offset.
                                    ; M001;
0000090C BB[4D0F]                   mov     bx,Win386_DOSVars       ; M001
0000090F 1E                         push    ds                      ; M001
00000910 07                         pop     es                      ; es:bx points at offset table ; M001
00000911 EB40                       jmp     short PopIret           ; M001

                                ; 15/12/2022
                                ;   ; Code to return Win386 2.xx instance table
                                ;OldWin386Init:
                                ;   pop     ax                      ; discard ds pushed on stack
                                ;   mov     si,OldInstanceJunk
                                ;                                   ; ds:si = instance table
                                ;   mov     ax,5248h ; 'RH'                 ; indicate instance data present
                                ;   ;jmp    next_i2f
                                ;   ; 15/12/2022
                                ;   jmp     short _next_i2f

                                dosmgr_func:
00000913 49                         dec     cx
00000914 7435                       jz      short win386_patch      ; call to patch DOS
00000916 49                         dec     cx
00000917 743A                       jz      short PopIret           ; remove DOS patches, ignore
00000919 49                         dec     cx
0000091A 7439                       jz      short win386_size       ; get size of DOS data structures
0000091C 49                         dec     cx
0000091D 7428                       jz      short win386_inst       ; instance more data
                                    ;dec    cx
                                    ;jnz    short PopIret           ; no functions above this
                                    ; 05/01/2024 (PCDOS 7.1 IBMDOS.COM DOSCODE:4771h)
0000091F E232                       loop    PopIret

                                    ; Get DOS device driver size -- es:di points at device driver header
                                    ; In DOS 4.x, the para before the device header contains an arena
                                    ; header for the driver.

00000921 8CC0                       mov     ax,es                   ; ax = device header segment

                                    ; We check to see if we have a memory arena for this device driver.
                                    ; The way to do this would be to look at the previous para to see if
                                    ; it has a 'D' marking it as an arena and also see if the owner-field
                                    ; in the arena is the same as the device header segment. These two
                                    ; checks together should take care of all cases

00000923 48                         dec     ax                      ; get arena header
00000924 06                         push    es
00000925 8EC0                       mov     es,ax                   ; arena header for device driver
```

```
                                    cmp     byte [es:di],'D'        ; is it a device arena?
                                    jnz     short cantsize          ; no, cant size this driver
                                    inc     ax                      ; get back device header segment
                                    cmp     [es:di+1],ax            ; owner field pointing at driver?
                                    jnz     short cantsize          ; no, not a proper arena

                                    mov     ax,[es:di+3]            ; get arena size in paras
                                    pop     es

                                    ; We have to multiply by 16 to get the number of bytes in (bx:cx)
                                    ; Speed is not critical and so we choose the shortest method
                                    ; -- use "mul"

                                    mov     bx,16
                                    mul     bx
                                    mov     cx,ax
                                    mov     bx,dx
                                    jmp     short win386_done       ; return with device driver size
                                cantsize:
                                    pop     es
                                    xor     ax,ax
                                win386_inst:        ; 05/01/2024
                                    xor     dx,dx                   ; ask DOSMGR to use its methods
                                    jmp     short PopIret           ; return

                                win386_patch:
                                    ; dx contains bits marking the patches to be applied. We return
                                    ; the field with all bits set to indicate that all patches have been
                                    ; done

                                    mov     bx,dx                   ; move patch bitfield to bx
                                    ;jmp    short win386_done       ; done, return
                                    ; 15/12/2022
                                    ; 15/12/2022
                                win386_done:
                                    mov     ax,WIN_OP_DONE          ; 0B97Ch
                                    mov     dx,DOSMGR_OP_DONE       ; 0A2ABh
                                PopIret:
                                    pop     ds
                                    iret

                                win386_size:
                                    ; Return the size of DOS data structures -- currently only CDS size

                                    ; 17/12/2022
                                    test    dl,1
                                    ;test   dx,1                    ; check for CDS size bit
                                    jz      short PopIret           ; no, unknown structure -- return

                                    mov     cx,curdirLen   ; 88     ; cx = CDS size
                                    jmp     short win386_done       ; return with the size

                                ; 05/01/2024
                                %if 0
                                win386_inst:
                                    ; WIN386 check to see if DOS has identified the CDS,SFT and device
                                    ; chain as instance data. Currently, we let the WIN386 DOSMGR handle
                                    ; this by returning a status of not previously instanced. The basic
                                    ; structure of these things have not changed and so the current
                                    ; DOSMGR code should be able to work it out

                                    xor     dx,dx                   ; make sure dx has a not done value
                                    jmp     short PopIret           ; skip done indication
                                %endif

                                    ; 15/12/2022
                                ;win386_done:
                                ;   mov     ax,WIN_OP_DONE          ; 0B97Ch
                                ;   mov     dx,DOSMGR_OP_DONE       ; 0A2ABh
                                ;PopIret:
                                ;   pop     ds
                                ;   iret                            ; return back up the chain

                                    ; 15/12/2022
                                ;win_nexti2f:
                                    ;pop    ds
                                    ;jmp    next_i2f                ; go to BIOS i2f handler

                                ;End WIN386 support

                                ; 15/05/2019

                                ;M044; Start of changes
                                ; Winoldap has a bug in that its calculations for the Windows memory image
                                ; to save is off by 1 para. This para can happen to be a Windows arena if the
                                ; DOS top of memory happens to be at an odd boundary (as is the case when
                                ; UMBs are present). This is because Windows builds its arenas only at even
                                ; para boundaries. This arena now gets trashed when Windows is swapped back
                                ; in leading to a crash. Winoldap issues callouts when it swaps Windows out
                                ; and back in. We sit on these callouts. On the Windows swapout, we save the
                                ; last para of the Windows memory block and then restore this para on the
                                ; Windows swapin callout.

                                getwinlast:
                                    ; 07/12/2022
0000095F 8B36[3003]                 mov     si,[CurrentPDB]
00000963 4E                         dec     si
00000964 8EC6                       mov     es,si
00000966 2603360300                 add     si,[es:3]
0000096B C3                         retn

                                ; 15/12/2022
                                %if 0
                                winold_swap:
                                    push    ds
                                    push    es
                                    push    si
                                    push    di
                                    push    cx

                                    ;getdseg <ds>                  ;ds = DOSDATA
                                    mov     ds,[cs:DosDSeg]

                                    cmp     al,1                    ;swap Windows out call
                                    jne     short swapin            ;no, check if Swap in call
                                    call    getwinlast
                                    push    ds
                                    pop     es
                                    mov     ds,si                   ;ds = memory arena of Windows
                                    xor     si,si
                                    mov     di,WinoldPatch1
                                    mov     cx,8
                                    cld
```

```
                                    push    cx
                                    rep     movsb                   ;save first 8 bytes
                                    pop     cx
                                    mov     di,WinoldPatch2
                                    rep     movsb                   ;save next 8 bytes
                                    jmp     short winold_done
                                swapin:
                                    cmp     al,2                    ;swap Windows in call?
                                    jne     short winold_done       ;no, something else, pass it on
                                    call    getwinlast
                                    mov     es,si
                                    xor     di,di
                                    mov     si,WinoldPatch1
                                    mov     cx,8
                                    cld
                                    push    cx
                                    rep     movsb                   ;restore first 8 bytes
                                    pop     cx
                                    mov     si,WinoldPatch2
                                    rep     movsb                   ;restore next 8 bytes
                                winold_done:
                                    pop     cx
                                    pop     di
                                    pop     si
                                    pop     es
                                    pop     ds
                                    jmp     next_i2f                ;chain on

                                %endif

                                ;M044; End of changes

                                ; ----------------------------------------------------------------------
                                ; 06/01/2024 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMDOS.COM/MSDOS.SYS)
                                ; PCDOS 7.1 IBMDOS.COM - DOSCODE:4811h

                                    ; INT 2Fh AX=1231h
                                    ; Windows95 - SET/CLEAR "REPORT WINDOWS TO DOS PROGRAMS" FLAG
                                    ;                       (Ref: Ralf Brown's Interrupt List)
                                int_2Fh_1231h:
0000096C 1E                         push    ds
0000096D 2E8E1E[0700]               mov     ds,[cs:DosDSeg]
00000972 31C0                       xor     ax,ax ; 0
00000974 08D2                       or      dl,dl
00000976 7507                       jnz     short not_1231_dl_0
00000978 C606[5C0F]01               mov     byte [IsWin386+1],1   ; set byte after "IsWIN386" to 01h
0000097D EB1A                       jmp     short int_2f_1231h_retn

                                    ;nop

                                not_1231_dl_0:
0000097F 80FA01                     cmp     dl,1
00000982 7507                       jne     short not_1231_dl_1   ; clear "IsWIN386" bit 1
00000984 800E[5B0F]02               or      byte [IsWin386],2     ; set "IsWIN386" bit 1
00000989 EB0E                       jmp     short int_2f_1231h_retn

                                    ;nop

                                not_1231_dl_1:
0000098B 80FA02                     cmp     dl,2
0000098E 7507                       jne     short not_1231_dl_2
00000990 8026[5B0F]FD               and     byte [IsWin386],0FDh  ; clear bit 1
00000995 EB02                       jmp     short int_2f_1231h_retn
                                not_1231_dl_2:
00000997 40                         inc     ax                    ; return error, ax = 1
00000998 F9                         stc
                                int_2f_1231h_retn:
00000999 1F                         pop     ds
0000099A C3                         retn

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

                                ; 15/05/2019

                                ; 06/01/2024 - Retro DOS v5.0
                                ; PCDOS 7.1 IBMDOS.COM - DOSCODE:4842h

                                DispatchDOS:
0000099B 2EFF36[D401]               PUSH    word [CS:FOO]         ; push return address
000009A0 2EFF36[D601]               PUSH    word [CS:DTab]        ; push table address
000009A5 50                         PUSH    AX                    ; push index
000009A6 55                         PUSH    BP
000009A7 89E5                       MOV     BP,SP
                                            ; stack looks like:
                                            ;   0   BP
                                            ;   2   DISPATCH
                                            ;   4   TABLE
                                            ;   6   RETURN
                                            ;   8   LONG-RETURN
                                            ;   C   FLAGS
                                            ;   E   AX
000009A9 8B460E                     MOV     AX,[BP+0Eh]           ; get AX value
000009AC 5D                         POP     BP
000009AD E84E0E                     call    TableDispatch
000009B0 E991FD                     JMP     BadFunc               ; return indicates invalid function

                                INT2F_etcetera:
                                    ;entry  DosGetGroup
                                DosGetGroup:
                                    ; MSDOS 3.3
                                    ;push   cs
                                    ;pop    ds
                                    ;retn

                                    ; MSDOS 6.0
                                ;SR; Cannot use CS now
                                ;
                                ;   PUSH    CS
                                ;   POP     DS

                                    ; 04/11/2022
                                    ; (MSDOS 5.0 MSDOS.SYS - DOSCODE:46FBh)

                                    ;getdseg <ds>
000009B3 2E8E1E[0700]               mov     ds,[cs:DosDSeg]
000009B8 C3                         retn

                                    ;entry  DOSInstall
                                DOSInstall:
000009B9 B0FF                       MOV     AL,0FFh
000009BB C3                         retn

                                ;ENDIF ; (*)
```

```
6694
6695
6696                                    ; 15/05/2019 - Retro DOS v4.0
6697                                    ; 06/01/2024 - Retro DOS v5.0
6698
6699                                    ;--------------------------------------------------------------------------
6700                                    ;
6701                                    ; Procedure Name : RW32_CONVERT
6702                                    ;
6703                                    ;Input: same as ABSDRD and ABSDWRT
6704                                    ;    ES:BP -> DPB
6705                                    ;Functions: convert 32bit absolute RW input parms to 16bit input parms
6706                                    ;Output: carry set when CX=-1 and drive is less then 32mb
6707                                    ;    carry clear, parms ok
6708                                    ;
6709                                    ;--------------------------------------------------------------------------
6710
6711                                        ; 06/01/2024
6712                                    RW32_CONVERT:
6713 000009BC 83F9FF                       CMP     CX,-1                   ;>32mb  new format ?  ;AN000;
6714                                        ;inc    cx ; *  ; 01 -> 0
6715 000009BF 7423                         JZ      short new32format       ;>32mb  yes           ;AN000;
6716                                        ;dec    cx
6717
6718                                        ;cmp    word [es:bp+0Fh],0
6719 000009C1 26837E0F00                   cmp     word [es:bp+DPB.FAT_SIZE],0 ; PCDOS 7.1
6720 000009C6 741A                         jz      short rw32_conv_err ; FAT32 fs
6721
6722 000009C8 50                           PUSH    AX                      ;>32mb  save ax               ;AN000;
6723 000009C9 52                           PUSH    DX                      ;>32mb  save dx               ;AN000;
6724                                        ;mov    ax,[es:bp+0Dh]
6725 000009CA 268B460D                     MOV     AX,[ES:BP+DPB.MAX_CLUSTER] ;>32mb  get max cluster # ;AN000;
6726                                        ;mov    dl,[es:bp+4]
6727 000009CE 268A5604                     MOV     DL,[ES:BP+DPB.CLUSTER_MASK] ;>32mb          ;AN000;
6728 000009D2 80FAFE                       CMP     DL,0FEh ; 254           ;>32mb  removable ?   ;AN000;
6729 000009D5 7407                         JZ      short letold            ;>32mb  yes           ;AN000;
6730                                        ;INC    DL                      ;>32mb                ;AN000;
6731                                        ; 17/12/2022
6732 000009D7 42                           inc     dx
6733 000009D8 30F6                         XOR     DH,DH                   ;>32mb  dx = sector/cluster ;AN000;
6734 000009DA F7E2                         MUL     DX                      ;>32mb  dx:ax= max sector # ;AN000;
6735 000009DC 09D2                         OR      DX,DX   ; (clears CF)   ;>32mb  > 32mb ?      ;AN000;
6736                                    letold:
6737 000009DE 5A                           POP     DX                      ;>32mb  restore dx    ;AN000;
6738 000009DF 58                           POP     AX                      ;>32mb  restore ax    ;AN000;
6739 000009E0 7418                         JZ      short old_style         ; cf=0 ;>32mb  no           ;AN000;
6740
6741                                        ; 06/01/2024
6742                                        ;push   ds
6743                                        ;;getdseg <ds>
6744                                        ;mov    ds,[cs:DosDSeg]
6745                                        ;mov    word [AbsDskErr],207h ;>32mb  bad address mark
6746                                        ;pop    ds
6747                                    rw32_conv_err:
6748 000009E2 F9                           STC                             ;>32mb                ;AN000;
6749 000009E3 C3                           retn                            ;>32mb                ;AN000;
6750
6751                                    new32format:
6752                                        ;mov    dx,[bx+2]
6753 000009E4 8B5702                       MOV     DX,[BX+ABS_32RW.SECTOR_RBA+2] ;>32mb        ;AN000;
6754
6755 000009E7 1E                           push    ds                      ; set up ds to DOSDATA
6756                                        ;getdseg <ds>
6757 000009E8 2E8E1E[0700]                 mov     ds,[cs:DosDSeg]
6758 000009ED 8916[0706]                   MOV     [HIGH_SECTOR],DX        ;>32mb                ;AN000;
6759 000009F1 1F                           pop     ds
6760
6761 000009F2 8B17                         mov     dx,[bx]
6762                                        ;MOV    DX,[BX+ABS_32RW.SECTOR_RBA]  ;>32mb         ;AN000;
6763                                        ;mov    cx,[bx+4]
6764 000009F4 8B4F04                       MOV     CX,[BX+ABS_32RW.ABS_RW_COUNT] ;>32mb        ;AN000;
6765                                        ;lds    bx,[bx+6]
6766 000009F7 C55F06                       LDS     BX,[BX+ABS_32RW.BUFFER_ADDR] ;>32mb         ;AN000;
6767                                    old_style:                          ;>32mb                ;AN000;
6768                                        ; 06/01/2024
6769                                        ; cf=0
6770                                        ;CLC                            ;>32mb                ;AN000;
6771 000009FA C3                           retn                            ;>32mb                ;AN000;
6772
6773                                    ;--------------------------------------------------------------------------
6774                                    ;
6775                                    ; Procedure Name : Fastxxx_Purge
6776                                    ;
6777                                    ; Input: None
6778                                    ; Functions: Purge Fastopen/ Cache Buffers
6779                                    ; Output: None
6780                                    ;
6781                                    ;--------------------------------------------------------------------------
6782
6783                                    ; 05/11/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
6784
6785                                    Fastxxx_Purge:
6786 000009FB 50                           PUSH    AX                      ; save regs.   ;AN000;
6787 000009FC 56                           PUSH    SI                                             ;AN000;
6788 000009FD 52                           PUSH    DX                                             ;AN000;
6789                                    topen:
6790 000009FE 1E                           push    ds                      ; set up ds to DOSDATA
6791                                        ;getdseg <ds>
6792 000009FF 2E8E1E[0700]                 mov     ds,[cs:DosDSeg]
6793
6794 00000A04 F606[4611]80                 TEST    byte [FastOpenFlg],Fast_yes ; 80h
6795                                                                       ; fastopen installed ?;AN000;
6796 00000A09 1F                           pop     ds
6797 00000A0A 740B                         JZ      short nofast            ; no                  ;AN000;
6798 00000A0C B401                         MOV     AH,FastOpen_ID ; 1                             ;AN000;
6799                                    dofast:
6800 00000A0E B005                         MOV     AL,FONC_purge  ;5       ; purge               ;AN000;
6801                                        ;;mov   dl,[es:bp+0]
6802                                        ; 05/11/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
6803                                        ;MOV    DL,[ES:BP+DPB.DRIVE]  ; set up drive number ;AN000;
6804                                        ; 15/12/2022
6805 00000A10 268A5600                     mov     dl,[es:bp]
6806                                        ;invoke Fast_Dispatch           ; call fastopen/seek  ;AN000;
6807 00000A14 E85923                       call    Fast_Dispatch
6808                                    nofast:
6809 00000A17 5A                           POP     DX                                             ;AN000;
6810 00000A18 5E                           POP     SI                      ; restore regs         ;AN000;
6811 00000A19 58                           POP     AX                                             ;AN000;
6812 00000A1A C3                           retn                            ; exit
6813
6814                                    ;==========================================================================
6815                                    ; DOSMES.INC (MSDOS 6.0, 1991)
6816                                    ;==========================================================================
6817                                    ; 29/04/2019 - Retro DOS v4.0
```

```
                                ;include dossym.inc
                                ;include dosmac.inc
                                ;include doscntry.inc

                                ; DOSCODE Segment

                                ; 17/07/2018 - Retro DOS v3.0  [ DOSMES.INC (MSDOS 3.3, 1987) ]
                                ; ----------------------------------------------------------------------------
                                ;include divmes.inc

                                ; DOSCODE:48C3h (PCDOS 7.1, IBMDOS.COM) - 06/04/2024 -
                                ; ------------------------------------
                                ; DOSCODE:4778h (MSDOS 6.21, MSDOS.SYS)
                                ; ------------------------------------
                                ; DOSCODE:476Bh (MSDOS 5.0, MSDOS.SYS) - 05/11/2022 -

                                ; THIS IS THE ONLY DOS "MESSAGE". IT DOES NOT NEED A TERMINATOR.
                                    ;PUBLIC DIVMES

00000A1B 0D0A44697669646520-    DIVMES:     DB      13,10,"Divide overflow",13,10
00000A24 6F766572666C6F770D-
00000A2D 0A

                                    ;PUBLIC DivMesLen
                                DivMesLen:
00000A2E 1300                       DW      $-DIVMES  ; 19 ; Length of the above message in bytes

                                ; DOSCODE:478Dh (MSDOS 6.21, MSDOS.SYS)
                                ; ------------------------------------
                                ; DOSCODE:4780h (MSDOS 5.0, MSDOS.SYS) - 05/11/2022 -

                                ; (MSDOS 6.0)
                                ; VxD not found error message

                                NoVxDErrMsg:
00000A30 596F75206D75737420-        db  'You must have the file WINA20.386 in the root of your boot drive'
00000A39 686176652074686520-
00000A42 66696C652057494E41-
00000A4B 32302E33383620696E-
00000A54 2074686520726F6F74-
00000A5D 206F6620796F757220-
00000A66 626F6F742064726976-
00000A6F 65
00000A70 0D0A746F2072756E20-        db  0Dh,0Ah,'to run Windows in Enhanced Mode',0Dh,0Ah
00000A79 57696E646F77732069-
00000A82 6E20456E68616E6365-
00000A8B 64204D6F64650D0A

                                VxDMesLen equ $ - NoVxDErrMsg  ; 99

                                ; 13/05/2019 - Retro DOS v4.0
                                ; 05/11/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)

                                ;include yesno.asm  (MNSDOS 6.0)
                                ; ------------------------------------
                                ; DOSCODE:47F0h (MSDOS 6.21, MSDOS.SYS)
                                ; DOSCODE:47E3h (MSDOS 5.0, MSDOS.SYS) - 05/11/2022 -

                                ; This is for country Yes and No

                                ; 06/01/2024 (Retro DOS 5.0 - PCDOS 7.1 IBMDOS.COM)
                                ;NLS_YES:   db 'Y'
                                ;NLS_NO:    db 'N'
                                ;NLS_yes2:  db 'y'
                                ;NLS_no2:   db 'n'

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

                                ; DOSCODE:493Bh (PCDOS 7.1, IBMDOS.COM) - 06/04/2024 -
                                ; ------------------------------------
                                ; DOSCODE:47F4h (MSDOS 6.21, MSDOS.SYS)
                                ; DOSCODE:47E7h (MSDOS 5.0, MSDOS.SYS) - 05/11/2022 -

                                ;SUBTTL EDIT FUNCTION ASSIGNMENTS AND HEADERS

                                ; The following two tables implement the current buffered input editing
                                ; routines. The tables are pairwise associated in reverse order for ease
                                ; in indexing. That is; The first entry in ESCTAB corresponds to the last
                                ; entry in ESCFUNC, and the last entry in ESCTAB to the first entry in ESCFUNC.

                                    ;PUBLIC CANCHAR
                                CANCHAR:
00000A93 1B                         DB      CANCEL  ; 1Bh   ;Cancel line character

                                    ;PUBLIC ESCCHAR
                                ESCCHAR:
00000A94 00                         DB      ESCCH   ; 0     ;Lead-in character for escape sequences

                                    ;IF     NOT Rainbow

                                ESCTAB:     ; LABEL BYTE

                                    ;IF     IBM
00000A95 40                         DB      64              ; Ctrl-Z - F6
00000A96 4D                         DB      77              ; Copy one char - -->
00000A97 3B                         DB      59              ; Copy one char - F1
00000A98 53                         DB      83              ; Skip one char - DEL
00000A99 3C                         DB      60              ; Copy to char - F2
00000A9A 3E                         DB      62              ; Skip to char - F4
00000A9B 3D                         DB      61              ; Copy line - F3
00000A9C 3D                         DB      61              ; Kill line (no change to template ) - Not used
00000A9D 3F                         DB      63              ; Reedit line (new template) - F5
00000A9E 4B                         DB      75              ; Backspace - <--
00000A9F 52                         DB      82              ; Enter insert mode - INS (toggle)
00000AA0 52                         DB      82              ; Exit insert mode - INS (toggle)
00000AA1 41                         DB      65              ; Escape character - F7
00000AA2 41                         DB      65              ; End of table
                                    ;ENDIF

                                ESCEND: ; LABEL BYTE

                                ESCTABLEN EQU ESCEND-ESCTAB

                                ESCFUNC: ; LABEL WORD

00000AA3 [DF19]                     short_addr  GETCH       ; Ignore the escape sequence
00000AA5 [5C1A]                     short_addr  TWOESC
00000AA7 [511B]                     short_addr  EXITINS
00000AA9 [511B]                     short_addr  ENTERINS
00000AAB [571A]                     short_addr  BACKSP
00000AAD [3D1B]                     short_addr  REEDIT
00000AAF [441A]                     short_addr  KILNEW
00000AB1 [D31A]                     short_addr  COPYLIN
```

```
6930 00000AB3 [051B]                    short_addr  SKIPSTR
6931 00000AB5 [D91A]                    short_addr  COPYSTR
6932 00000AB7 [FC1A]                    short_addr  SKIPONE
6933 00000AB9 [DE1A]                    short_addr  COPYONE
6934 00000ABB [DE1A]                    short_addr  COPYONE
6935 00000ABD [581B]                    short_addr  CTRLZ
6936
6937                                    ;ENDIF
6938
6939                                ; DOSMES.INC (MSDOS 6.0, 1991)
6940                                ; ----------------------------------------------------------------------------
6941                                ; DOSMES.ASM (MSDOS 2.11, 1983)
6942
6943                                ; OEMFunction key is expected to process a single function
6944                                ;   key input from a device and dispatch to the proper
6945                                ;   routines leaving all registers UNTOUCHED.
6946                                ;
6947                                ; Inputs:   CS, SS are DOSGROUP
6948                                ; Outputs:  None. This function is expected to JMP to onw of
6949                                ;           the following labels:
6950                                ;
6951                                ;           GetCh       - ignore the sequence
6952                                ;           TwoEsc      - insert an ESCChar in the buffer
6953                                ;           ExitIns     - toggle insert mode
6954                                ;           EnterIns    - toggle insert mode
6955                                ;           BackSp      - move backwards one space
6956                                ;           ReEdit      - reedit the line with a new template
6957                                ;           KilNew      - discard the current line and start from scratch
6958                                ;           CopyLin     - copy the rest of the template into the line
6959                                ;           SkipStr     - read the next character and skip to it in the template
6960                                ;           CopyStr     - read next char and copy from template to line until char
6961                                ;           SkipOne     - advance position in template one character
6962                                ;           CopyOne     - copy next character in template into line
6963                                ;           CtrlZ       - place a ^Z into the template
6964                                ; Registers that are allowed to be modified by this function are:
6965                                ;           AX, CX, BP
6966
6967                                ; 13/05/2019 - Retro DOS v4.0
6968                                ; ----------------------------------
6969                                ; DOSCODE:4820h (MSDOS 6.21, MSDOS.SYS)
6970
6971                                ; 05/11/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
6972                                ; ----------------------------------
6973                                ; DOSCODE:4813h (MSDOS 5.0, MSDOS.SYS)
6974
6975                                ; 06/01/2024 - Retro DOS v5.0
6976                                ; ----------------------------------
6977                                ; DOSCODE:4967h (PCDOS 7.1, IBMDOS.COM)
6978
6979                                OEMFunctionKey:
6980 00000ABF E8800E                    CALL    _$STD_CON_INPUT_NO_ECHO      ; Get the second byte of the sequence
6981 00000AC2 B10E                      MOV     CL,ESCTABLEN ; 14    ; length of table for scan
6982 00000AC4 57                        PUSH    DI                     ; save DI (cannot change it!)
6983 00000AC5 BF[950A]                  MOV     DI,ESCTAB            ; offset of second byte table
6984 00000AC8 06                        push    es
6985 00000AC9 0E                        push    cs
6986 00000ACA 07                        pop     es
6987 00000ACB F2AE                      REPNE   SCASB                  ; Look it up in the table
6988 00000ACD 07                        pop     es
6989 00000ACE 5F                        POP     DI                     ; restore DI
6990 00000ACF D1E1                      SHL     CX,1                   ; convert byte offset to word
6991 00000AD1 89CD                      MOV     BP,CX                  ; move to indexable register
6992                                    ;JMP    word [BP+ESCFUNC]    ; Go to the right routine
6993 00000AD3 2EFFA6[A30A]              JMP     word [CS:BP+ESCFUNC]
6994
6995                                ;DOSCODE ENDS
6996
6997                                ;============================================================================
6998                                ; TIME.ASM (MSDOS 6.0, 1991)
6999                                ;============================================================================
7000                                ; Retro DOS v3.0 - 18/07/2018
7001
7002                                ; SYSCALL.ASM (MSDOS 2.11, 1983)
7003                                ;----------------------------------------------------------------------------
7004                                ; Retro DOS v2.0 - 13/03/2018
7005
7006                                ;** TIME.ASM - System Calls and low level routines for DATE and TIME
7007
7008                                    ;BREAK <DATE AND TIME - SYSTEM CALLS 42,43,44,45>
7009
7010                                ;** $GET_DATE - Get Current Date
7011                                ;---------------------------------------
7012                                ;   ENTRY   none
7013                                ;   EXIT    (cx:dx) = current date
7014                                ;   USES    all
7015
7016                                ; 05/11/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
7017                                ; 06/01/2024 - Retro DOS v5.0 (Modified MSDOS 7.1 IBMDOS.COM)
7018
7019                                _$GET_DATE: ;System call 42
7020
7021 00000AD8 16                        PUSH    SS
7022 00000AD9 1F                        POP     DS
7023 00000ADA E8AD00                    CALL    READTIME        ;Check for rollover to next day
7024 00000ADD A1[5203]                  MOV     AX,[YEAR]
7025
7026                                ;   WARNING!!!! DAY and MONTH must be adjacently allocated!
7027
7028 00000AE0 8B1E[5003]                MOV     BX,[DAY]        ; fetch both day and month
7029 00000AE4 E890F9                    CALL    Get_User_Stack ;Get pointer to user registers
7030                                    ;MOV    [SI+6],BX       ;DH=month, DL=day
7031 00000AE7 895C06                    MOV     [SI+user_env.user_DX],BX
7032 00000AEA 05BC07                    ADD     AX,1980         ;Put bias back
7033                                    ;MOV    [SI+4],AX       ;CX=year
7034 00000AED 894404                    MOV     [SI+user_env.user_CX],AX
7035 00000AF0 36A0[5603]                MOV     AL,[SS:WEEKDAY]         ;hkn; SS override
7036                                RET20:      ; 05/11/2022
7037                                RET24:      ; 18/12/2022
7038 00000AF4 C3                        RETN
7039
7040                                ;** $SET_DATE - Set Current Date
7041                                ;---------------------------------------
7042                                ;   ENTRY   (cx:dx) = current date
7043                                ;   EXIT    (al) = -1 iff bad date
7044                                ;           (al) = 0 if ok
7045                                ;   USES    all
7046
7047                                _$SET_DATE: ;System call 43
7048
7049 00000AF5 B0FF                       MOV     AL,-1           ;Be ready to flag error
7050 00000AF7 81E9BC07                  SUB     CX,1980         ;Fix bias in year
7051                                    ;JC     SHORT RET24     ;Error if not big enough
7052                                    ; 05/11/2022
7053 00000AFB 72F7                      jc      short RET20
```

```
7054 00000AFD 83F977                    CMP     CX,119          ;Year must be less than 2100
7055 00000B00 77F2                      JA      SHORT RET24
7056 00000B02 08F6                      OR      DH,DH
7057                                    ;JZ     SHORT RET24
7058                                    ; 05/11/2022
7059 00000B04 74EE                      jz      short RET20
7060 00000B06 08D2                      OR      DL,DL
7061                                    ;JZ     SHORT RET24     ;Error if either month or day is 0
7062                                    ; 05/11/2022
7063 00000B08 74EA                      jz      short RET20
7064 00000B0A 80FE0C                    CMP     DH,12           ;Check against max. month
7065 00000B0D 77E5                      JA      SHORT RET24
7066 00000B0F 16                        PUSH    SS
7067 00000B10 1F                        POP     DS
7068                                    ;CALL   DODATE
7069                                    ; 18/12/2022
7070 00000B11 E90501                    jmp     DODATE
7071                                ;RET24:
7072                                    ;RETN
7073
7074                                ;** $GET_TIME - Get Current Time
7075                                ;---------------------------------------
7076                                ;   ENTRY   none
7077                                ;   EXIT    (cx:dx) = current time
7078                                ;   USES    all
7079
7080                                _$GET_TIME:             ;System call 44
7081
7082 00000B14 16                        PUSH    SS
7083 00000B15 1F                        POP     DS
7084 00000B16 E87100                    CALL    READTIME
7085 00000B19 E85BF9                    CALL    Get_User_Stack ;Get pointer to user registers
7086                                    ;MOV    [SI+6],DX
7087 00000B1C 895406                    MOV     [SI+user_env.user_DX],DX
7088                                    ;MOV    [SI+4],CX
7089 00000B1F 894C04                    MOV     [SI+user_env.user_CX],CX
7090                                set_time_ok:        ; 06/01/2024
7091 00000B22 30C0                      XOR     AL,AL
7092                                RET26:
7093 00000B24 C3                        RETN
7094
7095                                ;** $SET_TIME - Set Current Time
7096                                ;---------------------------------------
7097                                ;   ENTRY   (cx:dx) = time
7098                                ;   EXIT    (al) = 0 if Ok
7099                                ;           (al) = -1 if invalid
7100                                ;   USES    ALL
7101
7102                                _$SET_TIME:             ;System call 45
7103
7104 00000B25 B0FF                      MOV     AL,-1           ;Flag in case of error
7105 00000B27 80FD18                    CMP     CH,24           ;Check hours
7106 00000B2A 73F8                      JAE     SHORT RET26
7107 00000B2C 80F93C                    CMP     CL,60           ;Check minutes
7108 00000B2F 73F3                      JAE     SHORT RET26
7109 00000B31 80FE3C                    CMP     DH,60           ;Check seconds
7110 00000B34 73EE                      JAE     SHORT RET26
7111 00000B36 80FA64                    CMP     DL,100          ;Check 1/100's
7112 00000B39 73E9                      JAE     SHORT RET26
7113 00000B3B 51                        PUSH    CX
7114 00000B3C 52                        PUSH    DX
7115 00000B3D 16                        PUSH    SS
7116 00000B3E 1F                        POP     DS
7117
7118                                ; 07/02/2024
7119                                %if 0
7120                                    MOV     BX,TIMEBUF
7121                                    MOV     CX,6
7122                                    ; 06/02/2024 ; *
7123                                    ;;XOR   DX,DX
7124                                    ;;MOV   AX,DX
7125                                    ;xor    ax,ax
7126                                    ;cwd    ; 06/01/2024
7127                                    PUSH    BX
7128                                    ;CALL   SETREAD
7129                                    ; 06/02/2024 ; *
7130                                    call    SETREAD_X
7131                                %else
7132 00000B3F E8CA47                    call    SETREAD_XT
7133                                %endif
7134
7135 00000B42 1E                        PUSH    DS
7136 00000B43 C536[2E00]                LDS     SI,[BCLOCK]
7137 00000B47 E84B47                    CALL    DEVIOCALL2      ;Get correct day count
7138 00000B4A 1F                        POP     DS
7139 00000B4B 5B                        POP     BX
7140 00000B4C E8FA47                    CALL    SETWRITE
7141 00000B4F 8F06[BA03]                POP     WORD [TIMEBUF+4]
7142 00000B53 8F06[B803]                POP     WORD [TIMEBUF+2]
7143 00000B57 C536[2E00]                LDS     SI,[BCLOCK]
7144 00000B5B E83747                    CALL    DEVIOCALL2      ;Set the time
7145                                    ; 06/01/2024
7146                                    ;XOR    AL,AL
7147                                    ;RETN
7148 00000B5E EBC2                      jmp     short set_time_ok
7149
7150                                ; 11/07/2018 - Retro DOS v3.0
7151                                ; Retro DOS v2.0 - 14/03/2018
7152
7153                                FOURYEARS EQU 3*365 + 366  ; = 1461
7154
7155                                ;SUBTTL DATE16, READTIME, DODATE -- GUTS OF TIME AND DATE
7156                                ;--------------------------------------------------------
7157                                ; Date16 returns the current date in AX, current time in DX
7158                                ;   AX - YYYYYYYMMMMDDDDD  years months days
7159                                ;   DX - HHHHHMMMMMMSSSSS  hours minutes seconds/2
7160
7161                                DATE16:
7162
7163                                ;M048      Context DS
7164                                ;
7165                                ; Since this function can be called thru int 2f we shall not assume that SS
7166                                ; is DOSDATA
7167
7168                                    ;push   ss
7169                                    ;pop    ds
7170
7171                                    ;getdseg <ds>           ; M048
7172
7173                                    ; 13/05/2019 - Retro DOS v4.0
7174 00000B60 2E8E1E[0700]              mov     ds, [cs:DosDSeg]
7175
7176 00000B65 51                        PUSH    CX
7177 00000B66 06                        PUSH    ES
```

```
7178 00000B67 E82000                    CALL    READTIME
7179 00000B6A 07                        POP     ES
7180 00000B6B D0E1                      SHL     CL,1            ;Minutes to left part of byte
7181 00000B6D D0E1                      SHL     CL,1
7182 00000B6F D1E1                      SHL     CX,1            ;Push hours and minutes to left end
7183 00000B71 D1E1                      SHL     CX,1
7184 00000B73 D1E1                      SHL     CX,1
7185 00000B75 D0EE                      SHR     DH,1            ;Count every two seconds
7186 00000B77 08F1                      OR      CL,DH           ;Combine seconds with hours and minutes
7187 00000B79 89CA                      MOV     DX,CX
7188
7189                                ;   WARNING! MONTH and YEAR must be adjacently allocated
7190
7191 00000B7B A1[5103]                  MOV     AX,[MONTH]      ;Fetch month and year
7192 00000B7E B104                      MOV     CL,4
7193 00000B80 D2E0                      SHL     AL,CL           ;Push month to left to make room for day
7194 00000B82 D1E0                      SHL     AX,1
7195 00000B84 59                        POP     CX
7196 00000B85 0A06[5003]                OR      AL,[DAY]
7197                                RET21:
7198 00000B89 C3                        RETN
7199
7200                                ;--------------------------------------------------------
7201
7202                                READTIME:
7203
7204                                ;Gets time in CX:DX. Figures new date if it has changed.
7205                                ;Uses AX, CX, DX.
7206
7207 00000B8A C706[BD0D]0000            MOV     word [DATE_FLAG],0 ; reset date flag for CPMIO
7208 00000B90 56                        PUSH    SI
7209 00000B91 53                        PUSH    BX
7210
7211 00000B92 BB[B603]                  MOV     BX,TIMEBUF
7212                                ; 07/02/2024
7213                                %if 0
7214                                    MOV     CX,6
7215                                    ; 06/02/2024
7216                                    ;;XOR   DX,DX
7217                                    ;;MOV   AX,DX
7218                                    ;; 06/01/2024
7219                                    ;xor    ax,ax
7220                                    ;cwd
7221                                    ;CALL   SETREAD
7222                                    ; 06/02/2024
7223                                    call    SETREAD_X
7224                                %else
7225 00000B95 E87847                    call    SETREAD_XTC
7226                                %endif
7227 00000B98 1E                        PUSH    DS
7228 00000B99 C536[2E00]                LDS     SI,[BCLOCK]
7229 00000B9D E8F546                    CALL    DEVIOCALL2      ;Get correct date and time
7230 00000BA0 1F                        POP     DS
7231 00000BA1 5B                        POP     BX
7232 00000BA2 5E                        POP     SI
7233 00000BA3 A1[B603]                  MOV     AX,[TIMEBUF]
7234 00000BA6 8B0E[B803]                MOV     CX,[TIMEBUF+2]
7235 00000BAA 8B16[BA03]                MOV     DX,[TIMEBUF+4]
7236 00000BAE 3B06[5403]                CMP     AX,[DAYCNT]     ;See if day count is the same
7237                                    ;JZ     SHORT RET22
7238 00000BB2 74D5                      JZ      SHORT RET21 ; 18/07/2018
7239                                    ;cmp    ax,43830
7240 00000BB4 3D36AB                    CMP     AX,FOURYEARS*30 ;Number of days in 120 years
7241 00000BB7 733D                      JAE     SHORT RET22     ;Ignore if too large
7242 00000BB9 A3[5403]                  MOV     [DAYCNT],AX
7243 00000BBC 56                        PUSH    SI
7244 00000BBD 51                        PUSH    CX
7245 00000BBE 52                        PUSH    DX              ;Save time
7246 00000BBF 31D2                      XOR     DX,DX
7247                                    ;mov    cx,1461
7248 00000BC1 B9B505                    MOV     CX,FOURYEARS    ;Number of days in 4 years
7249 00000BC4 F7F1                      DIV     CX              ;Compute number of 4-year units
7250 00000BC6 D1E0                      SHL     AX,1
7251 00000BC8 D1E0                      SHL     AX,1
7252 00000BCA D1E0                      SHL     AX,1            ;Multiply by 8 (no. of half-years)
7253 00000BCC 89C1                      MOV     CX,AX           ;<240 implies AH=0
7254
7255 00000BCE BE[140D]                  MOV     SI,YRTAB        ;Table of days in each year
7256
7257                                    ;CALL   DSLIDE          ;Find out which of four years we're in
7258                                    ; 26/06/2024
7259 00000BD1 E82500                    call    DSLIDE1  ; ah = 0
7260 00000BD4 D1E9                      SHR     CX,1            ;Convert half-years to whole years
7261 00000BD6 7304                      JNC     SHORT SK        ;Extra half-year?
7262 00000BD8 81C2C800                  ADD     DX,200
7263                                SK:
7264 00000BDC E82400                    CALL    SETYEAR
7265 00000BDF B101                      MOV     CL,1            ;At least at first month in year
7266
7267 00000BE1 BE[1C0D]                  MOV     SI,MONTAB       ;Table of days in each month
7268
7269                                    ;CALL   DSLIDE          ;Find out which month we're in
7270                                    ; 26/06/2024
7271 00000BE4 E81200                    call    DSLIDE1  ; ah = 0
7272 00000BE7 880E[5103]                MOV     [MONTH],CL
7273 00000BEB 42                        INC     DX              ;Remainder is day of month (start with one)
7274 00000BEC 8816[5003]                MOV     [DAY],DL
7275 00000BF0 E88C00                    CALL    WKDAY           ;Set day of week
7276 00000BF3 5A                        POP     DX
7277 00000BF4 59                        POP     CX
7278 00000BF5 5E                        POP     SI
7279                                RET22:
7280 00000BF6 C3                        RETN
7281
7282                                ;--------------------------------------------------------
7283
7284                                DSLIDE:
7285                                    ; 26/06/2024
7286 00000BF7 B400                      MOV     AH,0
7287                                    ; 06/01/2024
7288                                    ; (AH=0)
7289                                DSLIDE1:
7290 00000BF9 AC                        LODSB                   ;Get count of days
7291 00000BFA 39C2                      CMP     DX,AX           ;See if it will fit
7292                                    ;JB     SHORT RET23     ;If not, done
7293 00000BFC 72F8                      jb      short RET22 ; 13/05/2019 - Retro DOS v4.0
7294 00000BFE 29C2                      SUB     DX,AX
7295 00000C00 41                        INC     CX              ;Count one more month/year
7296 00000C01 EBF6                      JMP     SHORT DSLIDE1
7297
7298                                ;--------------------------------------------------------
7299
7300                                SETYEAR:
7301
```

```
                                ;Set year with value in CX. Adjust length of February for this year.
                                ;
                                ; NOTE: This can also be called thru int 2f. If this is called then it will
                                ;     set DS to DOSDATA. Since the only guy calling this should be the DOS
                                ;   redir, DS will be DOSDATA anyway. It is going to be in-efficient to
                                ;   preserve DS as CHKYR is also called as a routine.
                                ;
                                    ; MSDOS 6.0 (18/07/2018) ; *
                                ;
                                    ;GETDSEG DS
                                ;
                                    ;PUSH   CS  ; *
                                    ;POP    DS  ; *
                                ;
                                    ; 13/05/2019 - Retro DOS v4.0
00000C03 2E8E1E[0700]               mov     ds,[cs:DosDSeg]
                                ;
                                    ; Offset 18CEh in IBMDOS.COM (MSDOS 3.3), 1987
                                    ; 05/11/2022
                                    ; DOSCODE:4970h in MSDOS.SYS (MSDOS 5.0), 1991
                                ;
00000C08 880E[5203]                 MOV     [YEAR],CL
                                CHKYR:
00000C0C F6C103                     TEST    CL,3            ;Check for leap year
00000C0F B01C                       MOV     AL,28
00000C11 7502                       JNZ     SHORT SAVFEB    ;28 days if no leap year
00000C13 FEC0                       INC     AL              ;Add leap day
                                SAVFEB:
00000C15 A2[1D0D]                   mov     [february],al
                                    ;MOV    [MONTAB+1],AL   ;Store for February
                                RET23:
00000C18 C3                         RETN
                                ;
                                ;---------------------------------------------------------
                                ;
                                DODATE:
00000C19 E8F0FF                     CALL    CHKYR           ;Set Feb. up for new year
00000C1C 88F0                       MOV     AL,DH
                                ;
00000C1E BB[1B0D]                   MOV     BX,MONTAB-1     ;DOSDATA:0D1Bh for MSDOS 6.21
                                                            ; 06/01/2024
                                                            ;DOSDATA:0D1Bh for PCDOS 7.1
                                ;
00000C21 D7                         XLAT                    ;Look up days in month
00000C22 38D0                       CMP     AL,DL
00000C24 B0FF                       MOV     AL,-1           ;Restore error flag, just in case
                                    ;JB     SHORT RET25     ;Error if too many days
00000C26 72F0                       jb      short RET23 ; 18/07/2018
00000C28 E8D8FF                     CALL    SETYEAR
                                ;
                                ; WARNING! DAY and MONTH must be adjacently allocated
                                ;
00000C2B 8916[5003]                 MOV     [DAY],DX        ;Set both day and month
00000C2F D1E9                       SHR     CX,1
00000C31 D1E9                       SHR     CX,1
                                    ;mov    ax,1461
00000C33 B8B505                     MOV     AX,FOURYEARS
00000C36 89D3                       MOV     BX,DX
00000C38 F7E1                       MUL     CX
00000C3A 8A0E[5203]                 MOV     CL,[YEAR]
00000C3E 80E103                     AND     CL,3
                                ;
00000C41 BE[140D]                   MOV     SI,YRTAB
                                ;
00000C44 89C2                       MOV     DX,AX
00000C46 D1E1                       SHL     CX,1            ;Two entries per year, so double count
00000C48 E84700                     CALL    DSUM            ;Add up the days in each year
00000C4B 88F9                       MOV     CL,BH           ;Month of year
                                ;
00000C4D BE[1C0D]                   MOV     SI,MONTAB
                                ;
00000C50 49                         DEC     CX              ;Account for months starting with one
00000C51 E83E00                     CALL    DSUM            ;Add up days in each month
00000C54 88D9                       MOV     CL,BL           ;Day of month
00000C56 49                         DEC     CX              ;Account for days starting with one
00000C57 01CA                       ADD     DX,CX           ;Add in to day total
00000C59 92                         XCHG    AX,DX           ;Get day count in AX
00000C5A A3[5403]                   MOV     [DAYCNT],AX
00000C5D 56                         PUSH    SI
00000C5E 53                         PUSH    BX
00000C5F 50                         PUSH    AX
                                ;
                                ; 07/02/2024
                                %if 0
                                    MOV     BX,TIMEBUF
                                    MOV     CX,6
                                    ; 06/02/2024 ; *
                                    ;;XOR   DX,DX
                                    ;;MOV   AX,DX
                                    ;; 06/01/2024
                                    ;xor    ax,ax
                                    ;cwd
                                    PUSH    BX
                                    ;CALL   SETREAD
                                    ; 06/02/2024 ; *
                                    call    SETREAD_X
                                %else
                                    call    SETREAD_XT
                                %endif
                                ;
00000C63 1E                         PUSH    DS
00000C64 C536[2E00]                 LDS     SI,[BCLOCK]
00000C68 E82A46                     CALL    DEVIOCALL2      ;Get correct date and time
00000C6B 1F                         POP     DS
00000C6C 5B                         POP     BX
00000C6D E8D946                     CALL    SETWRITE
00000C70 8F06[B603]                 POP     WORD [TIMEBUF]
00000C74 1E                         PUSH    DS
00000C75 C536[2E00]                 LDS     SI,[BCLOCK]
00000C79 E81946                     CALL    DEVIOCALL2      ;Set the date
00000C7C 1F                         POP     DS
00000C7D 5B                         POP     BX
00000C7E 5E                         POP     SI
                                WKDAY:
00000C7F A1[5403]                   MOV     AX,[DAYCNT]
00000C82 31D2                       XOR     DX,DX
00000C84 B90700                     MOV     CX,7
00000C87 40                         INC     AX
00000C88 40                         INC     AX              ;First day was Tuesday
00000C89 F7F1                       DIV     CX              ;Compute day of week
00000C8B 8816[5603]                 MOV     [WEEKDAY],DL
00000C8F 30C0                       XOR     AL,AL           ;Flag OK
                                RET25:
00000C91 C3                         RETN
```

```
                                ;--------------------------------------------------------
                                ;** DSUM - Compute the sum of a string of bytes
                                ;
                                ;   ENTRY   (cx) = byte count
                                ;           (ds:si) = byte address
                                ;           (dx) = sum register, initialized by caller
                                ;   EXIT    (dx) updated
                                ;   USES    ax, cx, dx, si, flags

                                DSUM:
00000C92 B400                       MOV     AH,0
00000C94 E305                       JCXZ    DSUM9 ; 13/05/2019 - Retro DOS v4.0
                                    ;JCXZ   RET25 ; 18/07/2018
                                DSUM1:
00000C96 AC                         LODSB
00000C97 01C2                       ADD     DX,AX
00000C99 E2FB                       LOOP    DSUM1
                                DSUM9:
00000C9B C3                         RETN

                                ;============================================================================
                                ; GETSET.ASM (MSDOS 6.0, 1991)
                                ;============================================================================
                                ; 29/04/2019 - Retro DOS v4.0
                                ; 18/07/2018 - Retro DOS v3.0 (GETSET.ASM, MSDOS 6.0, 1991)

                                ; 12/03/2018 - Retro DOS v2.0

                                ;TITLE      GETSET - GETting and SETting MS-DOS system calls
                                ;NAME       GETSET

                                ;CODE       SEGMENT BYTE PUBLIC  'CODE'
                                ;       ASSUME  SS:DOSGROUP,CS:DOSGROUP

                                ;USERNUM:
                                ;   DW      0                       ; 24 bit user number
                                ;       DB      0
                                ;;  IF      IBM
                                ;;OEMNUM: DB    0                   ; 8 bit OEM number
                                ;;  ELSE
                                ;OEMNUM: DB     0FFH                    ; 8 bit OEM number
                                ;;  ENDIF

                                ;MSVERS:           ; MS-DOS version in hex for $GET_VERSION
                                ;; 08/07/2018 - Retro DOS v3.0
                                ;MSMAJOR: DB       MAJOR_VERSION  ; DOS_MAJOR_VERSION
                                ;MSMINOR: DB       MINOR_VERSION  ; DOS_MINOR_VERSION

                                ;BREAK <$Get_Version -- Return MSDOS version number>
                                ;----------------------------------------------------------------------------

                                ; 05/11/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
                                ; DOSCODE:4A0Fh (MSDOS 5.0 MSDOS.SYS)

                                ; 06/01/2024 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMDOS.COM)
                                ; DOSCODE:4B5Fh (PCDOS 7.1 IBMDOS.COM)

                                _$GET_VERSION:

                                ; Inputs:
                                ;       None
                                ; Function:
                                ;       Return MS-DOS version number
                                ; Outputs:
                                ;       OEM number in BH
                                ;       User number in BL:CX (24 bits)
                                ;       Version number as AL.AH in binary
                                ;       NOTE: On pre 1.28 DOSs AL will be zero

                                ; MSDOS 6.0
                                ;
                                ;   Fake_Count is used to lie about the version numbers to support
                                ;   old binarys. See ms_table.asm for more info.
                                ;
                                ;           if input al = 00
                                ;             (bh) = OEM number
                                ;           else if input al = 01
                                ;             (bh) = version flags
                                ;
                                ;                   bits 0-2 = DOS internal revision
                                ;                   bits 3-7 = DOS type flags
                                ;                         bit 3    = DOS is in ROM
                                ;                         bit 4    = DOS in in HMA
                                ;                         bits 5-7 = reserved
                                ;               M007 change - only bit 3 is now valid. Other bits
                                ;               are 0 when AL = 1
                                ;
                                    ; 06/01/2024 (PCDOS 7.1 IBMDOS.COM)
00000C9C 36C50E[B203]               lds     cx, [ss:USERNUM]
00000CA1 8CDB                       mov     bx, ds

                                    ; MSDOS 3.3 (IBMDOS.COM, offset 196Dh)
                                    ;--------------------------------------
                                    ; MSDOS 6.21 (MSDOS.SYS, DOSCODE:4A1Ch)

00000CA3 16                             PUSH    SS
00000CA4 1F                             POP     DS

                                    ; 06/01/2024
                                    ;MOV    BX,[USERNUM+2]
                                        ;MOV        CX,[USERNUM]

                                    ; 13/05/2019 - Retro DOS v4.0

                                    ;If AL == 1, ROMDOS will return BH = dos internal version # &
                                    ;DOS flags

00000CA5 3C01                       cmp     AL,1
00000CA7 7502                       jne     short Norm_Vers

                                ;ifdef ROMDOS
                                ;   mov     BH,DOSINROM     ; Just set the bit for ROM version
                                ;                           (DOSINROM = 8)
                                ;else
00000CA9 30FF                           xor     bh,bh               ; Otherwise return 0
                                ;endif                              ;M007 end

                                Norm_Vers:
                                    ;MOV    AX,[MSVERS]  ; MSDOS 3.3

                                            ; MSDOS 6.0    ; MSVERS is a label in TABLE segment
                                    ; 26/06/2024 - Retro DOS v5.0
```

```
                                    ;       (PCDOS 7.1 IBMDOS.COM)
                                    ; 13/05/2019 - Retro DOS v4.0
                                    ;push   ds              ; Get the version number from the
00000CAB 8E1E[3003]                 mov     ds,[CurrentPDB]       ; current app's PSP segment
                                    ;mov    ax,[40h]
00000CAF A14000                     mov     ax,[PDB.Version] ; AX = DOS version number
                                    ; 07/12/2022
                                    ;pop    ds
00000CB2 E8C2F7                     call    Get_User_Stack
                                                            ; Put values for return registers
                                                            ; in the proper place on the user's
                                                            ; stack addressed by DS:SI
                                    ; 06/01/2024 (PCDOS 7.1 IBMDOS.COM)
                                gdrvfspc_ret:
                                        ;MOV       [SI+user_env.user_AX],AX
                                        MOV [SI],AX
                                        ;MOV       [SI+4],CX
00000CB5 8904                       mov     [SI+user_env.user_CX],CX
00000CB7 894C04                 set_user_bx:
                                    ;MOV    [SI+2],BX
00000CBA 895C02                     mov     [SI+user_env.user_BX],BX

00000CBD C3                         RETN

                                ; 18/07/2018 - Retro DOS v3.0

                                ;BREAK <$Get/Set_Verify_on_Write - return/set verify-after-write flag>
                                ;--------------------------------------------------------------------------

                                ;** $Get_Verify_On_Write - Get Status of Verify on write flag
                                ;
                                ;   ENTRY   none
                                ;   EXIT    (al) = value of VERIFY flag
                                ;   USES    all


                                _$GET_VERIFY_ON_WRITE:

                                ;hkn; SS override
00000CBE 36A0[FF02]                 MOV     AL,[SS:VERFLG] ; Retro DOS v2.0 - 12/03/2018
00000CC2 C3                         retn

                                ;** $Set_Verify_On_Write - Set Status of Verify on write flag
                                ;
                                ;   ENTRY   (al) = value of VERIFY flag
                                ;   EXIT    none
                                ;   USES    all

                                _$SET_VERIFY_ON_WRITE:

00000CC3 2401                       AND     AL,1
                                ;hkn; SS override
00000CC5 36A2[FF02]                 MOV     [SS:VERFLG],AL ; Retro DOS v2.0 - 12/03/2018
                                RET27:      ; 18/07/2018
00000CC9 C3                         retn

                                ; 19/07/2018 - Retro DOS v3.0

                                ;BREAK <$International - return country-dependent information>
                                ;--------------------------------------------------------------------------
                                ;
                                ; Procedure Name : $INTERNATIONAL
                                ;
                                ; Inputs:
                                ;   MOV     AH,International
                                ;   MOV     AL,country      (al = 0 => current country)
                                ;      [MOV BX,country]
                                ;   LDS     DX,block
                                ;   INT     21
                                ; Function:
                                ;   give users an idea of what country the application is running
                                ; Outputs:
                                ;   IF DX != -1 on input (get country)
                                ;     AL = 0 means return current country table.
                                ;     0<AL<0FFH means return country table for country AL
                                ;     AL = 0FF means return country table for country BX
                                ;     No Carry:
                                ;        Register BX will contain the 16-bit country code.
                                ;        Register AL will contain the low 8 bits of the country code.
                                ;        The block pointed to by DS:DX is filled in with the information
                                ;        for the particular country.
                                ;          BYTE  Size of this table excluding this byte and the next
                                ;          BYTE  Country code represented by this table
                                ;                 A sequence of n bytes, where n is the number specified
                                ;                 by the first byte above and is not > internat_block_max,
                                ;                 in the correct order for being returned by the
                                ;                 INTERNATIONAL call as follows:
                                ;          WORD   Date format 0=mdy, 1=dmy, 2=ymd
                                ;          5 BYTE  Currency symbol null terminated
                                ;          2 BYTE  thousands separator null terminated
                                ;          2 BYTE  Decimal point null terminated
                                ;          2 BYTE  Date separator null terminated
                                ;          2 BYTE  Time separator null terminated
                                ;          1 BYTE  Bit field.  Currency format.
                                ;                  Bit 0.  =0 $ before #  =1 $ after #
                                ;                  Bit 1.  no. of spaces between # and $ (0 or 1)
                                ;          1 BYTE  No. of significant decimal digits in currency
                                ;          1 BYTE  Bit field.  Time format.
                                ;                  Bit 0.  =0 12 hour clock  =1 24 hour
                                ;          DWORD  Call address of case conversion routine
                                ;          2 BYTE  Data list separator null terminated.
                                ;     Carry:
                                ;        Register AX has the error code.
                                ;   IF DX = -1 on input (set current country)
                                ;     AL = 0 is an error
                                ;     0<AL<0FFH means set current country to country AL
                                ;     AL = 0FF means set current country to country BX
                                ;     No Carry:
                                ;       Current country SET
                                ;       Register AL will contain the low 8 bits of the country code.
                                ;     Carry:
                                ;        Register AX has the error code.
                                ;--------------------------------------------------------------------------

                                ;procedure   $INTERNATIONAL,NEAR   ; DOS 3.3

                                ; 13/05/2019 - Retro DOS v4.0
                                ; DOSCODE:4A4Dh (MSDOS 6.21, MSDOS.SYS)

                                ; 05/11/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
                                ; DOSCODE:4A40h (MSDOS 5.0, MSDOS.SYS)

                                ; 06/01/2024 - Retro DOS v5.0 (Modified PCDOS 7.1 PCDOS.COM)
                                ; DOSCODE:4B8Dh (PCDOS 7.1, IBMDOS.COM)
```

```
                                _$INTERNATIONAL:  ; IBMDOS.COM (MSDOS 3.3), offset 1992h
                                
00000CCA 3CFF                       CMP     AL,0FFH
00000CCC 7404                       JZ      short BX_HAS_CODE     ; -1 means country code is in BX
00000CCE 88C3                       MOV     BL,AL                 ; Put AL country code in BX
00000CD0 30FF                       XOR     BH,BH
                                BX_HAS_CODE:
00000CD2 1E                         PUSH    DS
00000CD3 07                         POP     ES
00000CD4 52                         PUSH    DX
00000CD5 5F                         POP     DI                    ; User buffer to ES:DI
                                
                                ;hkn; SS is DOSDATA
                                ;   context DS
                                
00000CD6 16                         push    ss
00000CD7 1F                         pop     ds
                                
00000CD8 83FFFF                     CMP     DI,-1
00000CDB 745D                       JZ      short international_set
00000CDD 09DB                       OR      BX,BX
00000CDF 7505                       JNZ     short international_find
                                
                                ;hkn; country_cdpg is in DOSDATA segment.
00000CE1 BE[2A12]                   MOV     SI,COUNTRY_CDPG
                                
00000CE4 EB39                       JMP     SHORT international_copy
                                
                                international_find:
                                    ;MOV    BP,0                   ; flag it for GetCntry only
                                    ; 06/01/2024
00000CE6 31ED                       xor     bp,bp ; 0
00000CE8 E80A00                     CALL    international_get
00000CEB 7255                       JC      short errtn
                                    ;CMP    BX,0                   ; nlsfunc finished it ?
                                    ; 06/01/2024
00000CED 09DB                       or      bx,bx
00000CEF 752E                       JNZ     SHORT international_copy ; no, copy by myself
00000CF1 89D3                       MOV     BX,DX                  ; put country back
00000CF3 EB3A                       JMP     SHORT international_ok3
                                
                                international_get:
00000CF5 BE[2A12]                   MOV     SI,COUNTRY_CDPG
                                
                                ;hkn; country_cdpg is in DOSDATA segment.
                                ;hkn; use ss override to access COUNTRY_CDPG fields
                                
                                    ; MSDOS 3.3
                                    ;;cmp   bx,[SI+63h]
                                    ;CMP    BX,[SI+DOS_CCDPG.ccDosCountry]
                                    ;jz     short RET27
                                
                                    ; 13/05/2019 - Retro DOS v4.0
                                
                                    ; MSDOS 6.0
                                    ;cmp    bx,[ss:si+68h]
00000CF8 363B5C68                   CMP     BX,[ss:SI+DOS_CCDPG.ccDosCountry] ; = current country id
00000CFC 74CB                       jz      short RET27                   ; return if equal
                                
00000CFE 89DA                       MOV     DX,BX
00000D00 31DB                       XOR     BX,BX                 ; bx = 0, default code page
                                    ;CallInstall NLSInstall,NLSFUNC,0 ; check if NLSFUNC in memory
00000D02 B80014                     mov     ax,1400h
00000D05 CD2F                       int     2Fh    ; - Multiplex - NLSFUNC.COM - INSTALLATION CHECK
                                                   ; Return: AL = 00h not installed, OK to install
                                                   ; 01h not installed, not OK
                                                   ; FFh installed
00000D07 3CFF                       CMP     AL,0FFH
00000D09 7510                       JNZ     short interr          ; not in memory
                                
                                    ; 06/01/2024
00000D0B B80314                     mov     ax,1403h              ; set country info
                                
                                    ;cmp    bp,0
00000D0E 09ED                       or      bp,bp                 ; GetCntry ?
00000D10 7501                       JNZ     short stcdpg
                                
                                    ;CallInstall GetCntry,NLSFUNC,4      ; get country info
                                    ;mov    ax,1404h
00000D12 40                         inc     ax     ; AX = 1404h ; get country info
                                
                                    ; 06/01/2024
                                    ;int    2Fh    ; - Multiplex - NLSFUNC.COM - GET COUNTRY INFO
                                    ;              ; BX = code page, DX = country code,
                                    ;              ; DS:SI -> internal code page structure
                                    ;              ; ES:DI -> user buffer
                                    ;              ; Return: AL = status
                                    ;
                                    ;JMP    short chkok
                                
                                    ;nop
                                
                                stcdpg:
                                    ;CallInstall SetCodePage,NLSFUNC,3  ; set country info
                                    ; 06/01/2024
                                    ;mov     ax,1403h
                                gscdpg:
00000D13 CD2F                       int     2Fh    ; - Multiplex - NLSFUNC.COM - SET COUNTRY INFO
                                                   ; DS:SI -> internal code page structure
                                                   ; BX = code page, DX = country code
                                                   ; Return: AL = status
                                chkok:
00000D15 08C0                       or      al,al                 ; success ?
                                    ;retz                         ; yes
00000D17 74B0                       jz      short RET27
                                
                                setcarry:
00000D19 F9                         STC                           ; set carry
00000D1A C3                         retn
                                interr:
00000D1B B0FF                       MOV     AL,0FFH               ; flag nlsfunc error
00000D1D EBFA                       JMP     short setcarry
                                
                                international_copy:
                                
                                ;hkn; country_cdpg is in DOSDATA segment.
                                ;hkn; use ss override to access COUNTRY_CDPG fields
                                
                                    ; MSDOS 3.3
                                    ;;mov   bx,[SI+63h]
                                    ;mov    BX,[SI+DOS_CCDPG.ccDosCountry]
                                    ;mov    SI,COUNTRY_CDPG+DOS_CCDPG.ccDFormat ; 08/09/2018
```

```
                                    ; 13/05/2019 - Retro DOS v4.0
                                    ; MSDOS 6.0
                                    ;mov    bx,[ss:si+68h]
00000D1F 368B5C68                   MOV     BX,[ss:SI+DOS_CCDPG.ccDosCountry] ; = current country id
00000D23 BE[9612]                   MOV     SI,COUNTRY_CDPG+DOS_CCDPG.ccDFormat ; COUNTRY_CDPG + 108
                                    ;mov    cx,24
00000D26 B91800                     MOV     CX,OLD_COUNTRY_SIZE
                                    ; MSDOS 6.0
                                ;hkn;       must set up DS to SS so that international info can be copied
00000D29 1E                         push    ds
00000D2A 16                         push    ss                      ; cs -> ss
00000D2B 1F                         pop     ds
00000D2C F3A4                       REP     MOVSB                   ; copy country info
                                    ; MSDOS 6.0
00000D2E 1F                         pop     ds      ;hkn;   restore ds
                                international_ok3:
00000D2F E845F7                     call    Get_User_Stack
                                ;ASSUME     DS:NOTHING
                                    ;;MOV   [SI+2],BX
                                    ;MOV    [SI+user_env.user_BX],BX
                                    ; 26/06/2024 (PCDOS 7.1 IBMDOS.COM)
00000D32 E885FF                     call    set_user_bx
                                international_ok:
00000D35 89D8                       MOV     AX,BX                   ; Return country code in AX too.
                                ;SYS_RET_OK_jmp:
                                    ; 05/11/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
                                    ; 09/01/2024
                                ;nono:      ; 15/12/2022
                                SYS_RET_OK_jmp:
00000D37 E934F9                     jmp     SYS_RET_OK
                                international_set:
                                ;hkn; ASSUME       DS:DOSGROUP
                                ;ASSUME     DS:DOSDATA
00000D3A BD0100                     MOV     BP,1                    ; flag it for SetCodePage only
00000D3D E8B5FF                     CALL    international_get
00000D40 73F3                       JNC     short international_ok
                                errtn:
00000D42 3CFF                       CMP     AL,0FFH
00000D44 7403                       JZ      short errtn2
                                errtn1:
00000D46 E92FF9                     jmp     SYS_RET_ERR             ; return what we got from NLSFUNC
                                errtn2:
                                    ;error  error_invalid_function; NLSFUNC not existent
                                    ;mov    al,1
00000D49 B001                       mov     al,error_invalid_function
00000D4B EBF9                       jmp     short errtn1 ; 13/05/2019 - Retro DOS v4.0
                                ;errtn3:
                                ;   jmp     SYS_RET_ERR
                                ;EndProc $INTERNATIONAL
                                ; ---------------------------------------------------------------------------
                                    ; 06/01/2024 - Retro DOS v5.0 (Modified PCDOS 7.1 PCDOS.COM)
                                    ; DOSCODE:4C10h (PCDOS 7.1, IBMDOS.COM)
                                _$ExtCountryInfo:
                                                    ; INT 21h, AH = 70h
                                                    ; GET/SET INTERNATIONALIZATION INFORMATION
                                                    ; ****
                                                    ; AL = subfunction
                                                    ; 00h SET general internationalization info
                                                    ;    CX = buffer size (up to 38 bytes)
                                                    ;    DS:SI -> buffer containing internationalization info
                                                    ;  first three bytes are skipped, the rest is copied to
                                                    ;  somewhere in the DOS data segment
                                                    ; 01h SET extended internationalization info
                                                    ;    CX = number of bytes to set (up to 58 bytes)
                                                    ;    DS:SI -> buffer containing internationalization info
                                                    ; 02h GET extended internationalization info
                                                    ;    CX = buffer size in bytes (up to 58 bytes used)
                                                    ;    ES:DI -> buffer
                                                    ; ****
                                                    ; (Ref: Ralf Brown's Interrupt List) - had some mistakes -
00000D4D 3C02                       cmp     al,2
00000D4F 77F8                       ja      short errtn2
00000D51 06                         push     es
00000D52 16                         push     ss
00000D53 750F                       jnz     short ext_cntry_inf_1
00000D55 07                         pop     es              ; AX = GET 35 bytes info (from offset 3 to 37)
                                                            ; (38 bytes buffer is used)
00000D56 BF[9212]                   mov     di,_COUNTRY_ID
00000D59 268B45FE                   mov     ax,[es:di-2]    ; NEW_COUNTRY_SIZE = 38
00000D5D BB0300                     mov     bx,3            ; skip the 1st 3 bytes of the buffer
00000D60 01DE                       add     si,bx
00000D62 EB13                       jmp     short ext_cntry_inf_4
                                ext_cntry_inf_1:
00000D64 FEC8                       dec     al
00000D66 7506                       jnz     short ext_cntry_inf_2 ; AX = 2
                                                            ; AX = 1 (set)
00000D68 07                         pop     es
00000D69 BF[BA12]                   mov     di,_ENU         ; "ENU"
00000D6C EB04                       jmp     short ext_cntry_inf_3
                                ext_cntry_inf_2:
00000D6E 1F                         pop     ds              ; AX = 2 (get)
00000D6F BE[BA12]                   mov     si,_ENU         ; "ENU"
                                ext_cntry_inf_3:            ; CODE XREF: DOSCODE:4C2F^j
00000D72 31DB                       xor     bx,bx           ; 0
00000D74 B83A00                     mov     ax,58           ; 3Ah
                                ext_cntry_inf_4:
00000D77 39C1                       cmp     cx,ax           ; > 38 ? (58)
                                    ;jb     short ext_cntry_inf_5 ; no
00000D79 7602                       jna     short ext_cntry_inf_5 ; 06/01/2024
00000D7B 89C1                       mov     cx,ax           ; yes, decrease size to 38 (58)
```

```
                                ext_cntry_inf_5:
00000D7D 89C8                       mov     ax,cx           ; buffer (filled) size
00000D7F 29D9                       sub     cx,bx           ; copy byte count
00000D81 F3A4                       rep movsb
00000D83 07                         pop     es
00000D84 E91D03                     jmp     ret_ax_to_user_cx ; ax -> user's cx

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

                                ; 19/07/2018

                                ;BREAK <$GetExtCntry - return extended country-dependent information>

                                ;----------------------------------------------------------------------------
                                ;
                                ; Procedure Name : $GetExtCntry
                                ;
                                ; Inputs:
                                ;   if AL >= 20H
                                ;     AL= 20H    capitalize single char, DL= char
                                ;         21H    capitalize string, CX= string length
                                ;         22H    capitalize ASCIIZ string
                                ;         23H    YES/NO check, DL=1st char DH= 2nd char (DBCS)
                                ;         80H bit 0 = use normal upper case table
                                ;                 1 = use file upper case table
                                ;      DS:DX points to string
                                ;
                                ;   else
                                ;
                                ;   MOV     AH,GetExtCntry  ; DOS 3.3
                                ;   MOV     AL,INFO_ID      ( info type,-1 selects all )
                                ;   MOV     BX,CODE_PAGE    ( -1 = active code page )
                                ;   MOV     DX,COUNTRY_ID   ( -1 = active country )
                                ;   MOV     CX,SIZE         ( amount of data to return )
                                ;   LES     DI,COUNTRY_INFO ( buffer for returned data )
                                ;   INT     21
                                ; Function:
                                ;   give users extended country dependent information
                                ;   or capitalize chars
                                ; Outputs:
                                ;     No Carry:
                                ;        extended country info is succesfully returned
                                ;     Carry:
                                ;        Register AX has the error code.
                                ;        AX=0, NO   for YES/NO CHECK
                                ;           1, YES
                                ;----------------------------------------------------------------------------

                                ;procedure   $GetExtCntry,NEAR    ; DOS 3.3

                                    ; 06/11/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
                                    ; 06/01/2024 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMDOS.COM)

                                    ; MSDOS 6.0
                                _$GetExtCntry:
00000D87 3C20                       CMP     AL,CAP_ONE_CHAR       ; < 20H ?
00000D89 726A                       JB      short notcap
                                capcap:                           ;
00000D8B A880                       TEST    AL,UPPER_TABLE ; 80h  ; which upper case table
00000D8D 7505                       JNZ     short fileupper               ; file upper case

                                ;hkn; UCASE_TAB in DOSDATA
00000D8F BB[FC0A]                   MOV     BX,UCASE_TAB+2        ; get normal upper case
00000D92 EB05                       JMP     SHORT capit

                                fileupper:
                                    ; 06/01/2024 (PCDOS 7.1 IBMDOS.COM - DOSCODE:4C57h)
                                    ; ((Note: This must be a bugfix, because bit 7 of AX is 1 here!))
                                    ; AL >= 80h
00000D94 247F                       and     al,7Fh

                                ;hkn; FILE_UCASE_TAB in DOSDATA
00000D96 BB[7E0B]                   MOV     BX,FILE_UCASE_TAB+2 ; get file upper case
                                capit:                                ;
00000D99 3C20                       CMP     AL,CAP_ONE_CHAR ; 20h ; cap one char ?
00000D9B 750D                       JNZ     short chkyes          ; no
00000D9D 88D0                       MOV     AL,DL                 ; set up AL
00000D9F E81A50                     call    GETLET3               ; upper case it
00000DA2 E8D2F6                     call    Get_User_Stack        ; get user stack
                                    ;mov    [si+6],al
00000DA5 884406                     MOV     [SI+user_env.user_DX],AL ; user's DL=AL
00000DA8 EB24                       JMP     SHORT nono            ; done
                                chkyes:                           ;
00000DAA 3C23                       CMP     AL,CHECK_YES_NO       ; 23h  ; check YES or NO ?
00000DAC 7523                       JNZ     short capstring               ; no

00000DAE 31C0                       XOR     AX,AX                 ; presume NO

                                ;hkn; NLS_YES, NLS_NO, NLS_yes2, NLS_no2 is defined in msdos.cl3 which is
                                ;hkn; included in yesno.asm in the DOSCODE segment.

                                    ; 06/08/2018 - Retro DOS v3.0
                                    ; 13/05/2019 - Retro DOS v4.0
                                    ;cmp    dl,'Y'
00000DB0 2E3A16[3613]               CMP     DL,[cs:NLS_YES]               ; is 'Y' ?
00000DB5 7416                       JZ      short yesyes          ; yes
                                    ;cmp    dl,'y'
00000DB7 2E3A16[3813]               CMP     DL,[cs:NLS_yes2]      ; is 'y' ?
00000DBC 740F                       JZ      short yesyes          ; yes
                                    ;cmp    dl,'N'
00000DBE 2E3A16[3713]               CMP     DL,[cs:NLS_NO]        ; is  'N'?
00000DC3 7409                       JZ      short nono            ; no
                                    ;cmp    dl,'n'
00000DC5 2E3A16[3913]               CMP     DL,[cs:NLS_no2]               ; is 'n' ?
00000DCA 7402                       JZ      short nono            ; no
                                ;dbcs_char:                       ;
00000DCC 40                         INC     AX                    ; not YES or NO
                                yesyes:                           ;
00000DCD 40                         INC     AX                    ; return 1
                                    ; 15/12/2022
                                    ; 09/01/2024 - Retro DOS v5.0
                                nono:
                                    ;jmp    short SYS_RET_OK_jmp  ;
                                    ; 06/11/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
                                    ; 09/01/2024
00000DCE E99DF8                     jmp     SYS_RET_OK            ; done

                                capstring:                        ;
00000DD1 89D6                       MOV     SI,DX                 ; si=dx
00000DD3 3C21                       CMP     AL,CAP_STRING  ; 21h  ; cap string ?
00000DD5 750D                       JNZ     short capascii        ; no
                                    ;OR     CX,CX                 ; check count 0
                                    ;JZ     short nono            ; yes finished
                                    ; 06/01/2024
```

```
8046 00000DD7 E3F5                      jcxz    nono
8047                                concap:                         ;
8048 00000DD9 AC                        LODSB                       ; get char
8049 00000DDA E8DF4F                    call    GETLET3             ; upper case it
8050 00000DDD 8844FF                    MOV     byte [SI-1],AL      ; store back
8051                                ;next99:                        ;
8052 00000DE0 E2F7                      LOOP    concap              ; continue
8053 00000DE2 EBEA                      JMP     short nono          ; done
8054                                capascii:                       ;
8055 00000DE4 3C22                      CMP     AL,CAP_ASCIIZ  ; 22h ; cap ASCIIZ string ?
8056 00000DE6 7573                      JNZ     short capinval      ; no
8057                                concap2:                        ;
8058 00000DE8 AC                        LODSB                       ; get char
8059 00000DE9 08C0                      or      al,al               ; end of string ?
8060 00000DEB 74E1                      JZ      short nono          ; yes
8061 00000DED E8CC4F                    call    GETLET3             ; upper case it
8062 00000DF0 8844FF                    MOV     [SI-1],AL           ; store back
8063 00000DF3 EBF3                      JMP     short concap2       ; continue
8064
8065                                    ; MSDOS 3.3 (& MSDOS 6.0)
8066
8067                                ; Offset 1A19h in IBMDOS.COM (MSDOS 3.3), 1987
8068                                ; _$GetExtCntry:
8069
8070                                notcap:
8071 00000DF5 83F905                    CMP     CX,5                ; minimum size is 5
8072                                    ;jb     short sizeerror
8073                                    ; 09/01/2024
8074 00000DF8 7261                      jb      short capinval      ; (size error)
8075
8076                                GEC_CONT:
8077                                ;hkn; SS is DOSDATA
8078                                    ;context DS
8079
8080 00000DFA 16                        push    ss
8081                                    ;pop    es  ; ! (Retro DOS v3.0 BUG) !
8082 00000DFB 1F                        pop     ds  ; 13/05/2019 - Retro DOS v4.0
8083
8084                                ;hkn; COUNTRY_CDPG is in DOSDATA
8085 00000DFC BE[2A12]                  MOV     SI,COUNTRY_CDPG
8086
8087                                ; ----------------------------------------------------------
8088                                    ; 06/01/2024 - Retro DOS v5.0
8089                                    ; PCDOS 7.1 IBMDOS.COM - DOSCODE:4CC6h
8090                                    ;;;
8091 00000DFF 08C0                      or      al,al
8092 00000E01 752A                      jnz     short GETCNTRY
8093                                                ; AL = 0 (INT 21h, AX=6500h)
8094                                                ; Set extended country-dependent information
8095                                                ; (SET GENERAL INTERNATIONALIZATION INFO)
8096
8097 00000E03 83E907                    sub     cx,7            ; minimum 8 bytes
8098 00000E06 7653                      jbe     short capinval ; error_invalid_function
8099 00000E08 8B5C48                    mov     bx,[si+48h]     ; [SI+DOS_CCDPG.ccSysCodePage]
8100 00000E0B 8D7466                    lea     si,[si+66h]     ; SI+DOS_CCDPG.ccCountryInfoLen
8101 00000E0E 8B04                      mov     ax,[si]
8102 00000E10 83E804                    sub     ax,4
8103 00000E13 39C1                      cmp     cx,ax
8104 00000E15 7602                      jbe     short set_intern_inf
8105 00000E17 89C1                      mov     cx,ax
8106                                set_intern_inf:
8107 00000E19 89C8                      mov     ax,cx
8108 00000E1B 83C004                    add     ax,4
8109 00000E1E 26894501                  mov     [es:di+1], ax  ; info length/size (will be written)
8110 00000E22 83C606                    add     si,6            ; DOS_CCDPG.ccDFormat
8111 00000E25 83C707                    add     di,7            ; points to date format
8112 00000E28 E8B209                    call    XCHGP           ; ds:si = user's buffer + 6
8113                                                            ; es:di = country info buffer + 7
8114 00000E2B EB3F                      jmp     short OK_RETN
8115                                    ;;;
8116                                ; ----------------------------------------------------------
8117
8118                                    ; 06/01/2024
8119                                GETCNTRY:
8120 00000E2D 83FAFF                    CMP     DX,-1 ; COUNTRY_ID    ; active country ?
8121 00000E30 7503                      JNZ     short GETCDPG         ; no
8122
8123                                ;hkn; use DS override to accesss country_cdpg fields
8124                                    ;;mov   dx,[si+63h] ; MSDOS 3.3
8125                                    ;mov    dx,[si+68h] ; MSDOS 6.0
8126 00000E32 8B5468                    MOV     DX,[SI+DOS_CCDPG.ccDosCountry]
8127                                                            ; get active country id;smr;use DS
8128                                GETCDPG:
8129 00000E35 83FBFF                    CMP     BX,-1 ; CODE_PAGE     ; active code page?
8130 00000E38 7503                      JNZ     short CHKAGAIN        ; no, check again
8131
8132                                ;hkn; use DS override to accesss country_cdpg fields
8133                                    ;;mov   bx,[si+65h] ; MSDOS 3.3
8134                                    ;mov    bx,[si+6Ah] ; MSDOS 6.0
8135 00000E3A 8B5C6A                    MOV     BX,[SI+DOS_CCDPG.ccDosCodePage]
8136                                                            ; get active code page id;smr;Use DS
8137                                CHKAGAIN:
8138                                    ;cmp    dx,[si+68h] ; MSDOS 6.0
8139 00000E3D 3B5468                    CMP     DX,[SI+DOS_CCDPG.ccDosCountry]
8140                                                            ; same as active country id?;smr;use DS
8141 00000E40 7550                      JNZ     short CHKNLS          ; no
8142                                    ;cmp    bx,[si+6Ah] ; MSDOS 6.0
8143 00000E42 3B5C6A                    CMP     BX,[SI+DOS_CCDPG.ccDosCodePage]
8144                                                            ; same as active code pg id?;smr;use DS
8145 00000E45 754B                      JNZ     short CHKNLS          ; no
8146                                CHKTYPE:
8147                                    ;mov    bx,[si+48h]
8148 00000E47 8B5C48                    MOV     BX,[SI+DOS_CCDPG.ccSysCodePage]
8149                                                            ; bx = sys code page id;smr;use DS
8150 00000E4A 51                        PUSH    CX                    ; save cx
8151                                    ;mov    cx,[si+4Ah]
8152 00000E4B 8B4C4A                    MOV     CX,[SI+DOS_CCDPG.ccNumber_of_entries]  ;smr;use DS
8153                                    ;mov    si,COUNTRY_CDPG+76
8154 00000E4E BE[7612]                  MOV     SI,COUNTRY_CDPG+DOS_CCDPG.ccSetUcase   ;smr;CDPG in DOSDATA
8155                                NXTENTRY:
8156 00000E51 3A04                      CMP     AL,[SI]               ; compare info type;smr;use DS
8157 00000E53 740B                      JZ      short FOUNDIT
8158 00000E55 83C605                    ADD     SI,5                  ; next entry
8159 00000E58 E2F7                      LOOP    NXTENTRY
8160 00000E5A 59                        POP     CX
8161                                capinval:
8162                                    ;error  error_invalid_function; info type not found
8163                                    ;mov    al,1
8164 00000E5B B001                      mov     al,error_invalid_function
8165                                ;SYS_RET_ERR_jmp:
8166                                    ;jmp    SYS_RET_ERR
8167                                    ; 06/11/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
8168                                SYS_RET_ERR_jmp:
8169 00000E5D E918F8                    jmp     SYS_RET_ERR
```

```
                                FOUNDIT:
00000E60 A4                         MOVSB                       ; move info id byte
00000E61 59                         POP     CX                  ; restore char count
                                    ;cmp    al,1
00000E62 3C01                       CMP     AL,SetCountryInfo   ; select country info type ?
00000E64 7415                       JZ      short setsize
00000E66 B90400                     MOV     CX,4                ; 4 bytes will be moved
00000E69 B80500                     MOV     AX,5                ; 5 bytes will be returned in CX
                                OK_RETN:
00000E6C F3A4                       REP     MOVSB               ; copy info
00000E6E 89C1                       MOV     CX,AX               ; CX = actual length returned
00000E70 89D8                       MOV     AX,BX               ; return sys code page in ax
                                GETDONE:
00000E72 E802F6                     call    Get_User_Stack      ; return actual length to user's CX
                                    ;mov    [si+4],cx
00000E75 894C04                     MOV     [SI+user_env.user_CX],CX
                                    ;jmp    SYS_RET_OK
                                    ; 15/12/2022
                                    ; 25/06/2019
00000E78 E9F6F7                     jmp     SYS_RET_OK_clc
                                    ; 06/11/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
                                    ; 15/12/2022
                                ;nono_jmp:
                                    ;jmp    short nono
                                setsize:
00000E7B 83E903                     SUB     CX,3                ; size after length field
00000E7E 390C                       CMP     [SI],CX             ; less than table size ;smr;use ds
00000E80 7302                       JAE     short setsize2      ; no
00000E82 8B0C                       MOV     CX,[SI]             ; truncate to table size ;smr;use ds
                                setsize2:
00000E84 26890D                     MOV     [ES:DI],CX          ; copy actual length to user's buffer
                                    ;ADD    DI,2                ; update index
                                    ;ADD    SI,2
                                    ; 06/01/2024
00000E87 47                         inc     di
00000E88 47                         inc     di
00000E89 46                         inc     si
00000E8A 46                         inc     si
00000E8B 89C8                       MOV     AX,CX
00000E8D 83C003                     ADD     AX,3                ; AX has the actual length
00000E90 EBDA                       JMP     short OK_RETN       ; go move it
                                CHKNLS:
00000E92 30E4                       XOR     AH,AH
                                    ;PUSH   AX                  ; save info type
                                    ;POP    BP                  ; bp = info type
                                    ; 06/01/2024
00000E94 89C5                       mov     bp,ax

                                    ;CallInstall NLSInstall,NLSFUNC,0 ; check if NLSFUNC in memory
00000E96 B80014                     mov     ax,1400h
00000E99 CD2F                       int     2Fh     ; - Multiplex - NLSFUNC.COM - INSTALLATION CHECK
                                                    ; Return: AL = 00h not installed, OK to install
                                                    ; 01h not installed, not OK
                                                    ; FFh installed
00000E9B 3CFF                       CMP     AL,0FFH
00000E9D 7402                       JZ      short NLSNXT        ; in memory

                                sizeerror:
                                    ; 09/01/2024 - Retro DOS v5.0
                                    ;
                                ;   ;error  error_invalid_function
                                ;   ;mov    al,1
                                ;   mov     al,error_invalid_function
                                ;   ;jmp    SYS_RET_ERR
                                ;   ; 06/11/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
                                ;sys_ret_err_jmp2:
                                ;   jmp     short SYS_RET_ERR_jmp
                                    ; 09/01/2024
00000E9F EBBA                       jmp     short capinval

                                NLSNXT:
                                    ;CallInstall GetExtInfo,NLSFUNC,2 ;get extended info
00000EA1 B80214                     mov     ax,1402h
00000EA4 CD2F                       int     2Fh     ; - Multiplex - NLSFUNC.COM - GET COUNTRY INFO
                                                    ; BP = subfunction, BX = code page
                                                    ; DX = country code, DS:SI -> internal code page structure
                                                    ; ES:DI -> user buffer, CX = size of user buffer
                                                    ; Return: AL = status
                                                    ; 00h successful
                                                    ; else DOS error code

00000EA6 3C00                       CMP     AL,0                ; success ?
00000EA8 7505                       JNZ     short NLSERROR
                                    ;mov    ax,[si+48h] ; 13/05/2019
00000EAA 8B4448                     MOV     AX,[SI+DOS_CCDPG.ccSysCodePage]
                                                    ; ax = sys code page id;smr;use ds;
                                                    ;BUGBUG;check whether DS is OK after the above calls
00000EAD EBC3                       JMP     short GETDONE
                                seterr:
                                    ; 15/12/2022
                                NLSERROR:
                                    ;jmp    SYS_RET_ERR         ; return what is got from NLSFUNC
                                    ; 06/11/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
                                    ;jmp    short sys_ret_err_jmp2
                                    ; 15/12/2022
00000EAF EBAC                       jmp     short SYS_RET_ERR_jmp

                                ;EndProc $GetExtCntry

                                ; 13/05/2019 - Retro DOS v4.0
                                ; DOSCODE:4BD6h (MSDOS 6.21, MSDOS.SYS)

                                ;BREAK <$GetSetCdPg - get or set global code page>
                                ;--------------------------------------------------------------------------
                                ;** $GetSetCdPg - Get or Set Global Code Page
                                ;
                                ;   System call format:
                                ;
                                ;   MOV     AH,GetSetCdPg  ; DOS 3.3
                                ;   MOV     AL,n           ; n = 1 : get code page, n = 2 : set code page
                                ;   MOV     BX,CODE_PAGE   (set code page only)
                                ;   INT     21
                                ;
                                ;   ENTRY   (al) = n
                                ;           (bx) = code page
                                ;   EXIT    'C' clear
                                ;             global code page is set    (set global code page)
                                ;             (BX) = active code page id  (get global code page)
                                ;             (DX) = system code page id  (get global code page)
                                ;           'C' set
                                ;             (AX) = error code
                                ;
                                ;procedure  $GetSetCdPg,NEAR   ; DOS 3.3
```

```
                                    ; 06/11/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
                                    ; DOSCODE:4BC9h

                                    ; 06/01/2024 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMDOS.COM)
                                    ; DOSCODE:4D73h

                                _$GetSetCdPg:

                                ;hkn; SS is DOSDATA
                                    ;context DS

00000EB1 16                         push    ss
00000EB2 1F                         pop     ds

                                ;hkn; COUNTRY_CDPG is in DOSDATA
00000EB3 BE[2A12]                   MOV     SI,COUNTRY_CDPG         ; (DOSDATA:122Ah for MSDOS 6.21)

00000EB6 3C01                       CMP     AL,1                    ; get global code page
00000EB8 7512                       JNZ     short setglpg           ; set global code page

                                    ;;mov   bx,[si+65h] ; MSDOS 3.3
                                    ;mov    bx,[si+6Ah] ; MSDOS 6.0
00000EBA 8B5C6A                     MOV     BX,[SI+DOS_CCDPG.ccDosCodePage]
                                                                    ; get active code page id;smr;use ds
                                    ;mov    dx,[si+48h]
00000EBD 8B5448                     MOV     DX,[SI+DOS_CCDPG.ccSysCodePage]
                                                                    ; get sys code page id;smr;use ds
00000EC0 E8B4F5                     call    Get_User_Stack
                                ;ASSUME DS:NOTHING
                                    ;;mov   [si+2],bx
                                    ;MOV    [SI+user_env.user_BX],BX ; update returned bx
                                    ; 06/01/2024 (PCDOS 7.1 IBMDOS.COM)
00000EC3 E8F4FD                     call    set_user_bx    ; MOV [SI+user_env.user_BX],BX
                                    ;mov    [si+6],dx
00000EC6 895406                     MOV     [SI+user_env.user_DX],DX ; update returned dx
                                OK_RETURN:
                                    ; 15/12/2022
                                    ;transfer SYS_RET_OK
00000EC9 E9A2F7                     jmp     SYS_RET_OK
                                    ; 06/11/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
                                    ;jmp    short nono_jmp

                                ;hkn; ASSUME DS:DOSGROUP
                                ;ASSUME     DS:DOSDATA

                                setglpg:
00000ECC 3C02                       CMP     AL,2
00000ECE 752F                       JNZ     short nomem

                                    ;;mov   dx,[si+63h] ; MSDOS 3.3
                                    ;mov    dx,[si+68h] ; MSDOS 6.0
00000ED0 8B5468                     MOV     DX,[SI+DOS_CCDPG.ccDosCountry]      ;smr;use ds

                                    ;CallInstall NLSInstall,NLSFUNC,0 ; check if NLSFUNC in memory
00000ED3 B80014                     mov     ax,1400h
00000ED6 CD2F                       int     2Fh     ; - Multiplex - NLSFUNC.COM - INSTALLATION CHECK
                                                    ; Return: AL = 00h not installed, OK to install
                                                    ; 01h not installed, not OK
                                                    ; FFh installed
00000ED8 3CFF                       CMP     AL,0FFH
00000EDA 7523                       JNZ     short nomem             ; not in memory

                                    ;CallInstall SetCodePage,NLSFUNC,1 ;set the code page
00000EDC B80114                     mov     ax,1401h
00000EDF CD2F                       int     2Fh     ; - Multiplex - NLSFUNC.COM - CHANGE CODE PAGE
                                                    ; DS:SI -> internal code page structure
                                                    ; BX = new code page, DX = country code???
                                                    ; Return: AL = status
                                                    ; 00h successful
                                                    ; else DOS error code
                                    ;cmp    al,0
00000EE1 08C0                       or      al,al                   ; success ?
00000EE3 74E4                       JZ      short OK_RETURN                 ; yes

00000EE5 3C41                       CMP     AL,65                   ; set device code page failed
00000EE7 75C6                       JNZ     short seterr
                                    ;MOV    AX,65
                                    ; 06/01/2024
00000EE9 98                         cbw
00000EEA A3[2403]                   MOV     [EXTERR],AX
                                    ;mov    byte [EXTERR_ACTION],6
                                    ;mov    byte [EXTERR_CLASS],5
                                    ;mov    byte [EXTERR_LOCUS],4
00000EED C606[2603]06               MOV     byte [EXTERR_ACTION],errACT_Ignore
00000EF2 C606[2703]05               MOV     byte [EXTERR_CLASS],errCLASS_HrdFail
00000EF7 C606[2303]04               MOV     byte [EXTERR_LOCUS],errLOC_SerDev
                                    ;transfer From_GetSet
00000EFC E981F7                     jmp     From_GetSet

                                    ; 15/12/2022
                                ;seterr:
                                    ;;;transfer SYS_RET_ERR
                                    ;;jmp   SYS_RET_ERR
                                    ; 06/11/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
                                    ;jmp    short NLSERROR

                                nomem:
                                    ;error  error_invalid_function; function not defined
                                    ;mov    al,1
00000EFF B001                       mov     al,error_invalid_function
00000F01 EBAC                       jmp     short seterr

                                ;EndProc $GetSetCdPg

                                ; 13/05/2019 - Retro DOS v4.0
                                ; DOSCODE:4C2Bh (MSDOS 6.21, MSDOS.SYS)

                                ;BREAK <$Get_Drive_Freespace -- Return bytes of free disk space on a drive>
                                ;--------------------------------------------------------------------------
                                ;** $Get_Drive_Freespace - Return amount of drive free space
                                ;
                                ;   $Get_Drive_Freespace returns the # of free allocation units on a
                                ;           drive.
                                ;
                                ;   This call returns the same info in the same registers (except for the
                                ;   FAT pointer) as the old FAT pointer calls
                                ;
                                ;   ENTRY   DL = Drive number
                                ;   EXIT    AX = Sectors per allocation unit
                                ;              = -1 if bad drive specified
                                ;           On User Stack
                                ;               BX = Number of free allocation units
                                ;               DX = Total Number of allocation units on disk
```

```
                                ;               CX = Sector size
                                ;
                                ;procedure   $GET_DRIVE_FREESPACE,NEAR
                                ;
                                    ; 09/01/2024
                                    ; 06/01/2024 - Retro DOS v5.0
                                    ; (PCDOS 7.1 IBMDOS.COM - DOSCODE:4DC4h)
                                ;
                                _$GET_DRIVE_FREESPACE:
                                ;
                                ;hkn; SS is DOSDATA
                                    ;context DS
00000F03 16                         push    ss
00000F04 1F                         pop     ds
                                ;
00000F05 88D0                       MOV     AL,DL
                                    ;invoke GetThisDrv              ; Get drive
00000F07 E8836C                     call    GETTHISDRV
                                SET_AX_RET:
00000F0A 7220                       JC      short BADFDRV
                                ;
                                    ;invoke DISK_INFO
00000F0C E85424                     call    DISK_INFO
                                    ; 09/01/2024
                                    ;XCHG   DX,BX
                                    ;;JC    short SET_AX_RET        ; User FAILed to I 24
                                    ; 26/06/2024
                                    ; 06/01/2024
                                    ;jc     short BADFDRV
                                    ; 26/06/2024
00000F0F 7304                       jnc     short gdrvfspc_1
00000F11 87D3                       xchg    dx,bx
00000F13 EB17                       jmp     short BADFDRV
                                ;
                                    ; 09/01/2024 - Retro DOS v5.0 (PCDOS 7.1)
                                gdrvfspc_1:
00000F15 30E4                       XOR     AH,AH                   ; Chuck Fat ID byte
                                    ;;;
00000F17 57                         push    di
00000F18 E80E09                     call    TestNet
00000F1B 5F                         pop     di
00000F1C 7203                       jc      short gdrvfspc_2
00000F1E E84625                     call     modify_cluster_count
                                                    ; if hw of total clusters (di) > 0
                                                    ; sectors per cluster and cluster counts
                                                    ; will be modified (shifted)
                                                    ; (but sectors per clust * clust count will be same)
                                                    ; /// disk size -calculation- limit = 2 GB ///
                                gdrvfspc_2:
00000F21 87D3                       xchg    dx,bx           ; bx = free clusters (after xchg)
                                    ;;;
                                DoSt:
00000F23 E851F5                     call    Get_User_Stack
                                ;ASSUME     DS:NOTHING
                                    ;mov    [si+6],dx
                                    ;;mov   [si+4],cx
                                    ;;mov   [si+2],bx
                                    ; 09/01/2024
00000F26 895406                     MOV     [SI+user_env.user_DX],DX ; total clusters
                                    ;MOV    [SI+user_env.user_CX],CX
                                    ;MOV    [SI+user_env.user_BX],BX
                                    ;;MOV   [SI+user_env.user_AX],AX
                                    ;mov    [si],ax
                                    ;;return
                                    ;retn
                                    ; 09/01/2024
00000F29 E989FD                     jmp     gdrvfspc_ret  ; ax = sectors per cluster (modified)
                                ;
                                BADFDRV:
                                    ; MSDOS 3.3
                                    ;;mov   al,0Fh
                                    ;mov    al,error_invalid_drive; Assume error
                                ;
                                    ; 13/05/2019 - Retro DOS v4.0
                                ;
                                    ; MSDOS 6.0 & MSDOS 3.3
                                    ;invoke FCB_RET_ERR
00000F2C E85EF7                     call    FCB_RET_ERR
                                ;
00000F2F B8FFFF                     MOV     AX,-1
00000F32 EBEF                       JMP     short DoSt
                                ;
                                ;EndProc $GET_DRIVE_FREESPACE
                                ;
                                ;   BREAK <$Get_DMA, $Set_DMA -- Get/Set current DMA address>
                                ;----------------------------------------------------------------------------
                                ;** $Get_DMA - Get Disk Transfer Address
                                ;
                                ;   ENTRY   none
                                ;   EXIT    ES:BX is current transfer address
                                ;   USES    all
                                ;
                                    ; 09/01/2024
                                _$GET_DMA:
00000F34 368B1E[2C03]               MOV     BX,[SS:DMAADD]
00000F39 368B0E[2E03]               MOV     CX,[SS:DMAADD+2]
00000F3E E836F5                     call    Get_User_Stack
                                    ;mov    [si+2],bx
                                    ;mov    [si+10h],cx
                                    ; 09/01/2024
                                    ;MOV    [SI+user_env.user_BX],BX
00000F41 894C10                     MOV     [SI+user_env.user_ES],CX
                                    ;retn
                                    ; 09/01/2024
00000F44 E973FD                     jmp     set_user_bx
                                ;
                                ;** $Set_DMA - Set Disk Transfer Address
                                ;----------------------------------------------------------------------------
                                ;   ENTRY   DS:DX is current transfer address
                                ;   EXIT    none
                                ;   USES    all
                                ;
                                _$SET_DMA:
00000F47 368916[2C03]               MOV     [SS:DMAADD],DX
00000F4C 368C1E[2E03]               MOV     [SS:DMAADD+2],DS
00000F51 C3                         retn
                                ;
                                ;   BREAK <$Get_Default_Drive, $Set_Default_Drive -- Set/Get default drive>
                                ;----------------------------------------------------------------------------
                                ;
                                ;** $Get_Default_Drive - Get Current Default Drive
                                ;------------------------------------------------------
                                ;   ENTRY   none
                                ;   EXIT    (AL) = drive number
```

```
                                ;   USES    all
                                
                                _$GET_DEFAULT_DRIVE:
00000F52 36A0[3603]                 MOV     AL,[SS:CURDRV]
00000F56 C3                         retn
                                
                                ;** $Set_Default_Drive - Specify new Default Drive
                                ;-------------------------------------------------
                                ;   ENTRY   (DL) = Drive number for new default drive
                                ;   EXIT    (AL) = Number of drives, NO ERROR RETURN IF DRIVE NUMBER BAD
                                
                                _$SET_DEFAULT_DRIVE:
00000F57 88D0                       MOV     AL,DL
00000F59 FEC0                       INC     AL                      ; A=1, B=2...
00000F5B E8136C                     call    GetVisDrv               ; see if visible drive
00000F5E 7204                       JC      short SETRET            ; errors do not set
00000F60 36A2[3603]                 MOV     [SS:CURDRV],AL          ; no, set
                                
                                SETRET:
00000F64 36A0[4700]                 MOV     AL,[SS:CDSCOUNT]        ; let user see what the count really is
00000F68 C3                         retn
                                
                                ;BREAK <$Get/Set_Interrupt_Vector - Get/Set interrupt vectors>
                                ;-------------------------------------------------------------------------------
                                
                                ;** $Get_Interrupt_Vector - Get Interrupt Vector
                                ;-----------------------------------------------
                                ;   $Get_Interrupt_Vector is the official way for user pgms to get the
                                ;   contents of an interrupt vector.
                                ;
                                ;   ENTRY   (AL) = interrupt number
                                ;   EXIT    (ES:BX) = current interrupt vector
                                
                                _$GET_INTERRUPT_VECTOR:
00000F69 E82E00                     CALL    RECSET
00000F6C 26C41F                     LES     BX,[ES:BX]
00000F6F E805F5                     call    Get_User_Stack
                                set_user_es_bx:
                                    ; 09/01/2024 (PCDOS 7.1 IBMDOS.COM)
                                    ;;mov   [si+2],bx
                                    ;mov    [si+10h],es
                                    ;MOV    [SI+user_env.user_BX],BX
00000F72 8C4410                     MOV     [SI+user_env.user_ES],ES
                                    ;retn
00000F75 E942FD                     jmp     set_user_bx
                                
                                ;** $Set_Interrupt_Vector - Set Interrupt Vector
                                ;-----------------------------------------------
                                ;   $Set_Interrupt_Vector is the official way for user pgms to set the
                                ;   contents of an interrupt vector.
                                ;
                                ;   M004, M068: Also set A20OFF_COUNT to 1 if EXECA20OFF bit has been set
                                ;   and if A20OFF_COUNT is non-zero. See under tag M003 in inc\dossym.inc
                                ;   for explanation.
                                ;
                                ;   ENTRY   (AL) = interrupt number
                                ;           (ds:dx) = desired new vector value
                                ;   EXIT    none
                                ;   USES    all
                                
                                ; 07/12/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
                                ; 13/05/2019 - Retro DOS v4.0
                                
                                _$SET_INTERRUPT_VECTOR:
00000F78 E81F00                     CALL    RECSET
00000F7B FA                         CLI                             ; Watch out!!!!! Folks sometimes use
00000F7C 268917                     MOV     [ES:BX],DX              ;   this for hardware ints (like timer).
00000F7F 268C5F02                   MOV     [ES:BX+2],DS
00000F83 FB                         STI
                                                                    ; M004, M068 - Start
                                    ; MSDOS 6.0
00000F84 36F606[8600]04             test    byte [ss:DOS_FLAG],EXECA20OFF ; 4
                                                                    ; Q: was the previous call an int 21h
                                                                    ;    exec call
                                    ; 07/12/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
                                    ;jnz    short siv_1             ; Y: go set count
                                    ;retn                           ; N: return
                                    ; 15/12/2022
00000F8A 740D                       jz      short siv_2
                                siv_1:
00000F8C 36803E[8500]00             cmp     byte [ss:A20OFF_COUNT],0 ; Q: is count 0
00000F92 7505                       jnz     short siv_2              ; N: done
                                    ; 20/09/2023
00000F94 36FE06[8500]               inc     byte [ss:A20OFF_COUNT]
                                    ;mov    byte [ss:A20OFF_COUNT],1 ; Y: set it to 1 to indicate to dos
                                                                     ; dispatcher to turn A20 Off before
                                                                     ; returning to user.
                                siv_2:
                                    ; 07/12/2022
00000F99 C3                         retn                            ; M004, M068 - End
                                
                                RECSET:
00000F9A 31DB                       XOR     BX,BX
00000F9C 8EC3                       MOV     ES,BX
00000F9E 88C3                       MOV     BL,AL
00000FA0 D1E3                       SHL     BX,1
00000FA2 D1E3                       SHL     BX,1
00000FA4 C3                         retn
                                
                                ;   BREAK <$Char_Oper - hack on paths, switches so that xenix can look like PCDOS>
                                ;-------------------------------------------------------------------------------
                                
                                ;** $Char_Oper - Manipulate Switch Character
                                ;
                                ;   This function was put in to facilitate XENIX path/switch compatibility
                                ;
                                ;   ENTRY   AL = function:
                                ;               0 - read switch char
                                ;               1 - set switch char (char in DL)
                                ;               2 - read device availability
                                ;                  Always returns available
                                ;               3 - set device availability
                                ;                  No longer supported (NOP)
                                ;   EXIT    (al) = 0xff iff error
                                ;           (al) != 0xff if ok
                                ;             (dl) = character/flag, if "read switch char" subfunction
                                ;   USES    AL, DL
                                ;
                                ;   NOTE    This already obsolete function has been deactivated in DOS 5.0
                                ;           The character / is always returned for subfunction 0,
                                ;           subfunction 2 always returns -1, all other subfunctions are ignored.
                                
                                ; 13/05/2019 - Retro DOS v4.0
                                ; DOSCODE:4CC9h (MSDOS 6.21, MSDOS.SYS)
```

```
                                ; 06/11/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
                                ; DOSCODE:4CBCh (MSDOS 5.0, MSDOS.SYS)

                                _$CHAR_OPER:
                                    ; MSDOS 6.0
00000FA5 08C0                       or      al,al                       ; get switch?
00000FA7 B22F                       mov     dl,'/'                      ; assume yes
00000FA9 7407                       jz      short chop_1                ; jump if yes
00000FAB 3C02                       cmp     al,2                        ; check device availability?
00000FAD B2FF                       mov     dl,-1                       ; assume yes
00000FAF 7401                       jz      short chop_1                ; jump if yes
00000FB1 C3                         retn                                ; otherwise just quit

                                ; subfunctions requiring return of value to user come here. DL holds
                                ; value to return

                                chop_1:
00000FB2 E8C2F4                     call    Get_User_Stack
00000FB5 895406                     mov     [SI+user_env.user_DX],dx    ; store value for user
00000FB8 C3                         retn

                                    ; MSDOS 3.3
                                    ; Offset 1B87h in IBMDOS.COM (MSDOS 3.3), 1987
                                    ;push   ss
                                    ;pop    ds
                                    ;cmp    al,1
                                    ;jb     short chop_1
                                    ;jz     short chop_2
                                    ;cmp    al,3
                                    ;jb     short chop_3
                                    ;jz     short chop_5
                                    ;mov    al,0FFh
                                    ;retn
                                ;chop_1:
                                    ;mov    dl,[chSwitch]
                                    ;jmp    short chop_4
                                ;chop_2:
                                    ;mov    [chSwitch],dl
                                    ;retn
                                ;chop_3:
                                    ;mov    dl, FFh
                                ;chop_4:
                                    ;call   Get_User_Stack
                                    ;mov    [si+6],dx
                                ;chop_5:
                                    ;retn

                                ;** $GetExtendedError - Return Extended error code
                                ;-----------------------------------------------------------------------
                                ;   This function reads up the extended error info from the static
                                ;   variables where it was stored.
                                ;
                                ;   ENTRY   none
                                ;   EXIT    AX = Extended error code (0 means no extended error)
                                ;           BL = recommended action
                                ;           BH = class of error
                                ;           CH = locus of error
                                ;           ES:DI = may be pointer
                                ;   USES    ALL

                                    ; 06/11/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)

                                _$GetExtendedError:
00000FB9 16                         push    ss
00000FBA 1F                         pop     ds
00000FBB A1[2403]                   MOV     AX,[EXTERR]
00000FBE C43E[2803]                 LES     DI,[EXTERRPT]
00000FC2 8B1E[2603]                 MOV     BX,[EXTERR_ACTION]      ; BL = Action, BH = Class
00000FC6 8A2E[2303]                 MOV     CH,[EXTERR_LOCUS]
00000FCA E8AAF4                     call    Get_User_Stack
                                    ;mov    [si+0Ah],di
00000FCD 897C0A                     MOV     [SI+user_env.user_DI],DI

                                    ; 09/01/2024 (PCDOS 7.1 IBMDOS.COM)
                                    ;;mov   [si+10h],es
                                    ;MOV    [SI+user_env.user_ES],ES
                                    ;;mov   [si+2],bx
                                    ;MOV    [SI+user_env.user_BX],BX
00000FD0 E89FFF                     call    set_user_es_bx

                                    ;mov    [si+4],cx
00000FD3 894C04                     MOV     [SI+user_env.user_CX],CX
                                jmp_SYS_RET_OK:
                                    ; 15/12/2022
                                    ;jmp    SYS_RET_OK
                                    ; 25/06/2019
00000FD6 E998F6                     jmp     SYS_RET_OK_clc ; 15/12/2022
                                    ; 06/11/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
                                ;jmp_SYS_RET_OK:
                                    ;jmp    SYS_RET_OK

                                ; ---------------------------------------------------------------------------
                                ; 09/01/2024
                                %if 0
                                    ; 06/11/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
                                    ; DOSCODE:4CF3h
                                ;patch_or_unknown:
                                ;get_code_page:
                                    push    si
                                    mov     si, COUNTRY_CDPG
                                    ;mov    ax, [si+DOS_CCDPG.ccDosCodePage]
                                    mov     ax, [ss:si+6Ah]
                                    pop     si
                                    retn
                                %endif
                                ; ---------------------------------------------------------------------------

                                ; 29/04/2019 - Retro DOS v4.0

                                ;BREAK      <ECS_call - Extended Code System support function>
                                ;-----------------------------------------------------------------------
                                ; Inputs:
                                ;   AL = 0  get lead byte table
                                ;           on return DS:SI has the table location
                                ;
                                ;   AL = 1  set / reset interim console flag
                                ;           DL = flag (00H or 01H)
                                ;           no return
                                ;
                                ;   AL = 2  get interim console flag
                                ;           on return DL = current flag value
                                ;
                                ;   AL = OTHER then error, and returns with:
```

```
8790                                    ;           AX = error_invalid_function
8791                                    ;
8792                                    ;  NOTE: THIS CALL DOES GUARANTEE THAT REGISTER OTHER THAN
8793                                    ;    SS:SP WILL BE PRESERVED!
8794                                    ;--------------------------------------------------------------------------
8795
8796                                    _$ECS_Call:
8797 00000FD9 08C0                         or      al,al                   ; AL = 0 (get table)?
8798                                        ;jnz    short _okok
8799                                        ; 15/12/2022
8800 00000FDB 7403                         jz      short get_lbt
8801                                    ;_okok:
8802 00000FDD E98EF6                       jmp     SYS_RET_OK
8803                                    get_lbt:
8804 00000FE0 E894F4                       call    Get_User_Stack          ; *
8805
8806                                    ;hkn; dbcs_table moved low to dosdata
8807                                        ;mov    word [si+8],DBCS_TAB+2
8808 00000FE3 C74408[020D]                 mov     word [SI+user_env.user_SI],DBCS_TAB+2
8809
8810 00000FE8 06                           push    es
8811                                        ;getdseg <es>                   ; es = DOSDATA
8812 00000FE9 2E8E06[0700]                 mov     es,[cs:DosDSeg]
8813                                        ;mov    [si+14],es
8814 00000FEE 8C440E                       mov     [SI+user_env.user_DS],es
8815 00000FF1 07                           pop     es
8816
8817                                        ; 15/12/2022
8818 00000FF2 EBE2                         jmp     short jmp_SYS_RET_OK ; jmp SYS_RET_OK_clc ; *
8819                                    ;_okok:
8820                                        ; 15/12/2022
8821                                        ;;transfer SYS_RET_OK
8822                                        ;jmp    short jmp_SYS_RET_OK
8823                                        ; 06/11/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
8824                                        ;;jmp   SYS_RET_OK
8825                                        ;jmp    short jmp_SYS_RET_OK
8826
8827                                    ; --------------------------------------------------------------------------
8828                                    ; 10/01/2024 - Retro DOS v5.0 (PCDOS 7.1 IBMDOS.COM - DOSCODE:4EB4h)
8829                                    ; --------------------------------------------------------------------------
8830
8831                                        ; INT 21h, AH=71h - LONG FILENAME FUNCTIONS
8832                                        ; INT 21h, AH=72h - LFN FindClose
8833                                    _$LONGNAME:
8834 00000FF4 30C0                         xor     al,al                   ; longname functions are not supported
8835                                    lfn_error:
8836 00000FF6 E87EF4                       call    Get_User_Stack
8837 00000FF9 834C1601                     or      word [si+16h],1                 ; [SI+user_env.user_F],f_Carry
8838 00000FFD F9                           stc
8839 00000FFE 8904                         mov     [si],ax                 ; [SI+user_env.user_ax]
8840 00001000 C3                           retn
8841                                    ; --------------------------------------------------------------------------
8842
8843                                        ; FAT32 - EXTENDED FUNCTIONS
8844                                        ; INT 21h, AH=73h
8845                                    _$FAT32EXT:
8846 00001001 3C05                         cmp     al,5                    ; INT 21h AX = 7305h
8847 00001003 7609                         jbe     short valid_fat32_ext_function
8848 00001005 B001                         mov     al,1                    ; error_invalid_function
8849                                    fat32_ext_func_err:
8850 00001007 E96EF6                       jmp     SYS_RET_ERR
8851
8852                                    function_5_invalid_cx:
8853 0000100A B057                         mov     al,57h                  ; error_invalid_parameter
8854                                    fat32_ext_func_err_j:
8855 0000100C EBF9                         jmp     short fat32_ext_func_err
8856
8857                                    valid_fat32_ext_function:
8858 0000100E 7524                         jnz     short not_function_5
8859 00001010 83F9FF                       cmp     cx,0FFFFh       ; Function 5 - FAT32 - EXTENDED ABSOLUTE DISK READ/WRITE
8860 00001013 75F5                         jne     short function_5_invalid_cx
8861 00001015 F7C6FE9F                     test    si,9FFEh                ; read/write mode flags
8862 00001019 75EF                         jnz     short function_5_invalid_cx
8863 0000101B 88D0                         mov     al,dl                   ; drive number, 1 = A
8864 0000101D FEC8                         dec     al
8865 0000101F B401                         mov     ah,1
8866 00001021 F7C60100                     test    si,1
8867 00001025 7405                         jz      short function_5_read
8868 00001027 E8BAF5                       call    FAT32_ABSDWRT           ; INT 21h AX = 7305h (SI bit 0 = 1)
8869 0000102A EB03                         jmp     short fat32_absdrw_ret
8870
8871                                    function_5_read:
8872 0000102C E802F5                       call    FAT32_ABSDRD            ; INT 21h AX = 7305h (SI bit 0 = 0)
8873                                    fat32_absdrw_ret:
8874 0000102F 72C5                         jc      short lfn_error
8875 00001031 E93AF6                       jmp     SYS_RET_OK
8876
8877                                    not_function_5:
8878 00001034 3C03                         cmp     al,3            ; Function 3 - FAT32 - GET EXTENDED FREE SPACE ON DRIVE
8879 00001036 7475                         je      short function_73_3
8880 00001038 3C02                         cmp     al,2
8881 0000103A 7203                         jb      short chk_drive_lock_flush
8882 0000103C E90503                       jmp     _$GET_DPB       ; Function 2 - FAT32 - "Get_ExtDPB" - GET EXTENDED DPB
8883                                                               ; Function 4 - FAT32 - Set DPB TO USE FOR FORMATTING
8884
8885                                    chk_drive_lock_flush:
8886 0000103F 80FA1A                       cmp     dl,26                   ; MSDOS 7 - DRIVE LOCKING AND FLUSHING
8887 00001042 7604                         jbe     short drv_lock_flush_1
8888 00001044 B00F                         mov     al,0Fh                  ; invalid drive number
8889                                    drv_lock_flush_err:
8890                                        ;jmp    short fat32_ext_func_err_j ; ax = error code
8891                                        ; 10/01/2024
8892 00001046 EBBF                         jmp     short fat32_ext_func_err
8893
8894                                    drv_lock_flush_1:
8895 00001048 FECA                         dec     dl
8896 0000104A 7905                         jns     short drv_lock_flush_2
8897 0000104C 368A16[3603]                 mov     dl,[ss:CURDRV]          ; 0 = default/current drive)
8898                                    drv_lock_flush_2:
8899 00001051 B600                         mov     dh,0
8900 00001053 89D3                         mov     bx,dx
8901 00001055 80F901                       cmp     cl,1                    ; which flag to get or set
8902 00001058 7604                         jbe     short drv_lock_flush_3
8903 0000105A B001                         mov     al,1                    ; error_invalid_function
8904                                        ;jmp    short drv_lock_flush_err
8905                                        ; 10/01/2024
8906 0000105C EBA9                         jmp     short fat32_ext_func_err
8907
8908                                    drv_lock_flush_3:
8909 0000105E 368AA7[0813]                 mov     ah,[ss:drive_flags+bx]
8910 00001063 08C9                         or      cl,cl
8911 00001065 7422                         jz      short get_set_indctd_flag ; use bit 1 and bit 2
8912 00001067 08C0                         or      al,al                   ; get drive's dirty-buffers flag
8913 00001069 7419                         jz      short get_dirty_buf_flag ; use bit 3
```

```
8914 0000106B 80E4F7                    and     ah,0F7h                 ; clear bit 3
8915 0000106E 80E508                    and     ch,8                    ; izolate bit 3 of the new flag value
8916 00001071 08EC                      or      ah,ch                   ; set AH bit 3 according to CH bit 3
8917 00001073 3688A7[0813]              mov     [ss:drive_flags+bx],ah; set or reset dirty buffer flag
8918 00001078 F6C408                    test    ah,8
8919 0000107B 7505                      jnz     short set_dirty_flag_ok       ; bit 3 is set/1
8920 0000107D B0FF                      mov     al,0FFh
8921 0000107F E8165A                    call    FLUSHBUF
8922                                  set_dirty_flag_ok:
8923 00001082 EB26                      jmp     short jmp_to_SYS_RET_OK
8924
8925                                  get_dirty_buf_flag:
8926 00001084 80E408                    and     ah,8                    ; izolate dirty buffers flag
8927 00001087 EB19                      jmp     short mov_flag_cl_to_al       ; AH = new flag and 08h (bit 3 used)
8928
8929                                  get_set_indctd_flag:
8930 00001089 08C0                      or      al,al
8931 0000108B 7412                      jz      short get_indicated_flag
8932 0000108D 80E4F9                    and     ah,0F9h                 ; clear bit 1 and bit 2
8933 00001090 F6C502                    test    ch,2                    ; new value for indicated flag
8934 00001093 7403                      jz      short reset_indctd_flags ; bit 1 is zero
8935 00001095 80CC06                    or      ah,6                    ; set bit 1 and bit 2
8936                                  reset_indctd_flags:
8937 00001098 3688A7[0813]              mov     [ss:drive_flags+bx],ah
8938 0000109D EB0B                      jmp     short jmp_to_SYS_RET_OK
8939
8940                                  get_indicated_flag:
8941 0000109F 80E406                    and     ah,6                    ; AH = new flag and 06h (bits 1 and 2 used)
8942                                  mov_flag_cl_to_al:                ; CODE XREF: DOSCODE:4F47^j
8943 000010A2 88C8                      mov     al,cl                   ; value of CL on entry
8944                                  ret_ax_to_user_cx:  ; ax -> user's cx
8945 000010A4 E8D0F3                    call    Get_User_Stack
8946 000010A7 894404                    mov     [si+4],ax               ; [SI+user_env.user_cx] ; requested flag
8947                                  jmp_to_SYS_RET_OK:
8948 000010AA E9C1F5                    jmp     SYS_RET_OK
8949
8950                                  function_73_3:
8951 000010AD 89D6                      mov     si,dx           ; FAT32 - GET EXTENDED FREE SPACE ON DRIVE
8952                                                             ; AX = 7303h
8953                                                             ; DS:DX -> ASCIZ string for drive ("C:\" or "\\SERVER\Share")
8954                                                             ; ES:DI -> buffer for extended free space structure
8955                                                             ; CX = length of buffer for extended free space
8956 000010AF E8D76E                    call    DriveFromText
8957 000010B2 92                        xchg    ax,dx
8958 000010B3 AD                        lodsw
8959 000010B4 FECA                      dec     dl
8960 000010B6 80FA1A                    cmp     dl,26
8961 000010B9 7323                      jnb     short func_73_3_err2
8962 000010BB 08E4                      or      ah,ah
8963 000010BD 751F                      jnz     short func_73_3_err2
8964 000010BF E8284D                    call    PATHCHRCMP
8965 000010C2 751A                      jnz     short func_73_3_err2
8966 000010C4 FEC2                      inc     dl
8967 000010C6 83F92C                    cmp     cx,44                   ; buffer (Structure) size must be 44
8968 000010C9 721C                      jb      short func_73_3_err4
8969 000010CB 26837D0200                cmp     word [es:di+2],0        ; buffer structure version (must be 0)
8970 000010D0 7511                      jnz     short func_73_3_err3
8971 000010D2 16                        push    ss
8972 000010D3 1F                        pop     ds
8973 000010D4 88D0                      mov     al,dl
8974 000010D6 E8B46A                    call    GETTHISDRV
8975 000010D9 7310                      jnc     short fill_efs_struc_b
8976                                  func_73_3_err1:
8977 000010DB E8AFF5                    call    FCB_RET_ERR
8978                                  func_73_3_err2:
8979 000010DE B00F                      mov     al,0Fh                  ; error_invalid_drive
8980                                  jmp_to_SYS_RET_ERR:
8981 000010E0 E995F5                    jmp     SYS_RET_ERR
8982
8983                                  func_73_3_err3:
8984 000010E3 B057                      mov     al,57h                  ; error_invalid_parameter
8985                                  jmp_to_jmp_SYS_RET_ERR:
8986 000010E5 EBF9                      jmp     short jmp_to_SYS_RET_ERR
8987
8988                                  func_73_3_err4:
8989 000010E7 B018                      mov     al,18h                  ; error_bad_length
8990 000010E9 EBFA                      jmp     short jmp_to_jmp_SYS_RET_ERR
8991
8992                                  fill_efs_struc_b:
8993 000010EB E87522                    call    DISK_INFO
8994 000010EE 72EB                      jc      short func_73_3_err1
8995 000010F0 30E4                      xor     ah,ah
8996 000010F2 56                        push    si                      ; si:dx = free cluster count
8997 000010F3 57                        push    di                      ; di:bx = number of clusters
8998 000010F4 E880F3                    call    Get_User_Stack
8999 000010F7 8E4410                    mov     es,[si+10h]             ; user's buffer segment (in ES)
9000 000010FA 8B7C0A                    mov     di,[si+0Ah]             ; user's buffer offset/address (in DI)
9001                                    ; 10/01/2024
9002 000010FD 06                        push    es
9003 000010FE 1F                        pop     ds ; (*)
9004                                    ;
9005                                    ;mov    [es:di+10h],bx          ; total number of clusters on the drive
9006                                    ;mov    [es:di+20h],bx          ; total allocation units, without adjustment for compression
9007 000010FF 895D10                    mov     [di+10h],bx
9008 00001102 895D20                    mov     [di+20h],bx
9009 00001105 5B                        pop     bx
9010                                    ;mov    [es:di+12h],bx          ; total number of clusters on the drive, hw
9011                                    ;mov    [es:di+22h],bx          ; total allocation units, hw
9012                                    ;mov    [es:di+0Ch],dx          ; number of available clusters
9013                                    ;mov    [es:di+1Ch],dx          ; number of available allocation units, without adjustment
9014 00001106 895D12                    mov     [di+12h],bx
9015 00001109 895D22                    mov     [di+22h],bx
9016 0000110C 89550C                    mov     [di+0Ch],dx
9017 0000110F 89551C                    mov     [di+1Ch],dx
9018 00001112 5A                        pop     dx
9019                                    ;mov    [es:di+0Eh],dx          ; number of available clusters, hw
9020                                    ;mov    [es:di+1Eh],dx          ; number of available allocation units, hw
9021                                    ;mov    [es:di+8],cx            ; bytes per sector
9022                                    ;mov    [es:di+4],ax            ; sectors per cluster (with adjustment for compression)
9023 00001113 89550E                    mov     [di+0Eh],dx
9024 00001116 89551E                    mov     [di+1Eh],dx
9025 00001119 894D08                    mov     [di+8],cx
9026 0000111C 894504                    mov     [di+4],ax
9027 0000111F 89C1                      mov     cx,ax
9028 00001121 F7E3                      mul     bx                      ; 32 bit multiplication
9029 00001123 720B                      jc      short dsk_cap_calc_overf ; disk capacity calculation overflow error
9030 00001125 89C3                      mov     bx,ax
9031                                    ;mov    ax,[es:di+20h]          ; total allocation units, lw
9032 00001127 8B4520                    mov     ax,[di+20h]
9033 0000112A F7E1                      mul     cx
9034 0000112C 01DA                      add     dx,bx
9035 0000112E 7305                      jnb     short dsk_cap_calc_ok
9036                                  dsk_cap_calc_overf:
9037 00001130 B8FFFF                    mov     ax,0FFFFh               ; set to 0FFFFFFFFh
```

```
9038 00001133 89C2                      mov     dx,ax
9039                                dsk_cap_calc_ok:
9040                                    ; 10/01/2024
9041                                    ; ds = es (*)
9042                                    ;mov    [es:di+1Ah],dx
9043                                    ;mov    [es:di+18h],ax          ; total number of physical sectors on the drive,
9044                                                                    ; without adjustment for compression
9045 00001135 89551A                    mov     [di+1Ah],dx
9046 00001138 894518                    mov     [di+18h],ax
9047 0000113B 89C8                      mov     ax,cx                   ; 32 bit multiplication
9048                                    ;mul    word [es:di+0Eh]        ; number of available clusters, hw
9049 0000113D F7650E                    mul     word [di+0Eh]
9050 00001140 720B                      jc      short dsk_free_calc_overf
9051 00001142 89C3                      mov     bx,ax
9052                                    ;mov    ax,[es:di+0Ch]          ; number of available clusters, lw
9053 00001144 8B450C                    mov     ax,[di+0Ch]
9054 00001147 F7E1                      mul     cx
9055 00001149 01DA                      add     dx,bx
9056 0000114B 7305                      jnc     short dsk_free_calc_ok
9057                                dsk_free_calc_overf:
9058 0000114D B8FFFF                    mov     ax,0FFFFh
9059 00001150 89C2                      mov     dx,ax
9060                                dsk_free_calc_ok:
9061                                    ; 10/01/2024
9062                                    ;mov    [es:di+16h],dx          ; hw
9063                                    ;mov    [es:di+14h],ax          ; number of physical sectors available on the drive,
9064                                                                    ; without adjustment for compression
9065 00001152 895516                    mov     [di+16h],dx
9066 00001155 894514                    mov     [di+14h],ax
9067 00001158 31C0                      xor     ax,ax  ; 0
9068                                    ;mov    [es:di+0Ah],ax          ; number of bytes per sector, high word = 0
9069                                    ;mov    [es:di+6],ax            ; number of sectors per cluster, high word = 0
9070 0000115A 89450A                    mov     [di+0Ah],ax
9071 0000115D 894506                    mov     [di+6],ax
9072
9073                                    ;mov    [es:di+24h],ax          ; reserved, 8 bytes zero
9074                                    ;mov    [es:di+26h],ax
9075                                    ;mov    [es:di+28h],ax
9076                                    ;mov    [es:di+2Ah],ax
9077 00001160 83C724                    add     di,24h ; 36 ; 10/01/2024
9078 00001163 AB                        stosw
9079 00001164 AB                        stosw
9080 00001165 AB                        stosw
9081 00001166 AB                        stosw
9082 00001167 B82C00                    mov     ax,44
9083 0000116A 29C7                      sub     di,ax ; 10/01/2024
9084                                    ;mov    [es:di],ax              ; size of returned structure = 44
9085                                    ; 10/01/2024
9086                                    ;mov    [di],ax
9087 0000116C AB                        stosw
9088 0000116D E9FEF4                    jmp     SYS_RET_OK
9089
9090                                ; ---------------------------------------------------------------------------
9091
9092                                ; 12/01/2024 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMDOS.COM/MSDOS.SYS)
9093                                ; PCDOS 7.1 IBMDOS.COM - DOSCODE:504Ah
9094
9095                                Set_DPBforFormat:  ; INT 21h, AX = 7304h (continues from _$GET_DPB)
9096                                    ; 12/01/2024                    ; (Ref: Ralf Brown's Interrupt List)
9097 00001170 B81800                    mov     ax,18h
9098 00001173 394404                    cmp     [si+4],ax
9099                                    ;cmp    word ptr [si+4],24      ; [SI+user_env.user_CX]
9100                                                                    ; size of buffer (must be at least 18h)
9101                                    ;jnb    short setdpbf_2
9102                                    ;mov    al,18h                  ; error_bad_length
9103                                ;setdpbf_1:
9104                                    ;jmp    SYS_RET_ERR
9105 00001176 7223                      jb      short setdpbf_1 ; al = 18h
9106                                setdpbf_2:
9107 00001178 06                        push    es                      ; ES:BP = Drive parameter block
9108 00001179 55                        push    bp
9109 0000117A 8E4410                    mov     es,[si+10h]             ; [SI+user_env.user_ES]
9110 0000117D 8B7C0A                    mov     di,[si+0Ah]             ; [SI+user_env.user_DI]
9111 00001180 5E                        pop     si
9112 00001181 1F                        pop     ds
9113
9114 00001182 268B4504                  mov     ax,[es:di+4]            ; (call) function number
9115 00001186 26837D0200                cmp     word [es:di+2],0        ; structure version (must be 0)
9116 0000118B 750C                      jnz     short setdpbf_3
9117 0000118D 26837D0600                cmp     word [es:di+6],0        ; (must be 0)
9118 00001192 7505                      jnz     short setdpbf_3
9119 00001194 83F804                    cmp     ax,4                    ; (max) 5 functions (0 to 4)
9120 00001197 7605                      jbe     short setdpbf_4
9121                                setdpbf_3:
9122 00001199 B057                      mov     al,57h                  ; error_invalid_parameter
9123                                    ;jmp    short setdpbf_1
9124                                setdpbf_1:
9125 0000119B E9DAF4                    jmp     SYS_RET_ERR
9126
9127                                setdpbf_4:
9128 0000119E 26C7051800                mov     word [es:di],18h        ; (call) size
9129 000011A3 08C0                      or      al,al
9130 000011A5 7403                      jz      short setdpbf_5                 ; invalidate DPB counts
9131 000011A7 E98200                    jmp     setdpbf_18
9132
9133                                setdpbf_5:
9134 000011AA 31D2                      xor     dx,dx  ; 0
9135 000011AC 8B5C0D                    mov     bx,[si+0Dh]             ; DPB.MAX_CLUSTER
9136 000011AF 39540F                    cmp     [si+0Fh],dx ; 0
9137                                    ;cmp    word [si+0Fh],0                 ; DPB.FAT_SIZE
9138 000011B2 7506                      jnz     short setdpbf_6                 ; not FAT32
9139 000011B4 8B542F                    mov     dx,[si+2Fh]
9140 000011B7 8B5C2D                    mov     bx,[si+2Dh]             ; DPB.LAST_CLUSTER
9141                                setdpbf_6:
9142 000011BA 268B450A                  mov     ax,[es:di+0Ah]
9143 000011BE 268B4D08                  mov     cx,[es:di+8]            ; new DPB free count
9144                                                                ; (00000000h=no change, FFFFFFFFh=unknown)
9145 000011C2 09C0                      or      ax,ax
9146 000011C4 7502                      jnz     short setdpbf_7
9147 000011C6 E322                      jcxz    setdpbf_11
9148                                setdpbf_7:
9149 000011C8 83F8FF                    cmp     ax,0FFFFh
9150 000011CB 7505                      jne     short setdpbf_8
9151 000011CD 83F9FF                    cmp     cx,0FFFFh
9152 000011D0 7408                      je      short setdpbf_10        ; (set as UNKNOWN/INITIAL)
9153                                setdpbf_8:
9154 000011D2 39D0                      cmp     ax,dx                   ; must be < DPB.LAST_CLUSTER
9155 000011D4 7502                      jne     short setdpbf_9
9156 000011D6 39D9                      cmp     cx,bx
9157                                setdpbf_9:
9158 000011D8 73BF                      jnb     short setdpbf_3
9159                                setdpbf_10:
9160 000011DA 804C1801                  or      byte [si+18h],1                 ; DPB.FIRST_ACCESS (bit 0 = 1)
9161 000011DE 894C1F                    mov     [si+1Fh],cx             ; DPB.FREE_CNT
```

```
                                    cmp     word [si+0Fh],0             ; DP.FAT_SIZE
                                    jnz     short setdpbf_11        ; FAT12 or FAT16
                                    mov     [si+21h],ax             ; DPB.FREE_CNT_HW
                                setdpbf_11:
                                    mov     ax,[es:di+0Eh]
                                    mov     cx,[es:di+0Ch]          ; new DPB next-free
                                                                ; (00000000h=no change, FFFFFFFFh=unknown)
                                    or      ax,ax
                                    jnz     short setdpbf_12
                                    jcxz    setdpbf_17
                                setdpbf_12:
                                    cmp     ax,0FFFFh
                                    jne     short setdpbf_13
                                    cmp     cx,0FFFFh
                                    je      short setdpbf_16        ; (set as UNKNOWN/INITIAL)
                                setdpbf_13:
                                    and     ax,ax
                                    ;cmp    ax,0                    ; must be >= 2
                                    jnz     short setdpbf_14
                                    cmp     cx,2
                                ;setdpbf_14:
                                    jb      short setdpbf_3
                                setdpbf_14:
                                    cmp     ax,dx                   ; must be < DPB.LAST_CLUSTER
                                    jne     short setdpbf_15
                                    cmp     cx,bx
                                setdpbf_15:
                                    ja      short setdpbf_3
                                setdpbf_16:
                                    or      byte [si+18h],1             ; DPB.FIRST_ACCESS (bit 0 = 1)
                                    mov     [si+1Dh],cx             ; DPB.NEXT_FREE
                                    cmp     word [si+0Fh],0             ; DPB.FAT_SIZE
                                    jnz     short setdpbf_17        ; FAT16 or FAT12
                                    mov     [si+3Bh],ax
                                    mov     [si+39h],cx             ; DPB.FAT32_NXTFREE
                                setdpbf_17:
                                    mov     ax,7304h                ; done (successful)
                                    jmp     SYS_RET_OK

                                setdpbf_18:
                                    ;dec    al
                                    dec     ax
                                    jnz     short setdpbf_19
                                    push    ds                      ; rebuild DPB from BPB
                                    push    si
                                    lds     si,[es:di+8]            ; BIOS Parameter Block
                                    pop     bp
                                    pop     es
                                    mov     cx,4558h                ; 'XE' (NASM syntax)
                                    mov     dx,4152h                ; 'RA' (NASM syntax)
                                    call    _$SETDPB
                                    jmp     SYS_RET_OK

                                setdpbf_19:
                                    ;dec    al
                                    dec     ax
                                    jnz     short setdpbf_20
                                              ; force media change
                                              ; (next access to drive rebuild DPB)
                                    or      byte [si+18h],80h       ; DPB.FIRST_ACCESS (bit 7 = 1)
                                setdpbf_29: ; 12/01/2024
                                    jmp     SYS_RET_OK

                                setdpbf_20:
                                    cmp     word [si+0Fh],0             ; DPB.FAT_SIZE
                                    jz      short setdpbf_22 ; FAT32
                                    mov     al,0Fh                  ; error_invalid_drive
                                     ; (function 3 or 4 are only for drives with FAT32 fs)
                                setdpbf_21:
                                    jmp     SYS_RET_ERR

                                setdpbf_22:
                                    ;dec    al
                                    dec     ax
                                    jz      short setdpbf_30        ; get/set active FAT number and mirroring
                                    mov     ax,[si+35h]             ; DPB.ROOT_CLUSTER
                                                ; get/set root directory cluster number
                                    mov     [es:di+0Ch],ax          ; (ret) previous root directory cluster number
                                    mov     ax,[si+37h]
                                    mov     [es:di+0Eh],ax
                                    mov     cx,[es:di+0Ah]
                                    mov     ax,[es:di+8]            ; (call) new root directory cluster number
                                    cmp     ax,0FFFFh               ; -1 --> return only previous root dir cluster number
                                    jne     short setdpbf_23
                                    cmp     cx,0FFFFh
                                    je      short setdpbf_29
                                setdpbf_23:
                                    and     cx,cx
                                    ;cmp    cx,0                    ; cluster number must be >= 2
                                    ;jnz    short setdpbf_24
                                    jnz     short setdpbf_26
                                    cmp     ax,2
                                setdpbf_24:
                                    jnb     short setdpbf_26
                                setdpbf_25:
                                    ;jmp    setdpbf_3               ; error (invalid parameter)
                                    ; 12/01/2024
                                    mov     al,57h                  ; error_invalid_parameter
                                    jmp     short setdpbf_21

                                setdpbf_26:
                                    cmp     cx,[si+2Fh]             ; must be <= DPB.LAST_CLUSTER
                                    jne     short setdpbf_27
                                    cmp     ax,[si+2Dh]
                                setdpbf_27:
                                    ja      short setdpbf_25        ; error
                                    mov     [si+37h],cx             ; DPB.ROOT_CLUSTER
                                    mov     [si+35h],ax
                                    or      byte [si+18h],2             ; DPB.FIRST_ACCESS (bit 1 = 1)
                                    push    ds
                                    push    si
                                    pop     bp
                                    pop     es
                                    call    ECritDisk
                                    call    update_fat32_fsinfo
                                    call    LCritDisk
                                    jnc     short setdpbf_29
                                    mov     al,1Fh                  ; error_gen_failure
                                setdpbf_28:
                                    jmp     short setdpbf_21

                                    ; 12/01/2024
                                ;setdpbf_29:
                                    ;jmp    SYS_RET_OK
```

```
9286
9287                                setdpbf_30:
9288                                    ; 12/01/2024
9289                                    ; ax = 0
9290 000012AF 8164238F00                and     word [si+23h],8Fh     ; DPB.EXT_FLAGS (clear bit 4-6)
9291 000012B4 8B4C23                    mov     cx,[si+23h]
9292 000012B7 26894D0C                  mov     [es:di+0Ch],cx        ; (ret) previous active FAT/mirroring state
9293 000012BB 2689450E                  mov     [es:di+0Eh],ax ; 0
9294                                    ;mov    word [es:di+0Eh],0    ; put zero to Set_DPBforFormat struc offset 14
9295 000012BF 268B5508                  mov     dx,[es:di+8]          ; (call) new active FAT/mirroring state,
9296                                                                  ;  or FFFFFFFFh to get
9297 000012C3 83FAFF                    cmp     dx,0FFFFh
9298                                    ;jnz    short setdpbf_31
9299                                    ;jmp    SYS_RET_OK
9300                                    ; 12/01/2024
9301 000012C6 7482                      je      short setdpbf_29
9302
9303                                setdpbf_31:
9304 000012C8 F7C270FF                  test    dx,0FF70h
9305                                    ;jz     short setdpbf_33      ; bit 4-6 of DPB.EXT_FLAGS must be 0
9306                                    ;mov    al,57h                ; error_invalid_parameter
9307                                    ; 12/01/2024
9308 000012CC 75B6                      jnz     short setdpbf_25
9309                                setdpbf_32:
9310                                    ;;jmp   short setdpbf_28
9311                                    ; 12/01/2024
9312                                    ;jmp    short setdpbf_21
9313
9314                                setdpbf_33:
9315 000012CE B001                      mov     al,1                  ; error_invalid_function
9316                                                                  ; (modification is not allowed)
9317                                    ;jmp    short setdpbf_32
9318                                    ; 12/01/2024
9319 000012D0 EB83                      jmp     short setdpbf_21
9320
9321                                ; ---------------------------------------------------------------------------
9322
9323                                ;===========================================================================
9324                                ; PARSE.ASM, MSDOS 6.0, 1991
9325                                ;===========================================================================
9326                                ; 19/07/2018 - Retro DOS v3.0
9327                                ; 15/05/2019 - Retro DOS v4.0
9328
9329                                ; System calls for parsing command lines
9330                                ;
9331                                ;   $PARSE_FILE_DESCRIPTOR
9332                                ;
9333                                ;   Modification history:
9334                                ;
9335                                ;       Created: ARR 30 March 1983
9336                                ;                EE PathParse 10 Sept 1983
9337                                ;
9338
9339                                ;BREAK <$Parse_File_Descriptor -- Parse an arbitrary string into an FCB>
9340                                ;---------------------------------------------------------------------------
9341                                ; Inputs:
9342                                ;       DS:SI Points to a command line
9343                                ;       ES:DI Points to an empty FCB
9344                                ;       Bit 0 of AL = 1 At most one leading separator scanned off
9345                                ;                     = 0 Parse stops if separator encountered
9346                                ;       Bit 1 of AL = 1 If drive field blank in command line - leave FCB
9347                                ;                     = 0  "    "    "    "         "      "  - put 0 in FCB
9348                                ;       Bit 2 of AL = 1 If filename field blank - leave FCB
9349                                ;                     = 0  "        "       "       - put blanks in FCB
9350                                ;       Bit 3 of AL = 1 If extension field blank - leave FCB
9351                                ;                     = 0  "        "       "       - put blanks in FCB
9352                                ; Function:
9353                                ;       Parse command line into FCB
9354                                ; Returns:
9355                                ;       AL = 1 if '*' or '?' in filename or extension, 0 otherwise
9356                                ;       DS:SI points to first character after filename
9357                                ;---------------------------------------------------------------------------
9358
9359                                    ; 13/01/2024
9360                                    ; PCDOS 7.1 IBMDOS.COM - DOSCODE:51BFh
9361                                    ; MSDOS 6.22 MSDOS.SYS - DOSCODE:4D22h
9362
9363                                _$PARSE_FILE_DESCRIPTOR:
9364 000012D2 E88F49                    call    MAKEFCB
9365 000012D5 56                        PUSH    SI
9366 000012D6 E89EF1                    call    Get_User_Stack
9367                                    ;pop    word [si+8]
9368 000012D9 8F4408                    POP     word [SI+user_env.user_SI]
9369 000012DC C3                        retn
9370
9371                                ; ---------------------------------------------------------------------------
9372                                ; 13/01/2024 - Retro DOS v5.0 (PCDOS 7.1 IBMDOS.COM - DOSCODE:51CAh)
9373                                ; ---------------------------------------------------------------------------
9374
9375                                set_exerr_locus_unk:
9376 000012DD 50                        push    ax
9377 000012DE B001                      mov     al,1          ; errLOC_Unk
9378                                set_exerr_locus:
9379 000012E0 36A2[2303]                    mov [ss:EXTERR_LOCUS],al
9380 000012E4 58                            pop ax
9381 000012E5 C3                            retn
9382
9383                                set_exerr_locus_disk:
9384 000012E6 50                        push    ax
9385 000012E7 B002                      mov     al,2          ; errLOC_Disk
9386 000012E9 EBF5                      jmp     short set_exerr_locus
9387
9388                                set_exerr_locus_ser:
9389 000012EB 50                        push    ax
9390 000012EC B004                      mov     al,4          ; errLOC_SerDev
9391 000012EE EBF0                      jmp     short set_exerr_locus
9392
9393                                set_exerr_locus_mem:
9394 000012F0 50                        push    ax
9395 000012F1 B005                      mov     al,5          ; errLOC_Mem
9396 000012F3 EBEB                      jmp     short set_exerr_locus
9397
9398                                ; ---------------------------------------------------------------------------
9399
9400                                ;===========================================================================
9401                                ; MISC.ASM, MSDOS 6.0, 1991
9402                                ;===========================================================================
9403                                ; 19/07/2018 - Retro DOS v3.0
9404
9405                                ; 29/04/2019 - Retro DOS v4.0
9406
9407                                ;ENTRYPOINTSEG     EQU     0CH
9408                                ;MAXDIF            EQU     0FFFH
9409                                ;SAVEXIT    EQU     10
```

```
                        ;WRAPOFFSET EQU     0FEF0h

                        ;
                        ;--------------------------------------------------------------------------
                        ;
                        ;** $SLEAZEFUNC - Get a Pointer to the Media Byte
                        ;
                        ;   Return Stuff sort of like old get fat call
                        ;
                        ;   ENTRY   none
                        ;   EXIT    DS:BX = Points to FAT ID byte (IBM only)
                        ;                   GOD help anyone who tries to do ANYTHING except
                        ;                   READ this ONE byte.
                        ;           DX = Total Number of allocation units on disk
                        ;           CX = Sector size
                        ;           AL = Sectors per allocation unit
                        ;              = -1 if bad drive specified
                        ;   USES    all
                        ;
                        ;** $SLEAZEFUNCDL - Get a Pointer to the Media Byte
                        ;
                        ;   Identical to $SLEAZEFUNC except (dl) = drive
                        ;
                        ;   ENTRY   (dl) = drive (0=default, 1=A, 2=B, etc.)
                        ;   EXIT    DS:BX = Points to FAT ID byte (IBM only)
                        ;                   GOD help anyone who tries to do ANYTHING except
                        ;                   READ this ONE byte.
                        ;           DX = Total Number of allocation units on disk
                        ;           CX = Sector size
                        ;           AL = Sectors per allocation unit
                        ;              = -1 if bad drive specified
                        ;   USES    all
                        ;
                        ;--------------------------------------------------------------------------
                        ;
                            ; 10/01/2024 - Retro DOS v5.0
                            ; PCDOS 7.1 IBMDOS.COM - DOSCODE:51E2h
                        _$SLEAZEFUNC:
                            ; 15/05/2019 - Retro DOS v4.0
000012F5 B200               MOV     DL,0
                        _$SLEAZEFUNCDL:
000012F7 16                 push    ss
000012F8 1F                 pop     ds

000012F9 88D0               MOV     AL,DL
000012FB E88F68             call    GETTHISDRV              ; Get CDS structure
                        SET_AL_RET:
                            ; MSDOS 3.3
                            ;;mov   al, 0Fh
                            ;MOV    AL,error_invalid_drive; Assume error ;AC000;

                            ; MSDOS 6.0 & MSDOS 3.3
000012FE 7229               JC      short BADSLDRIVE

00001300 E86020             call    DISK_INFO
                            ;JC     short SET_AL_RET        ; User FAILed to I 24
00001303 7224               jc      short BADSLDRIVE
00001305 8826[9805]         MOV     [FATBYTE],AH    ; FAT (MEDIA) ID byte

                            ; 10/01/2024 - Retro DOS v5.0 (PCDOS 7.1)
                            ;;;
00001309 30E4               xor     ah, ah          ; AH = 0
                                                    ; AL = sectors per cluster
0000130B 57                 push    di              ; di:bx = number of clusters
0000130C E81A05             call    TestNet
0000130F 5F                 pop     di
00001310 7205               jc      short sleazefunc1
00001312 51                 push    cx              ; bytes per sector
00001313 E85121             call    modify_cluster_count
00001316 59                 pop     cx              ; ax = sectors per cluster (modified)
                                                    ; dx = number of clusters (modified)
                                                    ; if bytes per cluster > 16384
                                                    ;    and hw of cluster count > 0
                                                    ;    bx = 0FFFEh (invalidated parm sign)
                        sleazefunc1:
                            ;;;

                        ; NOTE THAT A FIXED MEMORY CELL IS USED --> THIS CALL IS NOT
                        ; RE-ENTRANT. USERS BETTER GET THE ID BYTE BEFORE THEY MAKE THE
                        ; CALL AGAIN

                            ;MOV    DI,FATBYTE
                            ; 10/01/2024
                            ;XOR    AH,AH                   ; AL has sectors/cluster
00001317 E85DF1             call    Get_User_Stack
                            ;mov    [si+4],cx
                            ;mov    [si+6],bx
                            ;mov    [si+2],di
0000131A 894C04             MOV     [SI+user_env.user_CX],CX
0000131D 895C06             MOV     [SI+user_env.user_DX],BX
                            ;MOV    [SI+user_env.user_BX],DI
                            ; 10/01/2024
00001320 C74402[9805]       MOV     word [SI+user_env.user_BX],FATBYTE

                            ;mov    [si+0Eh],ss
00001325 8C540E             MOV      [SI+user_env.user_DS],SS ; stash correct pointer

00001328 C3                 retn

                        BADSLDRIVE:
00001329 E961F3             jmp     FCB_RET_ERR

                        ;
                        ;--------------------------------------------------------------------------
                        ;
                        ;** $Get_INDOS_Flag - Return location of DOS Critical Section Flag
                        ;
                        ;   Returns location of DOS status for interrupt routines
                        ;
                        ;   ENTRY   none
                        ;   EXIT    (es:bx) = flag location
                        ;   USES    all
                        ;
                        ;--------------------------------------------------------------------------
                        ;

                        _$GET_INDOS_FLAG:
0000132C E848F1             CALL    Get_User_Stack
                            ;MOV    WORD [SI+2],INDOS
0000132F C74402[2103]       MOV     word [SI+user_env.user_BX],INDOS
                        getin_segm: ; 13/01/2024
                            ;MOV    [SI+10H],SS
00001334 8C5410             MOV     [SI+user_env.user_ES],SS
```

```
9534 00001337 C3                        RETN
9535
9536                                ;
9537                                ;--------------------------------------------------------------------------
9538                                ;
9539                                ;** $Get_IN_Vars - Return Pointer to DOS Variables
9540                                ;
9541                                ;   Return a pointer to interesting DOS variables This call is version
9542                                ;   dependent and is subject to change without notice in future versions.
9543                                ;   Use at risk.
9544                                ;
9545                                ;   ENTRY   none
9546                                ;   EXIT    (es:bx) = address of SYSINITVAR
9547                                ;   uses    ALL
9548                                ;
9549                                ;--------------------------------------------------------------------------
9550                                ;
9551
9552                                    ; 13/01/2024 - Retro DOS v5.0
9553                                    ; PCDOS 7.1 IBMDOS.COM - DOSCODE:5226h
9554                                    ; MSDOS 6.22 MSDOS.SYS - DOSCODE:4D65h
9555                                    ; MSDOS 5.0 MSDOS.SYS - DOSCODE:4D58h
9556
9557                                _$GET_IN_VARS:
9558 00001338 E83CF1                    CALL    Get_User_Stack
9559                                    ;MOV    WORD [SI+2],SYSINITVAR
9560                                    ;MOV    word [SI+user_env.user_BX],SYSINITVAR
9561 0000133B C74402[2600]              MOV     word [SI+user_env.user_BX],SYSINITVARS
9562                                    ; 13/01/2024
9563                                    ;;MOV   [SI+10H],SS
9564                                    ;MOV    [SI+user_env.user_ES],SS
9565                                    ;RETN
9566 00001340 EBF2                      jmp     short getin_segm
9567                                ;
9568                                ;--------------------------------------------------------------------------
9569                                ;
9570                                ;** $Get_Default_DPB - Return a pointer to the Default DPB
9571                                ;
9572                                ;   Return pointer to drive parameter table for default drive
9573                                ;
9574                                ;   ENTRY   none
9575                                ;   EXIT    (ds:bx) = DPB address
9576                                ;   USES    all
9577                                ;
9578                                ;** $Get_DPB - Return a pointer to a specified DPB
9579                                ;
9580                                ;   Return pointer to a specified drive parameter table
9581                                ;
9582                                ;   ENTRY   (dl) = drive # (0 = default, 1=A, 2=B, etc.)
9583                                ;   EXIT    (al) = 0 iff ok
9584                                ;             (ds:bx) = DPB address
9585                                ;           (al) = -1 if bad drive
9586                                ;   USES    all
9587                                ;
9588                                ;--------------------------------------------------------------------------
9589                                ;
9590
9591                                ; 10/01/2024
9592                                %if 0
9593
9594                                ; 15/05/2019 - Retro DOS v4.0
9595
9596                                _$GET_DEFAULT_DPB:
9597                                    MOV     DL,0
9598                                _$GET_DPB:
9599                                    push    ss
9600                                    pop     ds
9601
9602                                    MOV     AL,DL
9603                                    call    GETTHISDRV              ; Get CDS structure
9604                                    JC      short ISNODRV           ; no valid drive
9605                                    LES     DI,[THISCDS]            ; check for net CDS
9606                                    ;;test  word [es:di+43h],8000h
9607                                    ;TEST   word [ES:DI+curdir.flags],curdir_isnet
9608                                    ;test   byte [es:di+44h],80h
9609                                    test    byte [ES:DI+curdir.flags+1],(curdir_isnet>>8)
9610                                    JNZ     short ISNODRV           ; No DPB to point at on NET stuff
9611                                    call    ECritDisk
9612                                    call    FATREAD_CDS             ; Force Media Check and return DPB
9613                                    call    LCritDisk
9614                                    JC      short ISNODRV           ; User FAILed to I 24, only error we
9615                                                                    ;   have.
9616                                    call    Get_User_Stack
9617                                    ;mov    [si+2],bp
9618                                    MOV     [SI+user_env.user_BX],BP
9619                                    ;mov    [si+0Eh],es
9620                                    MOV     [SI+user_env.user_DS],ES
9621                                    XOR     AL,AL
9622                                    retn
9623                                ISNODRV:
9624                                    MOV     AL,-1
9625                                    retn
9626
9627                                %else
9628
9629                                ; 13/01/2024
9630                                ; 10/01/2024 - Retro DOS v5.0
9631                                ; PCDOS 7.1 IBMDOS.COM - DOSCODE:5230h
9632
9633                                _$GET_DEFAULT_DPB:
9634 00001342 B200                      MOV     DL,0
9635                                _$GET_DPB:
9636 00001344 16                        push    ss
9637 00001345 1F                        pop     ds
9638
9639 00001346 88D0                      MOV     AL,DL
9640 00001348 E84268                    call    GETTHISDRV              ; Get CDS structure
9641 0000134B B00F                      mov     al,0Fh                  ; error_invalid_drive
9642 0000134D 7315                      jnc     short getdpb_3
9643
9644                                    ; no valid drive
9645                                getdpb_1:
9646                                    ; 13/01/2024
9647 0000134F E825F1                    call    Get_User_Stack
9648 00001352 813C0273                  cmp     word [si],7302h                 ; INT 21h, AX = 7302h ? Get_ExtDPB
9649                                                                    ; GET EXTENDED DPB
9650 00001356 7406                      je       short getdpb_2
9651 00001358 813C0473                  cmp     word [si],7304h                 ; INT 21h, AX = 7304h ? Set_DPBforFormat
9652                                                                    ; Set DPB TO USE FOR FORMATTING
9653 0000135C 7503                      jne     short ISNODRV
9654                                getdpb_2:
9655 0000135E E917F3                    jmp     SYS_RET_ERR
9656
9657                                ISNODRV:
```

```
9658 00001361 B0FF                      mov     al,0FFh ; -1            ; invalid (or network) drive
9659 00001363 C3                        retn
9660
9661                                getdpb_3:
9662 00001364 C43E[A205]                LES     DI,[THISCDS]            ; check for net CDS
9663                                    ;;test  word [es:di+43h],8000h
9664                                    ;TEST   word [ES:DI+curdir.flags],curdir_isnet
9665                                    ;test   byte [es:di+44h],80h
9666 00001368 26F6454480               test    byte [ES:DI+curdir.flags+1],(curdir_isnet>>8)
9667                                    ;JNZ    short ISNODRV           ; No DPB to point at on NET stuff
9668                                    ; 10/01/2024
9669 0000136D 75E0                      jnz     short getdpb_1
9670 0000136F E87005                    call    ECritDisk
9671 00001372 E8E651                    call    FATREAD_CDS             ; Force Media Check and return DPB
9672 00001375 E89705                    call    LCritDisk
9673                                    ;JC     short ISNODRV           ; User FAILed to I 24,
9674 00001378 B053                      mov     al,53h ; error_FAIL_I24 ;  only error we have.
9675 0000137A 72D3                      jc      short getdpb_1
9676 0000137C E8F8F0                    call    Get_User_Stack
9677                                    ; 13/01/2024
9678 0000137F 813C0473                  cmp     word [si],7304h                 ; INT 21h, AX = 7304h ?
9679 00001383 7503                      jnz     short getdpb_4
9680 00001385 E9E8FD                    jmp     Set_DPBforFormat
9681
9682                                    ; 13/01/2024
9683                                getdpb_4:
9684 00001388 813C0273                  cmp     word [si],7302h                 ; INT 21h, AX = 7302h ?
9685 0000138C 7410                      je      short getdpb_5
9686 0000138E 26837E0F00                cmp     word [es:bp+0Fh],0
9687 00001393 74BA                      jz      short getdpb_1
9688 00001395 896C02                    mov     [si+2],bp
9689 00001398 8C440E                    mov     [si+0Eh],es
9690 0000139B 30C0                      xor     al,al                   ; status = 0 = successful
9691 0000139D C3                        retn
9692
9693                                getdpb_5:
9694 0000139E B018                      mov     al,18h                  ; error_bad_length
9695 000013A0 837C043F                  cmp     word [si+4],63          ; [SI+user_env.user_CX]
9696                                                                    ; length of buffer (must be 63 bytes)
9697 000013A4 72B8                      jb      short getdpb_2          ; error
9698
9699 000013A6 06                        push    es                      ; ES:BP = Drive parameter block
9700 000013A7 55                        push    bp
9701 000013A8 8E4410                    mov     es,[si+16]              ; [SI+user_env.user_ES]
9702 000013AB 8B7C0A                    mov     di,[si+10]              ; [SI+user_env.user_DI]
9703 000013AE 8B5C08                    mov     bx,[si+8]               ; [SI+user_env.user_SI] (!)
9704 000013B1 5E                        pop     si
9705 000013B2 1F                        pop     ds
9706
9707 000013B3 B83D00                    mov     ax,61                   ; length of following data (003Dh)
9708 000013B6 AB                        stosw
9709 000013B7 89C1                      mov     cx,ax
9710 000013B9 57                        push    di
9711 000013BA F3A4                      rep movsb
9712 000013BC 5F                        pop     di
9713
9714                                    ; 13/01/2024
9715 000013BD 06                        push    es
9716 000013BE 1F                        pop     ds
9717 000013BF 31C0                      xor     ax,ax ; 0
9718
9719 000013C1 81FBA6F1                  cmp     bx,0F1A6h               ; (!) signature (undocumented),
9720                                                                    ;  must be 0F1A6h to get device driver
9721                                                                    ;  address and next-DPB pointer
9722                                                                    ; (Ref: Ralf Brown's Interrupt List)
9723 000013C5 740E                      je      short getdpb_6
9724
9725                                    ;xor    ax,ax   ; 0
9726 000013C7 48                        dec     ax      ; -1
9727                                    ;mov    [es:di+19h],ax          ; pointer to next DPB (invalidated)
9728                                    ;mov    [es:di+1Bh],ax
9729                                    ;mov    [es:di+13h],ax          ; pointer to driver address (invalidated)
9730                                    ;mov    [es:di+15h],ax
9731                                    ; 13/01/2024
9732 000013C8 894519                    mov     [di+19h],ax ; -1        ; pointer to next DPB (invalidated)
9733 000013CB 89451B                    mov     [di+1Bh],ax
9734 000013CE 894513                    mov     [di+13h],ax             ; pointer to driver address (invalidated)
9735 000013D1 894515                    mov     [di+15h],ax
9736
9737                                    ; 13/01/2024
9738 000013D4 40                        inc     ax ; 0
9739                                getdpb_6:
9740                                    ; 13/01/2024
9741                                    ; ax = 0
9742 000013D5 39450F                    cmp     [di+0Fh],ax ; 0
9743                                    ;cmp    [es:di+0Fh],ax ; 0      ; FAT (16 bit) size ?
9744                                    ;;cmp   word [es:di+0Fh],0
9745 000013D8 7441                      jz      short getdpb_8          ; FAT32
9746                                                                    ; FAT16 or FAT12
9747                                    ;mov    ax,[es:di+0Dh]          ; DPB.MAX_CLUSTER
9748                                    ;mov    [es:di+2Dh],ax          ; DPB.LAST_CLUSTER
9749                                    ;mov    ax,[es:di+0Fh]          ; DPB.FAT_SIZE
9750                                    ;mov    [es:di+31h],ax          ; DPB.FAT32_SIZE
9751                                    ;mov    ax,[es:di+1Dh]          ; DPB.NEXT_FREE
9752                                    ;mov    [es:di+39h],ax          ; DPB.FAT32_NXTFREE
9753                                    ;mov    ax,[es:di+0Bh]          ; DPB.FIRST_SECTOR
9754                                    ;mov    [es:di+29h],ax          ; DPB.FCLUS_FSECTOR
9755                                    ; 13/01/2024
9756 000013DA 8B450D                    mov     ax,[di+0Dh]             ; DPB.MAX_CLUSTER
9757 000013DD 89452D                    mov     [di+2Dh],ax             ; DPB.LAST_CLUSTER
9758 000013E0 8B450F                    mov     ax,[di+0Fh]             ; DPB.FAT_SIZE
9759 000013E3 894531                    mov     [di+31h],ax             ; DPB.FAT32_SIZE
9760 000013E6 8B451D                    mov     ax,[di+1Dh]             ; DPB.NEXT_FREE
9761 000013E9 894539                    mov     [di+39h],ax             ; DPB.FAT32_NXTFREE
9762 000013EC 8B450B                    mov     ax,[di+0Bh]             ; DPB.FIRST_SECTOR
9763 000013EF 894529                    mov     [di+29h],ax             ; DPB.FCLUS_FSECTOR
9764
9765 000013F2 31C0                      xor     ax,ax   ; 0
9766                                    ;mov    [es:di+2Fh],ax          ; DPB.LAST_CLUSTER high word
9767                                    ;mov    [es:di+33h],ax          ; DPB.FAT32_SIZE high word
9768                                    ;mov    [es:di+3Bh],ax          ; DPB.FAT32_NXTFREE high word
9769                                    ;mov    [es:di+2Bh],ax          ; DPB.FCLUS_FSECTOR high word
9770                                    ;mov    [es:di+35h],ax          ; DPB.ROOT_CLUSTER
9771                                    ;mov    [es:di+37h],ax          ; DPB.ROOT_CLUSTER high word
9772                                    ;mov    [es:di+23h],ax          ; DPB.EXT_FLAGS
9773                                    ; 13/01/2024
9774 000013F4 89452F                    mov     [di+2Fh],ax ; 0                 ; DPB.LAST_CLUSTER high word
9775 000013F7 894533                    mov     [di+33h],ax             ; DPB.FAT32_SIZE high word
9776 000013FA 89453B                    mov     [di+3Bh],ax             ; DPB.FAT32_NXTFREE high word
9777 000013FD 89452B                    mov     [di+2Bh],ax             ; DPB.FCLUS_FSECTOR high word
9778 00001400 894535                    mov     [di+35h],ax             ; DPB.ROOT_CLUSTER
9779 00001403 894537                    mov     [di+37h],ax             ; DPB.ROOT_CLUSTER high word
9780 00001406 894523                    mov     [di+23h],ax             ; DPB.EXT_FLAGS
9781 00001409 48                        dec     ax      ; -1
```

```
                                    ;mov    [es:di+25h],ax        ; DPB.FSINFO_SECTOR (invalidated)
                                    ;mov    [es:di+27h],ax        ; DPB.BKBOOT_SECTOR (invalidated)
                                    ; 13/01/2024
0000140A 894525                     mov     [di+25h],ax           ; DPB.FSINFO_SECTOR (invalidated)
0000140D 894527                     mov     [di+27h],ax           ; DPB.BKBOOT_SECTOR (invalidated)
00001410 3B451F                     cmp     ax,[di+1Fh]
                                    ;cmp    ax,[es:di+1Fh]        ; DPB.FREE_COUNT (= -1 ?)
00001413 7401                       je      short getdpb_7
00001415 40                         inc     ax      ; -1 -> 0
                                getdpb_7:
00001416 894521                     mov     [di+21h],ax
                                    ;mov    [es:di+21h],ax        ; DPB.PB.FREE_COUNT high word
                                ;getdpb_8:
                                    ; 26/06/2024
00001419 31C0                       xor     ax,ax                 ; status = 0 = successful
                                getdpb_8:   ; 03/07/2024
0000141B E950F2                     jmp     SYS_RET_OK

                                %endif

                                ;
                                ;--------------------------------------------------------------------------
                                ;
                                ;** $Disk_Reset - Flush out Dirty Buffers
                                ;
                                ;   $DiskReset flushes and invalidates all buffers. BUGBUG - do
                                ;           we really invalidate? SHould we? THis screws non-removable
                                ;           caching. Maybe CHKDSK relies upon it, though....
                                ;
                                ;   ENTRY   none
                                ;   EXIT    none
                                ;   USES    all
                                ;
                                ;--------------------------------------------------------------------------
                                ;
                                    ; 06/11/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
                                    ; DOSCODE:4D94h
                                    ; 13/01/2024 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMDOS.COM)
                                    ; DOSCODE:5322h
                                _$DISK_RESET:
                                    ; 15/05/2019 - Retro DOS v4.0
0000141E B0FF                       mov     al,0FFh ; -1
00001420 16                         push    ss
00001421 1F                         pop     ds
                                    ; 06/11/2022
                                    ;MOV    AL,-1
00001422 E8BD04                     call    ECritDisk
                                    ; MSDOS 6.0
                                    ;;or    word [DOS34_FLAG],4
                                    ;or     word [DOS34_FLAG],FROM_DISK_RESET    ;AN000;
00001425 800E[1106]04               or      byte [DOS34_FLAG],FROM_DISK_RESET ; 4 ; 15/05/2019
0000142A E86B56                     call    FLUSHBUF
                                    ; MSDOS 6.0
                                    ;and    word [DOS34_FLAG],0FFFBh
                                    ; 06/11/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
                                    ;and    word [DOS34_FLAG],NO_FROM_DISK_RESET ;AN000;
                                    ; 15/12/2022
0000142D 8026[1106]FB               and     byte [DOS34_FLAG],NO_FROM_DISK_RESET ; 0FBh ; 15/05/2019

                                    ; 13/01/2024 - Retro DOS v5.0
                                    ; (PCDOS 7.1 IBMDOS.COM)
                                    ;;;
00001432 C42E[2600]                 les     bp,[DPBHEAD]
                                drst_1:
00001436 83FDFF                     cmp     bp,0FFFFh       ; -1 ?
00001439 7409                       je      short drst_2    ; yes, it is the last DPB
0000143B E82E20                     call    update_fat32_fsinfo ; update FSINFO (sector) parameters
0000143E 26C46E19                   les     bp,[es:bp+19h] ; DPB.NEXT_DPB
00001442 EBF2                       jmp     short drst_1
                                drst_2:
                                    ;;;

00001444 C706[8310]0000             mov     word [SC_STATUS],0    ; Throw out secondary cache ; M041
                                ;
                                ; We will "ignore" any errors on the flush, and go ahead and invalidate. This
                                ; call doesn't return any errors and it is supposed to FORCE a known state, so
                                ; let's do it.
                                ;
                                ; Invalidate 'last-buffer' used
                                ;
0000144A BBFFFF                     MOV     BX,-1 ; 0FFFFh
0000144D 891E[2000]                 MOV     [LastBuffer+2],BX
00001451 891E[1E00]                 MOV     [LastBuffer],BX

                                    ; MSDOS 3.3
                                    ; IBMDOS.COM, Offset 1C66h
                                    ;;;;
                                    ;lds    si,[BUFFHEAD]
                                    ;mov    ax,20FFh              ; .buf_ID,    AL = FFh (Free buffer)
                                                                  ; .buf_flags, AH = 0, reset/clear
                                ;DRST_1:
                                    ;;mov   [si+4],ax
                                    ;mov    [si+BUFFINFO.buf_ID],ax
                                    ;lds    si,[SI]
                                    ;cmp    si,bx ; -1
                                    ;je     short DRST_2
                                    ;;mov   [si+4],ax
                                    ;mov    [si+BUFFINFO.buf_ID],ax
                                    ;lds    si,[SI]
                                    ;cmp    si,bx
                                    ;jne    short DRST_1
                                    ;;;;
                                ;DRST_2:
00001455 E8B704                     call    LCritDisk
00001458 B8FFFF                     MOV     AX,-1
                                    ; 07/12/2022
                                    ;mov    ax,0FFFFh
                                    ;CallInstall NetFlushBuf,MultNET,32,AX,AX
0000145B 50                         push    ax ; * MSDOS 6.0 ; 15/05/2019
0000145C B82011                     mov     ax,1120h
0000145F CD2F                       int     2Fh     ; Multiplex - NETWORK REDIRECTOR - FLUSH ALL DISK BUFFERS
                                                    ; DS = DOS CS
                                                    ; Return: CF clear (successful)
00001461 58                         pop     ax ; * MSDOS 6.0 ; 15/05/2019

00001462 C3                         retn

                                    ; 19/07/2018 - Retro DOS v3.0

                                ;
                                ;   BREAK <$SetDPB - Create a valid DPB from a user-specified BPB>
                                ;
                                ;--------------------------------------------------------------------------
                                ;
```

```
                                ;** $SetDPB - Create a DPB
                                ;
                                ;   SetDPB Creates a valid DPB from a user-specified BPB
                                ;
                                ;   ENTRY   ES:BP Points to DPB
                                ;           DS:SI Points to BPB
                                ;   EXIT    DPB setup
                                ;   USES    ALL but BP, DS, ES
                                ;
                                ;----------------------------------------------------------------------------
                                ;

                                ; 10/05/2019 - Retro DOS v4.0

                                ; DOSCODE:4DD6h (MSDOS 6.21, MSDOS.SYS)

                                ; MSDOS 6.0
00001463 0300                   word3:      dw      3                       ; M008 -- word value for divides

                                ; 06/11/2022 - Retro DOS v4.0 (Modified MSDOS 5.0)
                                ; DOSCODE:4DC9h (MSDOS 5.0, MSDOS.SYS)

                                ; 13/01/2024 - Retro DOS v5.0 (Modified PCDOS 7.1)
                                ; DOSCODE:5369h (PCDOS 7.1, IBMDOS.COM)

                                ; 13/01/2024 - Retro DOS v5.0 (Modified PCDOS 7.1)
                                ; Windows ME IO.SYS (Extracted) - BIOSCODE:4EF8h

                                ;procedure   $SETDPB,NEAR

                                    ; 12/04/2024
                                _$SETDPB:
00001465 89EF                       MOV     DI,BP
                                    ;ADD    DI,2                    ; Skip over dpb_drive and dpb_UNIT
                                    ; 13/01/2024
00001467 47                         inc     di
00001468 47                         inc     di
00001469 AD                         LODSW
0000146A AB                         STOSW                           ; dpb_sector_size

                                    ; 13/01/2024
                                    ; Retro DOS v5.0 (PCDOS 7.1 IBMDOS.COM)
                                    ;;;
0000146B 81F95845                   cmp     cx,4558h                ; 'XE' (NASM syntax)
                                                                    ; CX = signature 4558h ('EX') for FAT32 extended BPB/DPB
0000146F 7506                       jne     short not_fat32_extension
00001471 81FA5241                   cmp     dx,4152h                ; 'RA' (NASM syntax)
                                                                    ; DX = signature 4152h ('AR') for FAT32 extended BPB/DPB
00001475 7402                       je      short chk_fat32_conditions
                                ;not_fat32_extension:                   ; ...
00001477 31C9                       xor     cx,cx   ; 0     ; (Do not use FAT32 extensions -32 bit parameters-)
                                chk_fat32_conditions:
00001479 51                         push    cx      ; (*)
                                    ;;;

                                    ; 13/01/2024
                                    ; MSDOS 6.0
                                    ;cmp    byte [si+3],0
                                    ;CMP    BYTE [SI+A_BPB.BPB_NUMBEROFFATS-2],0
0000147A 807C0300                   cmp     byte [SI+A_BPB.NUMBEROFFATS-2],0 ; FAT file system drive ;AN000;
                                    ;JNZ    short yesfat                ; yes                   ;AN000;
0000147E 740C                       jz      short nofat ; 13/01/2024 (PCDOS 7.1)

                                    ; 13/01/2024 - Retro DOS v5.0
                                    ;;;
                                    ;cmp    word [si+9],0  ; .BPB_SECTORSPERFAT ; BPB_FATSz16
00001480 837C0900                   cmp     word [si+A_BPB.SECTORSPERFAT-2],0
00001484 7516                       jnz     short yesfat
                                    ;cmp    word [si+29],0 ; .BPB_FAT32VERSION ; BPB_FSVer
00001486 837C1D00                   cmp     word [si+A_BPB.FSVER-2],0
0000148A 7410                       jz      short yesfat
                                nofat:
                                    ;;;

                                    ; 13/01/2024
                                    ;;mov   byte [es:di+4],0
                                    ;MOV    BYTE [ES:DI+DPB.FAT_COUNT-4],0
                                    ;JMP    short setend                ; NO            ;AN000;

                                    ; 13/01/2024 (WINME IO.SYS)
                                    ;mov    byte [es:di+4],0 ; DPB.FAT_COUNT
                                    ;xor    eax,eax
                                    ;jmp    setend

                                    ; 13/01/2024 (PCDOS7.1 IBMDOS.COM)
0000148C 31C0                       xor     ax,ax ; 0
0000148E 26884504                   mov     [es:di+DPB.FAT_COUNT-4],al ; DPB.FAT_COUNT = 0

                                    ; 13/01/2024 (not necessary)
                                    ;add    di,15   ; DPB.DRIVER_ADDR

00001492 83C60B                     add     si,11   ; .BPB_SECTORSPERTRACK

                                    ;mov    [es:bp+15],ax  ; DPB.FAT_SIZE = 0
00001495 2689460F                   mov     [es:bp+DPB.FAT_SIZE],ax

00001499 E9B500                     jmp     setend

                                yesfat: ; 10/08/2018
0000149C 89C2                       MOV     DX,AX
0000149E AC                         LODSB
                                    ;;;
                                    ; 13/01/2024 - Retro DOS v5.0 (PCDOS 7.1 IBMDOS.COM)
0000149F 08C0                       or      al,al
000014A1 74E9                       jz      short nofat
                                    ;;;
                                    ;DEC    AL
                                    ; 17/12/2022
000014A3 48                         dec     ax
000014A4 AA                         STOSB                           ; dpb_cluster_mask
                                    ;INC    AL
000014A5 40                         inc     ax
000014A6 30E4                       XOR     AH,AH
                                LOG2LOOP:
000014A8 A801                       test    AL,1
000014AA 7506                       JNZ     short SAVLOG
000014AC FEC4                       INC     AH
000014AE D0E8                       SHR     AL,1
000014B0 EBF6                       JMP     SHORT LOG2LOOP
                                SAVLOG:
000014B2 88E0                       MOV     AL,AH
000014B4 AA                         STOSB                           ; dpb_cluster_shift
000014B5 88C3                       MOV     BL,AL
000014B7 A5                         MOVSW                           ; dpb_first_FAT Start of FAT (# of reserved sectors)
```

```
10030 000014B8 AC                       LODSB
10031 000014B9 AA                       STOSB                   ; dpb_FAT_count Number of FATs
10032                                ;  OR      AL,AL                   ; NONFAT ?                              ;AN000;
10033                                ;  JZ      short setend            ; yes, don't do anything                ;AN000;
10034 000014BA 88C7                     MOV     BH,AL
10035 000014BC AD                       LODSW
10036 000014BD AB                       STOSW                   ; dpb_root_entries Number of directory entries
10037 000014BE B105                     MOV     CL,5
10038 000014C0 D3EA                     SHR     DX,CL                   ; Directory entries per sector
10039 000014C2 48                       DEC     AX
10040 000014C3 01D0                     ADD     AX,DX                   ; Cause Round Up
10041 000014C5 89D1                     MOV     CX,DX
10042 000014C7 31D2                     XOR     DX,DX
10043 000014C9 F7F1                     DIV     CX
10044 000014CB 89C1                     MOV     CX,AX                   ; Number of (root) directory sectors
10045 000014CD 47                       INC     DI
10046 000014CE 47                       INC     DI                      ; Skip dpb_first_sector
10047 000014CF A5                       MOVSW                   ; Total number of sectors in DSKSIZ (temp as dpb_max_cluster)
10048 000014D0 AC                       LODSB
10049                                   ;mov    [es:bp+17h],al
10050 000014D1 26884617                 MOV     [ES:BP+DPB.MEDIA],AL    ; Media byte
10051 000014D5 AD                       LODSW                           ; Number of sectors in a FAT
10052
10053                                   ;;;
10054                                   ;MSDOS 3.3
10055                                   ;
10056                                   ;STOSB          ; DPB.FAT_SIZE
10057                                   ;MUL    BH
10058
10059                                   ;MSDOS 6.0
10060                                   ;
10061 000014D6 AB                       STOSW           ; DPB.FAT_SIZE ;AC000;;>32mb dpb_FAT_size
10062
10063                                ; 13/01/2024
10064                                %if 0
10065                                   MOV     DL,BH                   ;AN000;;>32mb
10066                                   XOR     DH,DH                   ;AN000;;>32mb
10067                                   MUL     DX                      ;AC000;;>32mb Space occupied by all FATs
10068                                   ;;;
10069
10070                                   ;add    ax,[es:bp+6]
10071                                   ADD     AX,[ES:BP+DPB.FIRST_FAT]
10072                                   STOSW                           ; dpb_dir_sector
10073                                   ADD     AX,CX                   ; Add number of (root) directory sectors
10074                                   ;mov    [es:bp+0Bh],ax
10075                                   MOV     [ES:BP+DPB.FIRST_SECTOR],AX
10076
10077                                   ; MSDOS 6.0
10078                                   MOV     CL,BL                   ;F.C. >32mb                            ;AN000;
10079                                   ;;cmp   word [es:bp+0Dh],0
10080                                   ;CMP    WORD [ES:BP+DSKSIZ],0 ;F.C. >32mb                              ;AN000;
10081                                   ;JNZ    short normal_dpb        ;F.C. >32mb                            ;AN000;
10082                                   ; 06/11/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
10083                                   ; 15/12/2022
10084                                   ; 28/07/2019
10085                                   mov     bx,[ES:BP+DSKSIZ]
10086                                   or      bx,bx
10087                                   JNZ     short normal_dpb        ;F.C. >32mb                            ;AN000;
10088                                   ;CMP    WORD [ES:BP+DSKSIZ],0 ;F.C. >32mb                              ;AN000;
10089                                   ;JNZ    short normal_dpb        ;F.C. >32mb                            ;AN000;
10090
10091
10092                                   XOR     CH,CH                   ;F.C. >32mb                            ;AN000;
10093                                   ;mov    bx,[si+8]
10094                                   mov     bx,[si+A_BPB.BIGTOTALSECTORS-A_BPB.SECTORSPERTRACK] ; 01/01/2024 (temporary)
10095                                   ;MOV    BX,[SI+A_BPB.BPB_BIGTOTALSECTORS-A_BPB.BPB_SECTORSPERTRACK]     ;AN000;
10096                                   ;mov    dx,[si+10]
10097                                   mov     dx,[si+A_BPB.BIGTOTALSECTORS-A_BPB.SECTORSPERTRACK+2] ; 01/01/2024 (temporary)
10098                                   ;MOV    DX,[SI+A_BPB.BPB_BIGTOTALSECTORS-A_BPB.BPB_SECTORSPERTRACK+2]   ;AN000;
10099                                   SUB     BX,AX                   ;AN000;;F.C. >32mb
10100                                   SBB     DX,0                    ;AN000;;F.C. >32mb
10101                                   OR      CX,CX                   ;AN000;;F.C. >32mb
10102                                   JZ      short norot             ;AN000;;F.C. >32mb
10103                                rott:                                      ;AN000;;F.C. >32mb
10104                                   CLC                             ;AN000;;F.C. >32mb
10105                                   RCR     DX,1                    ;AN000;;F.C. >32mb
10106                                   RCR     BX,1                    ;AN000;;F.C. >32mb
10107                                   LOOP    rott                    ;AN000;;F.C. >32mb
10108                                norot:                                     ;AN000;
10109                                   ; 15/12/2022
10110                                   ;MOV    AX,BX                   ;AN000;;F.C. >32mb
10111                                   JMP     short setend            ;AN000;;F.C. >32mb
10112                                normal_dpb:
10113                                   ;;sub   ax,[es:bp+0Dh]
10114                                   ;SUB    AX,[ES:BP+DSKSIZ]
10115                                   ; 06/11/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
10116                                   ; 15/12/2022
10117                                   ; bx = [es:bp+DSKSIZ]
10118                                   ;sub    ax,bx ; 28/07/2019
10119                                   ;;SUB   AX,[ES:BP+DSKSIZ]
10120                                   ; 15/12/2022
10121                                   sub     bx,ax
10122                                   ;NEG    AX                      ; Sectors in data area
10123                                ;;  MOV     CL,BL                   ; dpb_cluster_shift
10124                                   ; 15/12/2022
10125                                   ; CL = cluster shift
10126                                   ; BX = number of data sectors
10127                                   ;SHR    AX,CL                   ; Div by sectors/cluster
10128                                   shr     bx,cl
10129                                setend:
10130                                ;  M008 - CAS
10131                                ;
10132                                   ; 15/12/2022
10133                                   inc     bx
10134                                   ;INC    AX                      ; +2 (reserved), -1 (count -> max)
10135                                ;
10136                                ;  There has been a bug in our fatsize calculation for so long
10137                                ;    that we can't correct it now without causing some user to
10138                                ;    experience data loss. There are even cases where allowing
10139                                ;    the number of clusters to exceed the fats is the optimal
10140                                ;    case -- where adding 2 more fat sectors would make the
10141                                ;    data field smaller so that there's nothing to use the extra
10142                                ;    fat sectors for.
10143                                ;
10144                                ;  Note that this bug had very minor known symptoms. CHKDSK would
10145                                ;    still report that there was a cluster left when the disk was
10146                                ;    actually full. Very graceful failure for a corrupt system
10147                                ;    configuration. There may be worse cases that were never
10148                                ;    properly traced back to this bug. The problem cases only
10149                                ;    occurred when partition sizes were very near FAT sector
10150                                ;    rounding boundaries, which were rare cases.
10151                                ;
10152                                ;  Also, it's possible that some third-party partition program might
10153                                ;    create a partition that had a less-than-perfect FAT calculation
```

```
10154                                  ;     scheme. In this hypothetical case, the number of allocation
10155                                  ;     clusters which don't actually have FAT entries to represent
10156                                  ;     them might be larger and might create a more catastrophic
10157                                  ;     failure. So we'll provide the safeguard of limiting the
10158                                  ;     max_cluster to the amount that will fit in the FATs.
10159                                  ;
10160                                  ;   ax = maximum legal cluster, ES:BP -> dpb
10161                                  
10162                                  ;   make sure the number of fat sectors is actually enough to
10163                                  ;     hold that many clusters. otherwise, back the number of
10164                                  ;     clusters down
10165                                  
10166                                      ; 15/12/2022
10167                                      ; bx = number of clusters
10168                                  
10169                                      ; 19/07/2018 - Retro DOS v3.0
10170                                      ; MSDOS 6.0
10171                                      ; 15/12/2022
10172                                      ;mov    bx,ax                  ; remember calculated # clusters
10173                                  
10174                                      ; 01/08/2018 (MSDOS 3.3)
10175                                      ;mov    al,[ES:BP+DPB.FAT_SIZE]
10176                                      ;xor    ah,ah
10177                                  
10178                                      ; 10/05/2019 - Retro DOS v4.0
10179                                      ;mov    ax,[ES:BP+0Fh]
10180                                      mov     ax,[ES:BP+DPB.FAT_SIZE]
10181                                  
10182                                      ;mul    word [es:bp+2]
10183                                      mul     word [ES:BP+DPB.SECTOR_SIZE] ; how big is the FAT?
10184                                      cmp     bx,4096-10  ; 0FF6h    ; test for 12 vs. 16 bit fat
10185                                      jb      short setend_fat12
10186                                      shr     dx,1
10187                                  
10188                                  ; 06/11/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
10189                                      ; 15/12/2022
10190                                  ;cs3 7/2/92
10191                                      jnz     short setend_faterr    ; some bonehead gave us more fatspace
10192                                                                     ; than enough for the maximum FAT,
10193                                                                     ; so go ahead and use the calculated
10194                                                                     ; number of clusters.
10195                                  ;cs3 7/2/92
10196                                  
10197                                      rcr     ax,1                   ; find number of entries
10198                                      cmp     ax,4096-10+1           ; would this truncation move us
10199                                  ;                                  ;  into 12-bit fatland?
10200                                      jb      short setend_faterr    ; then go ahead and let the
10201                                  ;                                  ;  inconsistency pass through
10202                                  ;                                  ;  rather than lose data by
10203                                  ;                                  ;  correcting the fat type
10204                                      jmp     short setend_fat16
10205                                  
10206                                  setend_fat12:
10207                                      add     ax,ax                  ; (fatsiz*2)/3 = # of fat entries
10208                                      adc     dx,dx
10209                                  
10210                                  ; 06/11/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
10211                                  ;cs3 7/2/92
10212                                      ; 15/12/2022
10213                                      cmp     dx,3                   ; if our fatspace is WAY more than
10214                                      jnb     short setend_faterr    ; we need, we may get an overflow
10215                                                                     ; here. Check for it and use
10216                                                                     ; the calculated size in this case.
10217                                  ;cs3 7/2/92
10218                                  
10219                                      div     word [cs:word3]
10220                                  
10221                                  setend_fat16:
10222                                      dec     ax                     ; limit at 1
10223                                      cmp     ax,bx                  ; is fat big enough?
10224                                      jbe     short setend_fat       ; use max value that'll fit
10225                                  
10226                                  setend_faterr:
10227                                      mov     ax,bx                  ; use calculated value
10228                                  
10229                                  setend_fat:
10230                                  
10231                                  ;   now ax = maximum legal cluster
10232                                  
10233                                  ;   end M008
10234                                  
10235                                  
10236                                      ;mov    [es:bp+0Dh], ax
10237                                      MOV     [ES:BP+DPB.MAX_CLUSTER],AX
10238                                  
10239                                      ;;mov   word [es:bp+1Ch],0  ; MSDOS 3.3
10240                                      ;mov    word [es:bp+1Dh],0  ; MSDOS 6.0
10241                                      MOV     word [ES:BP+DPB.NEXT_FREE],0
10242                                                                     ; Init so first ALLOC starts at
10243                                                                     ; begining of FAT
10244                                      ;;mov   word [es:bp+1Eh],-1 ; MSDOS 3.3
10245                                      ;mov    word [es:bp+1Fh],-1 ; MSDOS 6.0
10246                                      MOV     word [ES:BP+DPB.FREE_CNT],-1 ; current count is invalid.
10247                                  
10248                                      retn
10249                                  
10250                                  %else
10251                                      ; 13/01/2024 - Retro DOS v5.0
10252                                      ;;;
10253 000014D7 31D2                     xor     dx,dx ; 0
10254 000014D9 09C0                     or      ax,ax
10255 000014DB 750C                     jnz     short savlog1  ; 16 bit FAT size
10256 000014DD 5A                       pop     dx             ; (*) FAT32 extensions
10257                                                          ; (use 32 bit FAT and Root Dir size if >0)
10258 000014DE 52                       push    dx             ; (*)
10259 000014DF 09D2                     or      dx,dx
10260 000014E1 7406                     jz      short savlog1  ; Do not use FAT32 extensions
10261                                                          ; (do not use 32 bit FAT size field)
10262 000014E3 8B440C                   mov     ax,[si+12]     ; .BPB_SECTORSPERFAT32 ; BPB_FATSz32
10263 000014E6 8B540E                   mov     dx,[si+14]     ; .BPB_SECTORSPERFAT32+2
10264                                  savlog1:
10265 000014E9 51                       push    cx             ; (**) Root directory sectors
10266 000014EA 89C1                     mov     cx,ax          ; 32 bit multiply
10267 000014EC 88F8                     mov     al,bh          ; FAT count
10268 000014EE 30E4                     xor     ah,ah
10269 000014F0 F7E2                     mul     dx
10270 000014F2 91                       xchg    ax,cx
10271 000014F3 88FA                     mov     dl,bh          ; FAT count
10272 000014F5 30F6                     xor     dh,dh
10273 000014F7 F7E2                     mul     dx
10274 000014F9 01CA                     add     dx,cx
10275 000014FB 59                       pop     cx             ; (**)
10276 000014FC 39D0                     cmp     ax,dx
10277 000014FE 7504                     jne     short savlog2
```

```
10278
10279                                   ; 13/01/2024
10280                                   ;or     ax,ax
10281                                   ;jnz    short savlog2
10282
10283                                   ; 13/01/2024 (not necessary)
10284                                   ;inc    di
10285                                   ;inc    di
10286                                   ; di = DPB.DRIVER_ADDR
10287
10288                                   ;jmp    short setend
10289
10290                                   ; 13/01/2024
10291 00001500 09C0                    or      ax,ax
10292 00001502 744D                    jz      short setend
10293
10294                                 savlog2:
10295 00001504 26034606                 add     ax,[es:bp+DPB.FIRST_FAT]
10296                                   ;add    ax,[es:bp+6]   ; dx:ax = (total) FAT sectors
10297 00001508 83D200                   adc     dx,0
10298 0000150B AB                       stosw                  ; DPB.DIR_SECTOR
10299 0000150C 01C8                     add     ax,cx          ; + root directory size
10300 0000150E 2689460B                 mov     [es:bp+DPB.FIRST_SECTOR],ax
10301                                   ;mov    [es:bp+11],ax  ; DPB.FIRST_SECTOR ; First data sector
10302 00001512 83D200                   adc     dx,0
10303 00001515 59                       pop     cx             ; (*)
10304 00001516 51                       push    cx
10305 00001517 E308                     jcxz    savlog3
10306 00001519 26894629                 mov     [es:bp+DPB.FCLUS_FSECTOR],ax
10307                                   ;mov    [es:bp+41],ax  ; DPB.BIG_FIRST_SECTOR ; FAT32 first sector field
10308 0000151D 2689562B                 mov     [es:bp+DPB.FCLUS_FSECTOR+2],dx
10309                                   ;mov    [es:bp+43],dx
10310                                 savlog3:
10311 00001521 88D9                     mov     cl,bl          ; cluster shift
10312 00001523 26837E0D00               cmp     word [es:bp+DPB.MAX_CLUSTER],0
10313                                   ;cmp    word [es:bp+13],0 ; DPB.MAX_CLUSTER
10314                                                          ; (contains 16 bit .BPB_TOTALSECTORS as temporary)
10315 00001528 751D                     jnz     short normal_dpb
10316 0000152A 30ED                     xor     ch,ch
10317 0000152C 52                       push    dx     ; (**)
10318 0000152D 8B5C08                   mov     bx,[si+8]      ; SI points to .BPB_SECTORSPERTRACK and SI+8 is
10319                                                          ; .BPB_BIGTOTALSECTORS (32 bit total sectors)
10320 00001530 8B540A                   mov     dx,[si+10]
10321 00001533 29C3                     sub     bx,ax
10322 00001535 58                       pop     ax     ; (**)
10323 00001536 19C2                     sbb     dx,ax          ; dx:bx = data sectors (for cluster count calc)
10324 00001538 09C9                     or      cx,cx
10325 0000153A 7407                     jz      short norot
10326                                 rott:
10327 0000153C F8                       clc
10328 0000153D D1DA                     rcr     dx,1
10329 0000153F D1DB                     rcr     bx,1
10330 00001541 E2F9                     loop    rott
10331                                 norot:
10332 00001543 89D8                     mov     ax,bx          ; dx:ax = cluster count
10333 00001545 EB0A                     jmp     short setend
10334
10335                                 normal_dpb:
10336 00001547 262B460D                 sub     ax,[es:bp+DPB.MAX_CLUSTER]
10337                                   ;sub    ax,[es:bp+13]  ; first sector - total sectors
10338 0000154B 31D2                     xor     dx,dx
10339 0000154D F7D8                     neg     ax             ; data sectors = total sectors - first sector
10340 0000154F D3E8                     shr     ax,cl          ; cluster count
10341
10342                                 setend:
10343                                   ; 13/01/2024 (PCDOS 7.1 IBMDOS.COM)
10344 00001551 59                       pop     cx             ; (*) 0 = not 32 bit fat sectors
10345 00001552 51                       push    cx             ; (*)
10346
10347                                   ; si = BIOS Parameter Block + 13
10348
10349                                   ; 12/04/2024
10350                                   ; 13/01/2024 (PCDOS 7.1 IBMDOS.COM)
10351 00001553 83C001                   add     ax,1
10352 00001556 83D200                   adc     dx,0                   ; calculated # clusters HW
10353 00001559 89C3                     mov     bx,ax                  ; calculated # clusters LW
10354 0000155B 268B460F                 mov     ax,[es:bp+DPB.FAT_SIZE]
10355                                   ;mov    ax,[es:bp+15]          ; FAT size (16 bit)
10356 0000155F E30C                     jcxz    setend1                ; Do not use 32 bit FAT sectors field
10357 00001561 31C9                     xor     cx,cx
10358 00001563 09C0                     or      ax,ax
10359 00001565 7506                     jnz     short setend1
10360 00001567 8B440C                   mov     ax,[si+12]             ; .BPB_SECTORSPERFAT32  ; 32 bit FAT size field.
10361 0000156A 8B4C0E                   mov     cx,[si+14]             ; .BPB_SECTORSPERFAT32+2
10362                                 setend1:
10363 0000156D 52                       push    dx     ; (**)          ; cx:ax = FAT size (in sectors)
10364                                                                  ; dx:bx = calculated number of clusters
10365 0000156E 91                       xchg    ax,cx
10366 0000156F 26F76602                 mul     word [es:bp+DPB.SECTOR_SIZE]
10367                                   ;mul    word [es:bp+2]         ; DPB.SECTOR_SIZE
10368 00001573 91                       xchg    ax,cx
10369 00001574 26F76602                 mul     word [es:bp+DPB.SECTOR_SIZE]
10370                                   ;mul    word [es:bp+2]
10371 00001578 01CA                     add     dx,cx                  ; dx:ax = FAT size in bytes
10372 0000157A 59                       pop     cx     ; (**)          ; calculated # clusters HW
10373 0000157B 09C9                     or      cx,cx
10374                                   ;jnz    short setend2          ; FAT32
10375 0000157D 750B                     jnz     short setend3 ; 12/04/2024 (BugFix)
10376 0000157F 81FBF60F                 cmp     bx,0FF6h
10377 00001583 721E                     jb      short setend_fat12     ; FAT12
10378                                 setend2:
10379                                   ; (PCDOS 7.1 IBMDOS.COM)
10380                                   ;or     cx,cx                  ; HW of calculated cluster count
10381                                   ;jnz    short setend3
10382 00001585 83FBF6                   cmp     bx,0FFF6h
10383 00001588 720C                     jb      short setend4          ; FAT16
10384                                 setend3:
10385 0000158A D1EA                     shr     dx,1                   ; FAT32 ; 4 byte (32 bit) cluster number
10386                                                                  ; fatsiz/4 = # of fat entries
10387 0000158C D1D8                     rcr     ax,1
10388 0000158E D1EA                     shr     dx,1
10389 00001590 7408                     jz      short setend5          ; dx = 0
10390 00001592 D1D8                     rcr     ax,1
10391 00001594 EB1B                     jmp     short setend_fat16
10392
10393                                 setend4:
10394 00001596 D1EA                     shr     dx,1                   ; FAT16 ; 2 byte (16 bit) cluster number
10395                                                                  ; fatsiz/2 = # of fat entries
10396 00001598 7525                     jnz     short setend_faterr ; dx > 0
10397                                 setend5:
10398 0000159A D1D8                     rcr     ax,1                   ; FAT16 ; 2 byte (16 bit) cluster number
10399                                                                  ; fatsiz/2 = # of fat entries
10400 0000159C 3DF70F                   cmp     ax,0FF7h               ; 4096-10+1
10401 0000159F 721E                     jb      short setend_faterr
```

```
10402 000015A1 EB0E                     jmp     short setend_fat16
10403
10404                                   setend_fat12:
10405 000015A3 01C0                     add     ax,ax                   ; FAT12 ; 1.5 byte (12 bit) cluster number
10406                                                                   ; (fatsiz*2)/3 = # of fat entries
10407 000015A5 11D2                     adc     dx,dx
10408 000015A7 83FA03                   cmp     dx,3                    ; if our fatspace is more than we need
10409                                                                   ; use calculated size
10410 000015AA 7313                     jnb     short setend_faterr
10411 000015AC 2EF736[6314]             div     word [cs:word3]
10412                                   setend_fat16:
10413 000015B1 83E801                   sub     ax,1
10414 000015B4 83DA00                   sbb     dx,0
10415 000015B7 39CA                     cmp     dx,cx                   ; is fat big enough?
10416 000015B9 7704                     ja      short setend_faterr
10417 000015BB 39D8                     cmp     ax,bx
10418 000015BD 7604                     jbe     short setend_fat32      ; use max value that'll fit
10419                                   setend_faterr:
10420 000015BF 89D8                     mov     ax,bx                   ; use calculated value
10421 000015C1 89CA                     mov     dx,cx
10422                                   setend_fat32:
10423 000015C3 26837E0F00               cmp     word [es:bp+DPB.FAT_SIZE],0
10424                                   ;cmp    word [es:bp+15],0       ; DPB.FAT_SIZE ; 16 bit FAT size
10425 000015C8 750E                     jnz     short setend6
10426 000015CA 26C74611FFFF             mov     word [es:bp+DPB.DIR_SECTOR],-1
10427                                   ;mov    word [es:bp+17],0FFFFh; DPB.DIR_SECTOR
10428                                                                   ; (16 bit directory sector field)
10429 000015D0 26C7460D0000             mov     word [es:bp+DPB.MAX_CLUSTER],0
10430                                   ;mov    word [es:bp+13],0       ; DPB.MAX_CLUSTER (16 bit)
10431 000015D6 EB04                     jmp     short setend7
10432
10433                                   setend6:
10434 000015D8 2689460D                 mov     [es:bp+DPB.MAX_CLUSTER],ax
10435                                   ;mov    [es:bp+13],ax  ; DPB.MAX_CLUSTER = calculated last cluster number
10436                                   setend7:
10437 000015DC 59                       pop     cx      ; (*)           ; 1 = use FAT32 extensions
10438                                                                   ; 0 = don't use FAT32 extensions (32 bit fields)
10439 000015DD E353                     jcxz    setend_fat              ; do not use FAT32 extensions
10440                                   ;mov    [es:bp+45],ax           ; DPB.MAX_CLUSTER32 ; dx:ax = last cluster number
10441                                   ;mov    [es:bp+47],dx
10442 000015DF 2689462D                 mov     [es:bp+DPB.LAST_CLUSTER],ax
10443 000015E3 2689562F                 mov     [es:bp+DPB.LAST_CLUSTER+2],dx
10444 000015E7 B8FFFF                   mov     ax,0FFFFh ; -1
10445 000015EA 8D7E21                   lea     di,[bp+DPB.FREE_CNT_HW]
10446                                   ;lea    di,[bp+33]              ; DPB.FAT32_EXT ; FAT32 extensions
10447                                                                   ; -1 = ready
10448 000015ED AB                       stosw
10449 000015EE 8D7410                   lea     si,[si+16]              ; FAT32 flags
10450 000015F1 A5                       movsw                           ; DPB FAT32 flags ; [bp+23h]
10451 000015F2 83C606                   add     si,6
10452 000015F5 AD                       lodsw                           ; FSINFO structure sector number
10453 000015F6 8B54DC                   mov     dx,[si-24h]             ; .BPB_RESERVEDSECTORS ; Number of reserved sectors.
10454 000015F9 09C0                     or      ax,ax
10455 000015FB 7404                     jz      short setend8
10456 000015FD 39D0                     cmp     ax,dx
10457 000015FF 7203                     jb      short setend9
10458                                   setend8:
10459 00001601 B8FFFF                   mov     ax,0FFFFh ; -1          ; invalid
10460                                   setend9:
10461 00001604 AB                       stosw                           ; DPB FSINFO structure sector number
10462                                                                   ; [bp+25h]
10463 00001605 AD                       lodsw                           ; Sector number of the backup boot sector
10464 00001606 09C0                     or      ax,ax
10465 00001608 7404                     jz      short setend10
10466 0000160A 39D0                     cmp     ax,dx
10467 0000160C 7203                     jb      short setend11
10468                                   setend10:
10469 0000160E B8FFFF                   mov     ax,0FFFFh ; -1          ; invalid
10470                                   setend11:
10471 00001611 AB                       stosw                           ; DPB backup boot sector address
10472                                                                   ; [bp+27h]
10473 00001612 83C708                   add     di,8                    ; [bp+31h]
10474 00001615 31D2                     xor     dx,dx
10475 00001617 268B45DE                 mov     ax,[es:di-34]           ; [bp+0Fh] ; DPB.MAX_CLUSTER
10476 0000161B 39D0                     cmp     ax,dx
10477 0000161D 7506                     jnz     short setend12          ; > 0 (not FAT32)
10478 0000161F 8B44F0                   mov     ax,[si-16]              ; FAT32 Sectors per FAT ; .BPB_SECTORSPERFAT32
10479 00001622 8B54F2                   mov     dx,[si-14]
10480                                   setend12:
10481 00001625 AB                       stosw                           ; DPB FAT32 FAT size in sectors ; [bp+31h]
10482 00001626 89D0                     mov     ax,dx
10483 00001628 AB                       stosw
10484 00001629 83EE08                   sub     si,8                    ; Root directory cluster number
10485 0000162C A5                       movsw                           ; DPB Root Dir Cluster ; [bp+35h]
10486 0000162D A5                       movsw
10487 0000162E 31C0                     xor     ax,ax                   ; DPB reserved ; [bp+39h]
10488 00001630 AB                       stosw
10489 00001631 AB                       stosw
10490                                   %endif
10491                                       ;;;
10492
10493                                   setend_fat:
10494                                       ; 13/01/2024 - Retro DOS v5.0
10495 00001632 31C0                     xor     ax,ax ; 0
10496
10497                                       ;mov    word [es:bp+1Dh],ax ; 0
10498 00001634 2689461D                 mov     word [es:bp+DPB.NEXT_FREE],ax ; 0
10499                                                                   ; Init so first ALLOC starts at
10500                                                                   ; begining of FAT
10501 00001638 48                       dec     ax  ; -1
10502                                       ;mov    word [es:bp+1Fh],ax ; -1
10503 00001639 2689461F                 mov     word [es:bp+DPB.FREE_CNT],ax ; -1 ; current count is invalid.
10504
10505 0000163D C3                       retn
10506
10507                                   ;EndProc $SETDPB
10508
10509                                   ;BREAK <$Create_Process_Data_Block,SetMem -- Set up process data block>
10510
10511                                   ;
10512                                   ;--------------------------------------------------------------------------
10513                                   ;
10514                                   ;** $Dup_PDB
10515                                   ;
10516                                   ; Inputs:   DX is new segment address of process
10517                                   ;       SI is end of new allocation block
10518                                   ;
10519                                   ;--------------------------------------------------------------------------
10520                                   ;
10521                                       ; 14/01/2024 - Retro DOS 5.0
10522                                       ; PCDOS 7.1 IBMDOS.COM - DOSCODE:554Fh
10523
10524                                   _$DUP_PDB:
10525
```

```
                                ;hkn;       CreatePDB would have a CS override. This is not valid.
                                ;hkn;       Must set up ds in order to acess CreatePDB. Also SS is
                                ;hkn;       has been assumed to be NOTHING. It may not have DOSDATA.

                                    ; MSDOS 3.3
                                    ;MOV    byte [CS:CreatePDB],0FFh  ; indicate a new process
                                    ;MOV    DS,[CS:CurrentPDB]

                                    ; 15/05/2019 - Retro DOS v4.0
                                    ; MSDOS 6.0
0000163E 2E8E1E[0700]               mov     ds,[cs:DosDSeg]
00001643 C606[A803]FF               MOV     byte [CreatePDB],0FFh
00001648 8E1E[3003]                 MOV     DS,[CurrentPDB]

0000164C 56                         PUSH    SI
0000164D EB0A                       JMP     SHORT CreateCopy

                                ;
                                ;--------------------------------------------------------------------------
                                ;
                                ; Inputs:
                                ;   DX = Segment number of new base
                                ; Function:
                                ;   Set up program base and copy term and ^C from int area
                                ; Returns:
                                ;   None
                                ; Called at DOS init
                                ;
                                ;--------------------------------------------------------------------------
                                ;

                                ; 15/05/2019 - Retro DOS v4.0
                                ; DOSCODE:4EB6h (MSDOS 6.21, MSDOS.SYS)

                                ; 06/11/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
                                ; DOSCODE:4EA2h (MSDOS 5.0, MSDOS.SYS)

                                _$CREATE_PROCESS_DATA_BLOCK:
                                                        ; Offset 1D02h in IBMDOS.COM (MSDOS 3.3), 1987
0000164F E825EE                     CALL    Get_User_Stack
                                    ;mov    ds,[si+14h]
00001652 8E5C14                     MOV     DS,[SI+user_env.user_CS]
                                    ;push   word [2]
00001655 FF360200                   PUSH    word [PDB.BLOCK_LEN] ;*
                                CreateCopy:
00001659 8EC2                       MOV     ES,DX

0000165B 31F6                       XOR     SI,SI                   ; copy entire PDB
0000165D 89F7                       MOV     DI,SI
0000165F B98000                     MOV     CX,128
00001662 F3A5                       REP     MOVSW

                                ; DOS 3.3 7/9/86
                                    ;mov    cx,20
                                    ;MOV    CX,FILPERPROC           ; copy handles in case of
                                    ; 15/12/2022
00001664 B114                       mov     cl,FILPERPROC ; 06/07/2019
                                    ; 07/12/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
                                    ;mov    cx,FILPERPROC

                                    ;mov    di,18h
00001666 BF1800                     MOV     DI,PDB.JFN_TABLE        ; Set Handle Count has been issued
                                    ;;PUSH  DS ; * 15/05/2019
                                    ;;lds   si,[34h]
                                    ;LDS    SI,[PDB.JFN_Pointer]
                                    ;REP    MOVSB
                                    ;;POP   DS ; * 15/05/2019
                                    ; 06/11/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
                                    ; 05/12/2022
                                    ; (push ds then pop ds is not needed here!)
                                    ;push   ds
                                    ;lds    si,[34h]
00001669 C5363400                   lds     si,[PDB.JFN_Pointer]
0000166D F3A4                       rep     movsb
                                    ;pop    ds

                                ; DOS 3.3 7/9/86
                                    ;hkn ;CreatePDB would have a CS override. This is not valid.
                                    ;hkn ;Must set up ds in order to access CreatePDB. Also SS is
                                    ;hkn ;has been assumed to be NOTHING. It may not have DOSDATA.

0000166F 2E8E1E[0700]               mov     ds,[cs:DosDSeg] ; 15/05/2019

                                    ;;test  byte [cs:CreatePDB],0FFh
                                    ;cmp    byte [CS:CreatePDB],0 ; Shall we create a process?
                                    ; 17/12/2022
00001674 380E[A803]                 cmp     [CreatePDB],cl ; 0
                                    ;cmp    byte [CreatePDB],0 ; 15/05/2019
00001678 744A                       JZ      short Create_PDB_cont ; nope, old style call

                                ; Here we set up for a new process...

                                    ;PUSH   CS                      ; Called at DOSINIT time, NO SS
                                    ;POP    DS

                                    ; MSDOS 6.0
                                    ;;getdseg <ds>                  ; ds -> dosdata
                                    ;mov    ds,[cs:DosDSeg] ; 15/05/2019
                                    ; 07/12/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
                                    ; (nonsense! but i put this for addr compatibility as temporary)
                                    ; 15/12/2022
                                    ;mov    ds,[cs:DosDSeg] ; 15/05/2019

0000167A 31DB                       XOR     BX,BX                   ; dup all jfns
                                    ;mov    cx,20
                                    ; 06/11/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
                                    ;MOV    CX,FILPERPROC           ; only 20 of them
                                    ; 15/12/2022
0000167C B114                       mov     cl,FILPERPROC ; 06/07/2019

                                Create_dup_jfn:
0000167E 06                         PUSH    ES ;**                  ; save new PDB
0000167F E85160                     call    SFFromHandle            ; get sf pointer
00001682 B0FF                       MOV     AL,-1                   ; unassigned JFN
00001684 7224                       JC      short CreateStash       ; file was not really open
                                    ;;test  word [es:di+5],1000h
                                    ;TEST   word [ES:DI+SF_ENTRY.sf_flags],sf_no_inherit
                                    ; 15/05/2019
                                    ;test   byte [es:di+6],10h
00001686 26F6450610                 test    byte [ES:DI+SF_ENTRY.sf_flags+1],(sf_no_inherit>>8)
0000168B 751D                       JNZ     short CreateStash       ; if no-inherit bit is set, skip dup.

                                ; We do not inherit network file handles.

```

```
                                    ;mov    ah,[es:di+2]
000016BD 268A6502                   MOV     AH,[ES:DI+SF_ENTRY.sf_mode]
                                    ;and    ah,0F0h
00001691 80E4F0                     AND     AH,SHARING_MASK
                                    ;cmp    ah,70h
00001694 80FC70                     CMP     AH,SHARING_NET_FCB
00001697 7411                       jz      short CreateStash

                                ; The handle we have found is duplicatable (and inheritable). Perform
                                ; duplication operation.

00001699 893E[9E05]                 MOV     [THISSFT],DI
0000169D 8C06[A005]                 MOV     [THISSFT+2],ES
000016A1 E8101A                     call    DOS_DUP                 ; signal duplication

                                ; get the old sfn for copy

000016A4 E80F60                     call    pJFNFromHandle          ; ES:DI is jfn
000016A7 268A05                     MOV     AL,[ES:DI]              ; get sfn

                                ; Take AL (old sfn or -1) and stash it into the new position

                                CreateStash:
000016AA 07                         POP     ES ;**
                                    ;mov    [es:bx+18h],al
000016AB 26884718                   MOV     [ES:BX+PDB.JFN_TABLE],AL ; copy into new place!
000016AF 43                         INC     BX                      ; next jfn...
000016B0 E2CC                       LOOP    Create_dup_jfn

000016B2 8B1E[3003]                 MOV     BX,[CurrentPDB]                 ; get current process
                                    ; 06/11/2022
                                    ;mov    [es:16h],bx
000016B6 26891E1600                 MOV     [ES:PDB.PARENT_PID],BX; stash in child
000016BB 8C06[3003]                 MOV     [CurrentPDB],ES
                                    ;MOV    DS,BX ; 28/07/2019
                                    ; 07/12/2022
                                    ;mov    ds,[cs:DosDSeg]
                                    ; 15/12/2022
                                    ; ds = [cs:DosDSeg]
000016BF C606[A803]00               mov     byte [CreatePDB],0    ; reset flag
                                    ;mov    ds,bx
                                    ; 07/12/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
                                    ; 15/12/2022
                                    ;mov    ds,bx

                                ; end of new process create

                                Create_PDB_cont:
                                    ;MOV    BYTE [CS:CreatePDB],0 ; reset flag

                                ;hkn; It comes to this point from 2 places. So, change to DOSDATA temporarily

                                    ;; 28/07/2019
                                    ;;push  ds
                                    ;;mov   ds,[cs:DosDSeg]
                                    ;mov    byte [CreatePDB],0
                                    ;;pop   ds

                                ; 05/12/2022
                                ;   ; 06/11/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
                                ;   ; (push-pop ds is nonsense here!
                                ;   ;  but i am using same code with original MSDOS.SYS
                                ;   ;  for address compatibility.)
                                ;   push    ds
                                ;   ; ds = [cs:DosDSeg] !
                                ;   mov     ds,[cs:DosDSeg]  ; again !
                                ;   mov     byte [CreatePDB],0
                                ;   pop     ds

000016C4 58                         POP     AX ;*

                                    ;entry  SETMEM

                                    ; 17/12/2022
                                    ; cx = 0

                                ;--------------------------------------------------------------------------
                                ; Inputs:
                                ;   AX = Size of memory in paragraphs
                                ;   DX = Segment
                                ; Function:
                                ;   Completely prepares a program base at the
                                ;   specified segment.
                                ; Called at DOS init
                                ; Outputs:
                                ;   DS = DX
                                ;   ES = DX
                                ;   [0] has INT int_abort
                                ;   [2] = First unavailable segment
                                ;   [5] to [9] form a long call to the entry point
                                ;   [10] to [13] have exit address (from int_terminate)
                                ;   [14] to [17] have ctrl-C exit address (from int_ctrl_C)
                                ;   [18] to [21] have fatal error address (from int_fatal_abort)
                                ; DX,BP unchanged. All other registers destroyed.
                                ;--------------------------------------------------------------------------

                                SETMEM:
                                    ;XOR    CX,CX
                                    ; 17/12/2022
                                    ; cx = 0
000016C5 8ED9                       MOV     DS,CX
000016C7 8EC2                       MOV     ES,DX
                                    ;mov    si,88h
000016C9 BE8800                     MOV     SI,addr_int_terminate
                                    ;mov    di,10 ; 0Ah
000016CC BF0A00                     MOV     DI,SAVEXIT
                                    ;MOV    CX,6
                                    ; 15/12/2022
000016CF B106                       mov     cl,6
000016D1 F3A5                       REP     MOVSW
000016D3 26A30200                   MOV     [ES:2],AX
000016D7 29D0                       SUB     AX,DX
000016D9 3DFF0F                     CMP     AX,MAXDIF ; 0FFFh
000016DC 7603                       JBE     short HAVDIF
000016DE B8FF0F                     MOV     AX,MAXDIF
                                HAVDIF:
000016E1 83E810                     SUB     AX,10h                  ; Allow for 100h byte "stack"
000016E4 BB0C00                     MOV     BX,ENTRYPOINTSEG ; 0Ch;       in .COM files
000016E7 29C3                       SUB     BX,AX
000016E9 B104                       MOV     CL,4
000016EB D3E0                       SHL     AX,CL
000016ED 8EDA                       MOV     DS,DX

                                    ; (MSDOS 6.0 note)
```

```
                                    ;
                                    ; The address in BX:AX will be F01D:FEF0 if there is 64K or more
                                    ; memory in the system. This is equivalent to 0:c0 if A20 is OFF.
                                    ; If DOS is in HMA this equivalence is no longer valid as A20 is ON.
                                    ; But the BIOS which now resides in FFFF:30 has 5 bytes in FFFF:D0
                                    ; (F01D:FEF0) which is the same as the ones in 0:C0, thereby
                                    ; making this equvalence valid for this particular case. If however
                                    ; there is less than 64K remaining the address in BX:AX will not
                                    ; be the same as above. We will then stuff 0:c0, the call 5 address
                                    ; into the PSP.
                                    ;
                                    ; Therefore for the case where there is less than 64K remaining in
                                    ; the system old CPM Apps that look at PSP:6 to determine memory
                                    ; requirements will not work. Call 5, however will continue to work
                                    ; for all cases.
                                    ;

                                    ;mov    [6],ax
                                    ;mov    [8],bx

000016EF A30600                     MOV     [PDB.CPM_CALL+1],AX
000016F2 891E0800                   MOV     [PDB.CPM_CALL+3],BX

                                    ; 06/05/2019 - Retro DOS v4.0
000016F6 3DF0FE                     cmp     ax,WRAPOFFSET ; 0FEF0h; Q: does the system have >= 64k of
                                                                ;    memory left
000016F9 740C                       je      short addr_ok       ; Y: the above calculated address is
                                                                ;    OK
                                                                ; N:

000016FB C7060600C000               MOV     WORD [PDB.CPM_CALL+1],0C0h
00001701 C70608000000               MOV     WORD [PDB.CPM_CALL+3],0
                                addr_ok:
                                    ;mov    word [0],20CDh
00001707 C7060000CD20               MOV     word [PDB.EXIT_CALL],(int_abort*256) + mi_INT
                                    ;mov    byte [5],9Ah
0000170D C606050 09A                MOV     BYTE [PDB.CPM_CALL],mi_Long_CALL
                                    ;mov    word [50h],21CDh
00001712 C7065000CD21               MOV     WORD [PDB.CALL_SYSTEM],(int_command*256) + mi_INT
                                    ;mov    byte [52h],0CBh
00001718 C6065200CB                 MOV     BYTE [PDB.CALL_SYSTEM+2],mi_Long_RET
                                    ;mov    word [34h],18h
0000171D C70634001800               MOV     WORD [PDB.JFN_Pointer],PDB.JFN_TABLE
                                    ;mov    word [36h],ds
00001723 8C1E3600                   MOV     WORD [PDB.JFN_Pointer+2],DS
                                    ;mov    word [32h],20
00001727 C70632001400               MOV     WORD [PDB.JFN_Length],FILPERPROC
                                ;
                                ; The server runs several PDB's without creating them VIA EXEC.  We need to
                                ; enumerate all PDB's at CPS time in order to find all references to a
                                ; particular SFT.  We perform this by requiring that the server link together
                                ; for us all sub-PDB's that he creates. The requirement for us, now, is to
                                ; initialize this pointer.
                                ;
                                    ;mov    word [38h],-1
0000172D C7063800FFFF               MOV     word [PDB.Next_PDB],-1
                                    ;mov    word [3Ah],-1
00001733 C7063A00FFFF               MOV     word [PDB.Next_PDB+2],-1

                                    ; 06/05/2019
                                    ; Set the real version number in the PSP - 5.00

                                    ;mov    word [es:PDB.Version],1406h ; MSDOS 6.21 (DOSCODE:4FB6h)
                                    ; 07/12/2022
00001739 26C7064000070A             mov     word [ES:PDB.Version],(MINOR_VERSION*256)+MAJOR_VERSION

00001740 C3                         retn

                                ; 29/04/2019 - Retro DOS v4.0

                                ;BREAK <$GSetMediaID -- get set media ID>

                                ;--------------------------------------------------------------------------
                                ; Inputs:
                                ;   BL= drive number as defined in IOCTL
                                ;   AL= 0 get media ID
                                ;       1 set media ID
                                ;   DS:DX= buffer containing information
                                ;           DW  0  info level (set on input)
                                ;           DD  ?  serial #
                                ;           DB  11 dup(?)  volume id
                                ;           DB   8 dup(?)  file system type
                                ; Function:
                                ;   Get or set media ID
                                ; Returns:
                                ;   carry clear, DS:DX is filled
                                ;   carry set, error
                                ;--------------------------------------------------------------------------

                                    ; 06/11/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)

                                ; 14/01/2024
                                %if 0

                                _$GSetMediaID:
                                    ; RAWIO - GET_MEDIA_ID
                                    mov     cx,0866h            ;AN000;MS.; assume get for IOCTL
                                    cmp     al,0                ;AN001;MS.; get ?
                                    je      short doioctl       ;AN000;MS.; yes
                                    ;cmp    al,1                ;AN000;MS.; set ?
                                    ;jne    short errorfunc             ;AN000;MS.; no
                                    ; 15/12/2022
                                    dec     al
                                    jnz     short errorfunc ; al > 1
                                    ; RAWIO - SET_MEDIA_ID
                                    ;mov    cx,0846h            ;AN001;MS.;
                                    ; 15/12/2022
                                    mov     cl,46h  ; cx = 0846h
                                doioctl:                        ;AN000;
                                    mov     al,0Dh              ;AN000;MS.; generic IOCTL
                                    ;invoke $IOCTL              ;AN000;MS.; let IOCTL take care of it
                                    ;call   _$IOCTL
                                    ;retn                       ;AN000;MS.;
                                    ; 15/12/2022
                                    jmp     _$IOCTL
                                errorfunc:                      ;AN000;
                                    ;error  error_invalid_function;AN000;MS. ; invalid function
                                    ;mov    al,1
                                    mov     al,error_invalid_function
                                    jmp     SYS_RET_ERR

                                %else
                                    ; 14/01/2024 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMDOS.COM)
                                    ; PCDOS 7.1 IBMDOS.COM - DOSCODE:5667h
```

```
10898
10899                                   _$GSetMediaID:
10900 00001741 1E                           push    ds
10901 00001742 56                           push    si
10902 00001743 36C536[A205]                 lds     si,[ss:THISCDS]
10903 00001748 C57445                       lds     si,[si+45h]             ; [si+curdir.devptr]
10904                                                                       ; local pointer to DPB or net device
10905 0000174B 8B740F                       mov     si,[si+0Fh]             ; [si+DPB.FAT_SIZE]
10906 0000174E B96608                       mov     cx,0866h                ; assume get for IOCTL
10907 00001751 3C01                         cmp     al,1                    ; set ?
10908 00001753 7208                         jb      short doioctl1          ; get
10909 00001755 7733                         ja      short errorfunc                 ; invalid
10910 00001757 B146                         mov     cl,46h  ; cx = 0846h
10911                                       ;or     si,si
10912 00001759 21F6                         and     si,si ; 14/01/2024
10913 0000175B 7424                         jz      short doioctl2
10914                                   doioctl1:
10915 0000175D 09F6                         or      si,si
10916 0000175F 7522                         jnz     short doioctl
10917 00001761 1F                           pop     ds
10918 00001762 1E                           push    ds
10919 00001763 89D6                         mov     si,dx   ; disk info
10920                                                       ; .............................
10921                                                       ;
10922                                                       ; 00h    WORD    0000h (info level)
10923                                                       ; 02h    DWORD   disk serial number (binary)
10924                                                       ; 06h 11 BYTEs   volume label or "NO NAME    " if none present
10925                                                       ; 11h  8 BYTEs   (AL=00h only) filesystem type
10926                                                       ;               "FAT12   "
10927                                                       ;               "FAT16   "
10928                                                       ;               "FAT32   " ; PCDOS 7.1
10929                                                       ;               "CDROM   "
10930                                                       ;               "CD001   "
10931                                                       ;               "CDAUDIO "
10932                                                       ; .............................
10933                                                       ; (ref: Ralf Brown's Interrupt List)
10934
10935 00001765 817C114641                   cmp     word [si+11h],4146h ; 'FA'
10936 0000176A 7517                         jne     short doioctl
10937 0000176C 817C135433                   cmp     word [si+13h],3354h ; 'T3'
10938 00001771 7510                         jne     short doioctl
10939 00001773 817C153220                   cmp      word [si+15h],2032h ; '2 '
10940 00001778 7509                         jne     short doioctl
10941 0000177A 817C172020                   cmp     word [si+17h],2020h ; '  '
10942 0000177F 7502                         jne     short doioctl
10943                                   doioctl2:
10944 00001781 B548                         mov     ch,48h  ; cx = 4846h
10945                                   doioctl:
10946 00001783 5E                           pop     si
10947 00001784 1F                           pop     ds
10948 00001785 B00D                         mov     al,0Dh                  ; generic IOCTL
10949                                       ;call   _$IOCTL
10950                                           ;retn
10951 00001787 E9FA10                       jmp     _$IOCTL                 ; let IOCTL take care of it
10952                                   errorfunc:
10953 0000178A B001                         mov     al,1                    ; error_invalid_function
10954 0000178C E9E9EE                       jmp     SYS_RET_ERR
10955
10956                                   %endif
10957
10958                                   ; 16/05/2019 - Retro DOS v4.0
10959
10960                                   ;============================================================================
10961                                   ; MISC2.ASM, MSDOS 6.0, 1991
10962                                   ;============================================================================
10963                                   ; 20/07/2018 - Retro DOS v3.0
10964                                   ; 29/04/2019 - Retro DOS v4.0
10965
10966                                   ; Break <STRCMP - compare two ASCIZ strings DS:SI to ES:DI>
10967                                   ;--------------------------------------------------------------------------
10968                                   ;
10969                                   ;   Strcmp - compare ASCIZ DS:SI to ES:DI. Case INSENSITIVE. '/' = '\'
10970                                   ;        Strings of different lengths don't match.
10971                                   ;   Inputs:  DS:SI - pointer to source string  ES:DI - pointer to dest string
10972                                   ;   Outputs: Z if strings same, NZ if different
10973                                   ;   Registers modified: NONE
10974                                   ;--------------------------------------------------------------------------
10975
10976                                       ; 07/12/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
10977
10978                                       ; 14/01/2024 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMDOS.COM)
10979                                       ; (PCDOS 7.1 IBMDOS.COM DOSCODE:56B6h)
10980                                       ; (MSDOS 6.22 MSDOS.SYS DOSCODE:4FD7h)
10981                                   StrCmp:
10982 0000178F 56                           push    si
10983 00001790 57                           push    di
10984 00001791 50                           push    ax
10985                                   Cmplp:
10986 00001792 AC                           LODSB
10987 00001793 E80146                       call    UCase                   ; convert to upper case
10988 00001796 E85146                       call    PATHCHRCMP              ; convert '/' to '\' ; 07/12/2022 ('\')
10989 00001799 88C4                         MOV     AH,AL
10990 0000179B 268A05                       MOV     AL,[ES:DI]
10991 0000179E 47                           INC     DI
10992 0000179F E8F545                       call    UCase                   ; convert to upper case
10993 000017A2 E84546                       call    PATHCHRCMP              ; convert '/' to '\' ; 07/12/2022 ('\')
10994 000017A5 38C4                         CMP     AH,AL
10995 000017A7 7504                         JNZ     short PopRet            ; Strings dif
10996
10997 000017A9 08C0                         OR      AL,AL
10998 000017AB 75E5                         JNZ     short Cmplp             ; More string
10999                                   PopRet:
11000 000017AD 58                           pop     ax
11001 000017AE 5F                           pop     di
11002 000017AF 5E                           pop     si
11003 000017B0 C3                           retn
11004
11005                                   ;Break <STRCPY - copy ASCIZ string from DS:SI to ES:DI>
11006                                   ;--------------------------------------------------------------------------
11007                                   ;
11008                                   ;   Strcpy - copy an ASCIZ string from DS:SI to ES:DI and make uppercase
11009                                   ;   FStrcpy - copy an ASCIZ string from DS:SI to ES:DI. no modification of
11010                                   ;   characters.
11011                                   ;
11012                                   ;   Inputs: DS:SI - pointer to source string
11013                                   ;           ES:DI - pointer to destination string
11014                                   ;   Outputs:       ES:DI point byte after nul byte at end of dest string
11015                                   ;           DS:SI point byte after nul byte at end of source string
11016                                   ;   Registers modified: SI,DI
11017                                   ;--------------------------------------------------------------------------
11018
11019                                   StrCpy:
11020 000017B1 50                           push    ax
11021                                   CPYLoop:
```

```
11022 000017B2 AC                       LODSB
11023 000017B3 E8E145                   call    UCase                   ; convert to upper case
11024 000017B6 E83146                   call    PATHCHRCMP              ; convert / to \ ;
11025 000017B9 AA                       STOSB
11026
11027 000017BA 08C0                     OR      AL,AL
11028 000017BC 75F4                     JNZ     short CPYLoop
11029 000017BE 58                       pop     ax
11030 000017BF C3                       retn
11031
11032                                 ;--------------------------------------------------------------------------
11033                                 ; Procedure Name : FStrCpy
11034                                 ;--------------------------------------------------------------------------
11035
11036                                 FStrCpy:
11037 000017C0 50                       push    ax
11038                                 FCPYLoop:
11039 000017C1 AC                       LODSB
11040 000017C2 AA                       STOSB
11041 000017C3 08C0                     OR      AL,AL
11042 000017C5 75FA                     JNZ     short FCPYLoop
11043 000017C7 58                       pop     ax
11044 000017C8 C3                       retn
11045
11046                                 ; 20/07/2018 - Retro DOS v3.0
11047                                 ;--------------------------------------------------------------------------
11048                                 ; UCase, IBMDOS.COM (MSDOS 3.3), 1987 - Offset 1E2Fh
11049                                 ;--------------------------------------------------------------------------
11050                                 ;
11051                                 ;UCase:
11052                                 ;   call    _UCase  ; Offset 5518h (GetLet, Offset 5517h)
11053                                 ;   retn
11054
11055                                 ;Break <StrLen - compute length of string ES:DI>
11056                                 ;--------------------------------------------------------------------------
11057                                 ;** StrLen - Compute Length of String
11058                                 ;
11059                                 ;   StrLen computes the length of a string, including the trailing 00
11060                                 ;
11061                                 ;   ENTRY   (es:di) = address of string
11062                                 ;   EXIT    (cx) = size of string
11063                                 ;   USES    cx, flags
11064                                 ;--------------------------------------------------------------------------
11065
11066                                 StrLen:
11067 000017C9 57                       push    di
11068 000017CA 50                       push    ax
11069                                   ;MOV    CX,-1
11070 000017CB B9FFFF                   mov     cx,65535
11071 000017CE 30C0                     XOR     AL,AL
11072 000017D0 F2AE                     REPNE   SCASB
11073 000017D2 F7D1                     NOT     CX
11074 000017D4 58                       pop     ax
11075 000017D5 5F                       pop     di
11076 000017D6 C3                       retn
11077
11078                                 ;--------------------------------------------------------------------------
11079                                 ;** DStrLen - Compute Length of String
11080                                 ;
11081                                 ;   ENTRY   (ds:si) = address of string
11082                                 ;   EXIT    (cx) = size of string, including trailing NUL
11083                                 ;   USES    cx, flags
11084                                 ;--------------------------------------------------------------------------
11085
11086                                 DStrLen:    ; BUGBUG - this guy is a pig, who uses him?
11087 000017D7 E80300                   CALL    XCHGP
11088 000017DA E8ECFF                   CALL    StrLen
11089                                   ;CALL   XCHGP
11090                                   ;retn
11091                                   ; 18/12/2022
11092                                   ;jmp    short XCHGP
11093
11094                                 ;--------------------------------------------------------------------------
11095                                 ;** XCHGP - Exchange Source and Destination Pointers
11096                                 ;
11097                                 ;   XCHGP exchanges (DS:SI) and (ES:DI)
11098                                 ;
11099                                 ;   ENTRY   none
11100                                 ;   EXIT    pairs exchanged
11101                                 ;   USES    SI, DI, DS, ES
11102                                 ;--------------------------------------------------------------------------
11103
11104                                 XCHGP:
11105 000017DD 1E                       push    ds
11106 000017DE 06                       push    es
11107 000017DF 1F                       pop     ds
11108 000017E0 07                       pop     es
11109 000017E1 87F7                     XCHG    SI,DI
11110                                 xchgp_retn:
11111 000017E3 C3                       retn
11112
11113                                 ;Break      <Idle - wait for a specified amount of time>
11114                                 ;--------------------------------------------------------------------------
11115                                 ;
11116                                 ;   Idle - when retrying an operation due to a lock/sharing violation,
11117                                 ;      we spin until RetryLoop is exhausted.
11118                                 ;
11119                                 ;   Inputs: RetryLoop is the number of times we spin
11120                                 ;   Outputs:        Wait
11121                                 ;   Registers modified: none
11122                                 ;--------------------------------------------------------------------------
11123
11124                                 Idle:
11125                                   ;test   byte [SS:FSHARING],0FFh
11126 000017E4 36803E[7205]00           cmp     byte [SS:FSHARING],0  ;hkn; SS override
11127                                   ;retnz
11128 000017EA 75F7                     jnz     short xchgp_retn
11129                                   ;SAVE   <CX>
11130 000017EC 51                       push    cx
11131 000017ED 368B0E[1C00]             MOV     CX,[ss:RetryLoop]     ;hkn; SS override
11132 000017F2 E308                     JCXZ    Idle3
11133                                 Idle1:
11134 000017F4 51                       PUSH    CX
11135 000017F5 31C9                     XOR     CX,CX
11136                                 Idle2:
11137 000017F7 E2FE                     LOOP    Idle2
11138 000017F9 59                       POP     CX
11139 000017FA E2F8                     LOOP    Idle1
11140                                 Idle3:
11141                                   ;RESTORE <CX>
11142 000017FC 59                       pop     cx
11143 000017FD C3                       retn
11144
11145                                 ;Break      <TableDispatch - dispatch to a table>
```

```
11146                                   ;--------------------------------------------------------------------------
11147                                   ;
11148                                   ;   TableDispatch - given a table and an index, jmp to the approptiate
11149                                   ;   routine. Preserve all input registers to the routine.
11150                                   ;
11151                                   ;   Inputs: Push    return address
11152                                   ;           Push    Table address
11153                                   ;           Push    index (byte)
11154                                   ;   Outputs:       appropriate routine gets jumped to.
11155                                   ;           return indicates invalid index
11156                                   ;   Registers modified: none.
11157                                   ;--------------------------------------------------------------------------
11158
11159                                   struc TFrame        ; TableFrame
11160 00000000 ????                    .OldBP:     resw 1  ; 0
11161 00000002 ????                    .OldRet: resw 1  ; 2
11162 00000004 ??                      .Index:     resb 1  ; 4
11163 00000005 ??                      .Pad:       resb 1  ; 5
11164 00000006 ????                    .Tab:       resw 1  ; 6
11165 00000008 ????                    .NewRet: resw 1  ; 8
11166                                   endstruc
11167
11168                                   TableDispatch:
11169 000017FE 55                           PUSH    BP
11170 000017FF 89E5                         MOV     BP,SP
11171 00001801 53                           PUSH    BX                      ; save BX
11172                                       ;mov    bx,[bp+6]
11173 00001802 8B5E06                       MOV     BX,[BP+TFrame.Tab]      ; get pointer to table
11174 00001805 2E8A1F                       MOV     BL,[CS:BX]              ; maximum index
11175                                       ;cmp    [bp+4],bl
11176 00001808 385E04                       CMP     [BP+TFrame.Index],BL    ; table error?
11177 0000180B 7317                         JAE     short TableError        ; yes
11178                                       ;mov    bl,[bp+4]
11179 0000180D 8A5E04                       MOV     BL,[BP+TFrame.Index]    ; get desired table index
11180 00001810 30FF                         XOR     BH,BH                   ; convert to word
11181 00001812 D1E3                         SHL     BX,1                    ; convert to word pointer
11182 00001814 43                           INC     BX                      ; point past first length byte
11183                                       ; 17/08/2018
11184                                       ;add    bx,[bp+6]
11185 00001815 035E06                       ADD     BX,[BP+TFrame.Tab]      ; get real offset
11186 00001818 2E8B1F                       MOV     BX,[CS:BX]              ; get contents of table entry
11187                                       ;mov    [bp+6],bx
11188 0000181B 895E06                       MOV     [BP+TFrame.Tab],BX      ; put table entry into return address
11189 0000181E 5B                           POP     BX                      ; restore BX
11190 0000181F 5D                           POP     BP                      ; restore BP
11191 00001820 83C404                       ADD     SP,4                    ; clean off Index and our return addr
11192 00001823 C3                           retn                            ; do operation
11193                                   TableError:
11194 00001824 5B                           POP     BX                      ; restore BX
11195 00001825 5D                           POP     BP                      ; restore BP
11196 00001826 C20600                       RETN    6                       ; clean off Index, Table and RetAddr
11197
11198                                   ;Break      <TestNet - determine if a CDS is for the network>
11199                                   ;--------------------------------------------------------------------------
11200                                   ;
11201                                   ;   TestNet - examine CDS pointed to by ThisCDS and see if it indicates a
11202                                   ;   network CDS. This will handle NULL cds also.
11203                                   ;
11204                                   ;   Inputs: ThisCDS points to CDS or NULL
11205                                   ;   Outputs:       ES:DI = ThisCDS
11206                                   ;           carry Set => network
11207                                   ;           carry Clear => local
11208                                   ;   Registers modified: none.
11209                                   ;--------------------------------------------------------------------------
11210
11211                                   TestNet:
11212                                       ;LES    DI,[CS:THISCDS]
11213
11214                                       ; 16/05/2019 - Retro DOS v4.0
11215 00001829 2E8E06[0700]                mov     es,[cs:DosDSeg]
11216 0000182E 26C43E[A205]                LES     DI,[ES:THISCDS]
11217 00001833 83FFFF                      CMP     DI,-1
11218 00001836 7408                        JZ      short CMCRet            ; UNC? carry is clear
11219                                      ;;test  word [es:di+43h],8000h
11220                                      ;TEST   word [ES:DI+curdir.flags],curdir_isnet
11221                                      ;test   byte [es:di+44h],80h
11222 00001838 26F6454480                  TEST    byte [ES:DI+curdir.flags+1],(curdir_isnet>>8)
11223 0000183D 7501                        JNZ     short CMCRet            ; jump has carry clear
11224 0000183F C3                          retn                            ; carry is clear
11225                                   CMCRet:
11226 00001840 F5                          CMC
11227 00001841 C3                          retn
11228
11229                                   ;Break      <IsSFTNet - see if an sft is for the network>
11230                                   ;--------------------------------------------------------------------------
11231                                   ;
11232                                   ;   IsSFTNet - examine SF pointed to by ES:DI and see if it indicates a
11233                                   ;   network file.
11234                                   ;
11235                                   ;   Inputs: ES:DI point to SFT
11236                                   ;   Outputs:       Zero set if not network sft
11237                                   ;           zero reset otherwise
11238                                   ;           Carry CLEAR!!!
11239                                   ;   Registers modified: none.
11240                                   ;--------------------------------------------------------------------------
11241
11242                                   IsSFTNet:
11243                                       ;;test  word [es:di+5],8000h
11244                                       ;TEST   word [ES:DI+SF_ENTRY.sf_flags],sf_isnet
11245                                       ; 16/05/2019
11246                                       ;test   byte [es:di+6],80h
11247 00001842 26F6450680                  TEST    byte [ES:DI+SF_ENTRY.sf_flags+1],(sf_isnet>>8)
11248 00001847 C3                          retn
11249
11250                                   ;Break      <FastInit - Initialize FastTable entries >
11251                                   ;--------------------------------------------------------------------------
11252                                   ;   DOS 4.00   2/9/87
11253                                   ;   FastInit  - initialize the FASTXXX routine entry
11254                                   ;             in the FastTable
11255                                   ;
11256                                   ;   Inputs: BX = FASTXXX ID ( 1=fastopen )
11257                                   ;           DS:SI = address of FASTXXX routine entry
11258                                   ;              SI = -1 for query only
11259                                   ;   Outputs:       Carry flag clear, if success
11260                                   ;           Carry flag set,   if failure
11261                                   ;
11262                                   ;
11263                                   ;--------------------------------------------------------------------------
11264
11265                                   ;Procedure FastInit,NEAR
11266                                   ;   ASSUME  CS:DOSCODE,SS:NOTHING
11267
11268                                   ;   ; MSDOS 3.3
11269                                   ;   ; IBMDOS.COM (1987) - Offset 1EB3h
```

```
11270                                   ;FastInit:
11271                                   ;   mov     di,FastTable ; FastOpenTable
11272                                   ;   mov     ax,[cs:di+4]          ; Entry segment
11273                                   ;   mov     bx,cs                 ; get DOS segment
11274                                   ;   cmp     ax,bx                 ; first time installed ?
11275                                   ;   je      short ok_install      ; yes
11276                                   ;   stc                           ; set carry
11277                                   ;   retn                          ; (cf=1 means) already installed !
11278                                   ;
11279                                   ;ok_install:
11280                                   ;   mov     bx,FastTable ; FastOpenTable
11281                                   ;   mov     cx,ds
11282                                   ;   ; set address of FASTXXX (FASTOPEN) routine entry
11283                                   ;   mov     [cs:bx+4],cx
11284                                   ;   mov     [cs:bx+2],si
11285                                   ;   retn
11286
11287                                   ; 16/05/2019 - Retro DOS v4.0
11288
11289                                   ; 14/01/2024 - Retro DOS v5.0
11290                                   ; PCDOS 7.1 IBMDOS.COM - DOSCODE:5773h
11291
11292                                   FastInit:
11293                                       ; MSDOS 6.0
11294                                       ;hkn; set up es to dosdataseg.
11295 00001848 06                        push    es
11296                                       ;getdseg <es>                 ; es -> dosdata
11297 00001849 2E8E06[0700]               mov     es,[cs:DosDSeg]
11298
11299                                       ;hkn; FastTable is in DOSDATA
11300 0000184E BF[3E11]                   MOV     DI,FastTable+2        ;AN000;FO. points to fastxxx entry
11301 00001851 4B                         DEC     BX                    ;AN000;FO.;; decrement index
11302 00001852 89DA                       MOV     DX,BX                 ;AN000;FO.;; save bx
11303 00001854 D1E3                       SHL     BX,1                  ;AN000;FO.;; times 4, each entry is DWORD
11304 00001856 D1E3                       SHL     BX,1                  ;AN000;FO.
11305 00001858 01DF                       ADD     DI,BX                 ;AN000;FO. index to the entry
11306 0000185A 268B4502                   MOV     AX,[ES:DI+2]          ;AN000;FO. get entry segment
11307                                   fcheck:                         ;AN000;
11308 0000185E 8CC9                       MOV     CX,CS                 ;AN000;FO.;; get DOS segment
11309 00001860 39C8                       CMP     AX,CX                 ;AN000;FO.;; first time installed ?
11310 00001862 7405                       JZ      short ok_install      ;AN000;FO.;; yes
11311 00001864 09C0                       OR      AX,AX                 ;AN000;FO.;
11312                                     ;JZ     short ok_install      ;AN000;FO.;
11313                                     ;STC                          ;AN000;FO.;; already installed !
11314                                     ;JMP    SHORT FSret           ;AN000;FO. set carry
11315                                     ; 14/01/2024
11316 00001866 F9                         stc
11317 00001867 7517                       jnz     short FSret
11318                                   ok_install:                     ;AN000;
11319 00001869 83FEFF                     CMP     SI,-1                 ;AN000;FO.; Query only ?
11320 0000186C 7412                       JZ      short FSret           ;AN000;FO.; yes
11321 0000186E 8CD9                       MOV     CX,DS                 ;AN000;FO.; get FASTXXX entry segment
11322 00001870 26894D02                   MOV     [ES:DI+2],CX          ;AN000;FO.; initialize routine entry
11323 00001874 268935                     MOV     [ES:DI],SI            ;AN000;FO.; initialize routine offset
11324
11325                                   ;hkn; FastFlg moved to DOSDATA
11326 00001877 BF[4611]                   MOV     DI,FastFlg            ;AN000;FO.; get addr of FASTXXX flags
11327 0000187A 01D7                       ADD     DI,DX                 ;AN000;FO.; index to a FASTXXX flag
11328                                     ;or     byte [es:di],80h
11329 0000187C 26800D80                   OR      byte [ES:DI],Fast_yes ;AN000;FO.; indicate installed
11330                                   FSret:                                  ;AN000;
11331 00001880 07                         pop     es
11332 00001881 C3                         retn                          ;AN000;FO.
11333
11334                                   ;EndProc FastInit
11335
11336                                   ;Break      <FastRet - initial routine in FastOpenTable >
11337                                   ;-----------------------------------------------------------------------------
11338                                   ;   DOS 3.3   6/10/86
11339                                   ;   FastRet - indicate FASTXXXX  not in memory
11340                                   ;
11341                                   ;   Inputs: None
11342                                   ;   Outputs:       AX = -1 and carry flag set
11343                                   ;
11344                                   ;   Registers modified: none.
11345                                   ;-----------------------------------------------------------------------------
11346
11347                                   FastRet:
11348                                       ;mov    ax,-1
11349                                       ;stc
11350                                       ;retf
11351 00001882 F9                         STC
11352 00001883 19C0                       sbb     ax,ax          ; (ax) = -1, 'C' set
11353 00001885 CB                         RETF
11354
11355                                   ;Break      <NLS_OPEN - do $open for NLSFUNC>
11356                                   ;-----------------------------------------------------------------------------
11357                                   ;   DOS 3.3   6/10/86
11358                                   ;   NLS_OPEN       - call $OPEN for NLSFUNC
11359                                   ;
11360                                   ;   Inputs: Same input as $OPEN except CL = mode
11361                                   ;   Outputs:       same output as $OPEN
11362                                   ;
11363                                   ;-----------------------------------------------------------------------------
11364
11365                                   ;hkn; NOTE! SS MUST HAVE BEEN SET UP TO DOSDATA BY THE TIME THESE
11366                                   ;hkn; NLS FUNCTIONS ARE CALLED!!! THERE FORE WE WILL USE SS OVERRIDES
11367                                   ;hkn; IN ORDER TO ACCESS DOS DATA VARIABLES!
11368
11369                                   NLS_OPEN:
11370                                   ;   MOV     BL,[CPSWFLAG]   ; disable code page matching logic
11371                                   ;   MOV     BYTE [CPSWFLAG],0
11372                                   ;   PUSH    BX              ; save current state
11373
11374 00001886 88C8                       MOV     AL,CL           ; set up correct interface for $OPEN
11375 00001888 E83E67                     call    _$OPEN
11376
11377                                   ;   POP     BX              ; restore current state
11378                                   ;   MOV     [CPSWFLAG],BL
11379
11380 0000188B C3                         RETN
11381
11382                                   ;Break      <NLS_LSEEK - do $LSEEK for NLSFUNC>
11383                                   ;-----------------------------------------------------------------------------
11384                                   ;   DOS 3.3   6/10/86
11385                                   ;   NLS_LSEEK      - call $LSEEK for NLSFUNC
11386                                   ;
11387                                   ;   Inputs: BP = open mode
11388                                   ;   Outputs:       same output as $LSEEK
11389                                   ;
11390                                   ;-----------------------------------------------------------------------------
11391
11392                                   ; 16/05/2019 - Retro DOS v4.0
11393
```

```
11394                                   NLS_LSEEK:
11395 0000188C 36FF36[8405]                   PUSH    word [SS:USER_SP] ; save user stack
11396 00001891 36FF36[8605]                   PUSH    word [SS:USER_SS]
11397 00001896 E81000                         CALL    Fake_User_Stack
11398 00001899 89E8                           MOV     AX,BP           ; set up correct interface for $LSEEK
11399 0000189B E83860                         call    _$LSEEK
11400                                   NLS_SEEK_RET:       ; 26/06/2024
11401 0000189E 368F06[8605]                   POP     word [SS:USER_SS] ; restore user stack
11402 000018A3 368F06[8405]                   POP     word [SS:USER_SP]
11403 000018A8 C3                             RETN
11404
11405                                   ;Break      <Fake_User_Stack - save user stack>
11406                                   ;--------------------------------------------------------------------------
11407                                   ;   DOS 3.3   6/10/86
11408                                   ;   Fake_User_Stack - save user stack pointer
11409                                   ;
11410                                   ;--------------------------------------------------------------------------
11411
11412                                   Fake_User_Stack:
11413 000018A9 36A1[9E0D]                     MOV     AX,[SS:USER_SP_2F] ; replace with INT 2Fh stack
11414 000018AD 36A3[8405]                     MOV     [SS:USER_SP],AX
11415 000018B1 8CD0                           MOV     AX,SS
11416 000018B3 36A3[8605]                     MOV     [SS:USER_SS],AX
11417 000018B7 C3                             RETN
11418
11419                                   ;Break      <GetDevList - get device header list pointer>
11420                                   ;--------------------------------------------------------------------------
11421                                   ;   DOS 3.3   7/25/86
11422                                   ;   GetDevList - get device header list pointer
11423                                   ;
11424                                   ;   Output: AX:BX points to the device header list
11425                                   ;--------------------------------------------------------------------------
11426
11427                                   GetDevList:
11428                                       ; 16/05/2019 - Retro DOS v4.0
11429 000018B8 BE[580D]                       MOV     SI,SysInitTable
11430 000018BB 2E8E1E[0700]                   mov     ds,[cs:DosDSeg]
11431 000018C0 C534                           LDS     SI,[SI]
11432                                         ;mov    ax,[si+34]  ; SSYSINITVARS offset 34 = [SI+SYSI.DEV]
11433 000018C2 8B4422                         MOV     AX,[SI+SYSI.DEV]
11434                                         ;mov    bx,[si+36]  ; SSYSINITVARS offset 36 = [SI+SYSI.DEV+2]
11435 000018C5 8B5C24                         MOV     BX,[SI+SYSI.DEV+2]
11436 000018C8 C3                             RETN
11437
11438                                   ;Break      <NLS_IOCTL - do $IOCTL for NLSFUNC>
11439                                   ;--------------------------------------------------------------------------
11440                                   ;   DOS 3.3   7/25/86
11441                                   ;   NLS_IOCTL       - call $IOCTL for NLSFUNC
11442                                   ;
11443                                   ;   Inputs: BP = function code 0CH
11444                                   ;   Outputs:        same output as generic $IOCTL
11445                                   ;
11446                                   ;--------------------------------------------------------------------------
11447
11448                                   NLS_IOCTL:
11449                                       ; 16/05/2019 - Retro DOS v4.0
11450 000018C9 36FF36[8405]                   PUSH    word [SS:USER_SP] ; save user stack
11451 000018CE 36FF36[8605]                   PUSH    word [SS:USER_SS]
11452 000018D3 E8D3FF                         CALL    Fake_User_Stack
11453 000018D6 89E8                           MOV     AX,BP           ; set up correct interface for $IOCTL
11454 000018D8 E8A90F                         call    _$IOCTL
11455                                         ;POP    word [SS:USER_SS] ; restore user stack
11456                                         ;POP    word [SS:USER_SP]
11457                                         ;RETN
11458                                         ; 26/06/2024 - Retro DOS v5.0 (PCDOS 7.1 IBMDOS.COM)
11459 000018DB EBC1                           jmp     short NLS_SEEK_RET
11460
11461                                   ;Break      <NLS_GETEXT- get extended error for NLSFUNC>
11462                                   ;--------------------------------------------------------------------------
11463                                   ;   DOS 3.3   7/25/86
11464                                   ;   NLS_GETEXT      -
11465                                   ;
11466                                   ;   Inputs: none
11467                                   ;   Outputs:        AX = extended error
11468                                   ;
11469                                   ;--------------------------------------------------------------------------
11470
11471                                   NLS_GETEXT:
11472                                       ; 16/05/2019 - Retro DOS v4.0
11473 000018DD 36A1[2403]                     MOV     AX,[SS:EXTERR]  ; return extended error
11474                                         ; 23/09/2023
11475                                   MSG_RETRIEVAL:
11476 000018E1 C3                             RETN
11477
11478                                   ; 29/04/2019 - Retro DOS v4.0
11479
11480                                   ;Break      <MSG_RETRIEVAL- get beginning addr of system and parser messages>
11481
11482                                   ;--------------------------------------------------------------------------
11483                                   ;   DOS 4.00
11484                                   ;
11485                                   ;   Inputs: DL=0 get extended error message addr
11486                                   ;            =1 set extended error message addr
11487                                   ;            =2 get parser error message addr
11488                                   ;            =3 set parser error message addr
11489                                   ;            =4 get critical error message addr
11490                                   ;            =5 set critical error message addr
11491                                   ;            =6 get file system error message addr
11492                                   ;            =7 set file system error message addr
11493                                   ;            =8 get address for code reduction
11494                                   ;            =9 set address for code reduction
11495                                   ;   Function:      get/set message address
11496                                   ;   Outputs:       ES:DI points to addr when get
11497                                   ;--------------------------------------------------------------------------
11498
11499                                   ;Procedure MSG_RETRIEVAL,NEAR
11500                                   ;   ASSUME  CS:DOSCODE,SS:NOTHING
11501
11502                                   ; 23/09/2023
11503                                   ;MSG_RETRIEVAL:
11504
11505                                   ;;  NOTE:  This function lives in command.com resident code now.
11506                                   ;;  If the int 2F ever gets this far, we'll return registers
11507                                   ;;  unchanged, which produces the same result as before, if
11508                                   ;;  command.com wasn't present (and therefore no messages available).
11509                                   ;;
11510                                   ;;  I didn't point the entry in the 2F table to No_Op because
11511                                   ;;  No_Op zeroes AL.
11512                                   ;;
11513                                   ;;;hkn; set up ds to point to DOSDATA
11514                                   ;;  push    ds
11515                                   ;;  getdseg <ds>                    ; ds -> dosdata
11516                                   ;;
11517                                   ;;  PUSH    AX                 ;AN000;;MS. save regs
```

```
11518                                 ;;  PUSH    SI                ;AN000;;MS. save regs
11519                                 ;;  MOV     AX,DX             ;AN000;;MS.
11520                                 ;;  MOV     SI,OFFSET DOSDATA:MSG_EXTERROR ;AN000;;MS.
11521                                 ;;  test    AL,1              ;AN000;;MS. get ?
11522                                 ;;  JZ      toget             ;AN000;;MS. yes
11523                                 ;;  DEC     AL                ;AN000;;MS.
11524                                 ;;toget:                        ;AN000;
11525                                 ;;  SHL     AL,1              ;AN000;;MS. times 2
11526                                 ;;  XOR     AH,AH             ;AN000;;MS.
11527                                 ;;  ADD     SI,AX             ;AN000;;MS. position to the entry
11528                                 ;;  test    DL,1              ;AN000;;MS. get ?
11529                                 ;;  JZ      getget                  ;AN000;;MS. yes
11530                                 ;;  MOV     WORD PTR DS:[SI],DI    ;AN000;;MS. set MSG
11531                                 ;;  MOV     WORD PTR DS:[SI+2],ES  ;AN000;;MS. address to ES:DI
11532                                 ;;  JMP     SHORT MSGret            ;AN000;;MS. exit
11533                                 ;;getget:                       ;AN000;
11534                                 ;;  LES     DI,DWORD PTR DS:[SI]    ;AN000;;MS. get msg addr
11535                                 ;;MSGret:                       ;AN000;
11536                                 ;;  POP     SI                      ;AN000;;MS.
11537                                 ;;  POP     AX                      ;AN000;;MS.
11538                                 ;;
11539                                 ;;  pop     ds
11540
11541                                 ;   return                          ;AN000;;MS. exit
11542
11543                                 ; 23/09/2023
11544                                 ;   retn    ; 29/04/2019
11545
11546                                 ;============================================================================
11547                                 ; ECritDisk, LCritDisk, ECritDevice, LCritDevice
11548                                 ; IBMDOS.COM (MSDOS 3.3), 1987 - Offset 1F36h
11549                                 ;============================================================================
11550                                 ; 20/07/2018 - Retro DOS v3.0
11551
11552                                 ;   ; MSDOS 3.3
11553                                 ;   ; 08/08/2018 - Retro DOS v3.0
11554                                 ;ECritMEM:
11555                                 ;ECritSFT:
11556                                 ;   ;
11557                                 ;ECritDisk:
11558                                 ;   retn
11559                                 ;   ;push  ax
11560                                 ;
11561                                 ;   mov     ax,8001h
11562                                 ;   int     2Ah     ; Microsoft Networks - BEGIN DOS CRITICAL SECTION
11563                                 ;                   ; AL = critical section number (00h-0Fh)
11564                                 ;   pop     ax
11565                                 ;   retn
11566                                 ;
11567                                 ;   ; MSDOS 3.3
11568                                 ;   ; 08/08/2018 - Retro DOS v3.0
11569                                 ;LCritMEM:
11570                                 ;LCritSFT:
11571                                 ;   ;
11572                                 ;LCritDisk:
11573                                 ;   retn
11574                                 ;   ;push  ax
11575                                 ;
11576                                 ;   mov     ax,8101h
11577                                 ;   int     2Ah     ; Microsoft Networks - END DOS CRITICAL SECTION
11578                                 ;                   ; AL = critical section number (00h-0Fh)
11579                                 ;   pop     ax
11580                                 ;   retn
11581                                 ;
11582                                 ;ECritDevice:
11583                                 ;   retn
11584                                 ;   ;push  ax
11585                                 ;
11586                                 ;   mov     ax,8002h
11587                                 ;   int     2Ah     ; Microsoft Networks - BEGIN DOS CRITICAL SECTION
11588                                 ;                   ; AL = critical section number (00h-0Fh)
11589                                 ;   pop     ax
11590                                 ;   retn
11591                                 ;
11592                                 ;LCritDevice:
11593                                 ;   retn
11594                                 ;   ;push  ax
11595                                 ;
11596                                 ;   mov     ax,8102h
11597                                 ;   int     2Ah     ; Microsoft Networks - END DOS CRITICAL SECTION
11598                                 ;                   ; AL = critical section number (00h-0Fh)
11599                                 ;   pop     ax
11600                                 ;   retn
11601
11602                                 ;============================================================================
11603                                 ; CRIT.ASM, MSDOS 6.0, 1991
11604                                 ;============================================================================
11605                                 ; 12/05/2019 - Retro DOS v4.0
11606
11607                                 ; Critical Section Routines
11608
11609                                 ; MSDOS 6.21 - MSDOS.SYS - DOSCODE:513Ah
11610
11611                                 ; 06/11/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
11612                                 ; DOSCODE:5126h (MSDOS 5.0 MSDOS.SYS)
11613
11614                                 ; ---------------------------------------------------------------------------
11615                                 ; Each handler must leave everything untouched; including flags!
11616                                 ;
11617                                 ; Sleaze for time savings: first instruction is a return. This is patched
11618                                 ; by the sharer to be a PUSH AX to complete the correct routines.
11619                                 ; ---------------------------------------------------------------------------
11620
11621                                 ; (DOSMAC.INC, MSDOS 6.0, 1991)
11622                                 ; ---------------------------------------------------------------------------
11623                                 ; Some old versions of the 80286 have a bug in the chip. The popf instruction
11624                                 ; will enable interrupts. Therefore in a section of code with interrupts
11625                                 ; disabled and you need a popf instruction use the 'popff' macro instead.
11626                                 ; ---------------------------------------------------------------------------
11627
11628                                 ;%macro POPFF 0
11629                                 ;   jmp     $+3
11630                                 ;   iret
11631                                 ;   push    cs
11632                                 ;   call    $-2
11633                                 ;%endmacro
11634
11635                                 ; ----------------------------
11636
11637                                 ; 14/01/2024 - Retro DOS v5.0
11638                                 %if 0
11639
11640                                 ;Procedure  ECritDisk,NEAR
11641                                     ;public  ECritMEM
```

```
11642                                   ;public  ECritSFT
11643                                 ECritMEM:
11644                                 ECritSFT:
11645                                 ;
11646                                 ECritDisk:
11647
11648                                 ;SR; Check if critical section is to be entered
11649
11650                                     pushf
11651                                     cmp     byte [ss:redir_patch],0
11652                                     jz      short ECritDisk_2
11653
11654                                 ; 06/11/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
11655                                 ;   ;popff  ; * (macro)
11656                                 ;   jmp     short ECritDisk_1 ; *
11657                                 ;
11658                                 ;ECritDisk_iret: ; *
11659                                 ;   iret ; *
11660
11661                                     ; 16/12/2022
11662                                     ; 13/11/2022
11663                                     ;jmp    short ECritDisk_1
11664                                     ; 06/11/2022
11665                                 ;ECritDisk_iret:
11666                                 ;   iret
11667
11668                                     ; 06/12/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
11669                                 ECritDisk_1:
11670                                     push    cs ; *
11671                                     call    ECritDisk_iret ; *
11672
11673                                 ECritDisk_0:
11674                                     PUSH     AX
11675                                     ;MOV    AX,8000h+critDisk
11676                                     ;INT    int_IBM
11677                                     mov     ax,8001h
11678                                     int     2Ah     ; Microsoft Networks - BEGIN DOS CRITICAL SECTION
11679                                                     ; AL = critical section number (00h-0Fh)
11680                                     POP     AX
11681                                     retn
11682
11683                                     ; 16/12/2022
11684                                     ; 13/11/2022
11685                                 ECritDisk_iret:  ; 12/05/2019 - Retro DOS v4.0
11686                                 LCritDisk_iret:
11687                                     iret
11688
11689                                 ECritDisk_2:
11690                                     ;;popff ; *
11691                                     ;;retn
11692                                 ;   jmp     short ECritDisk_3 ; *
11693                                 ;ECritDisk_iret2: ; *
11694                                 ;   iret
11695
11696                                     ; 16/12/2022
11697                                     ; 13/11/2022
11698                                     ;jmp    short ECritDisk_3
11699                                 ;ECritDisk_iret2:
11700                                     ;iret
11701
11702                                 ECritDisk_3:
11703                                     push    cs ; *
11704                                     ; 13/11/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
11705                                     ;call   ECritDisk_iret2 ; *
11706                                     ;retn
11707                                     ; 16/12/2022
11708                                     call    ECritDisk_iret
11709                                     retn
11710
11711                                 ;EndProc ECritDisk
11712
11713                                 ; ---------------------------
11714
11715                                 ;Procedure   LCritDisk,NEAR
11716                                     ;public  LCritMEM
11717                                     ;public  LCritSFT
11718                                 LCritMEM:
11719                                 LCritSFT:
11720                                 ;
11721                                 LCritDisk:
11722
11723                                 ;SR; Check if critical section is to be entered
11724
11725                                     pushf
11726                                     cmp     byte [ss:redir_patch],0
11727                                     jz      short LCritDisk_2
11728                                     ;popff  ; * (macro)
11729                                 ;   jmp     short LCritDisk_1 ; *
11730                                 ;
11731                                 ;LCritDisk_iret: ; *
11732                                 ;   iret ; *
11733
11734                                     ; 16/12/2022
11735                                     ; 13/11/2022
11736                                     ;jmp    short LCritDisk_1
11737                                 ;LCritDisk_iret:
11738                                     ;iret
11739
11740                                 LCritDisk_1:
11741                                     push    cs ; *
11742                                     call    LCritDisk_iret ; *
11743
11744                                 LCritDisk_0:
11745                                     PUSH    AX
11746                                     ;MOV    AX,8100h+critDisk
11747                                     ;INT    int_IBM
11748                                     mov     ax,8101h
11749                                     int     2Ah     ; Microsoft Networks - END DOS CRITICAL SECTION
11750                                                     ; AL = critical section number (00h-0Fh)
11751                                     POP     AX
11752                                     retn
11753
11754                                 ;LCritDisk_iret:  ; 12/05/2019 - Retro DOS v4.0
11755                                 ;   iret
11756
11757                                 LCritDisk_2:
11758                                     ;;popff ; *
11759                                     ;;retn
11760                                 ;   jmp     short LCritDisk_3 ; *
11761                                 ;LCritDisk_iret2: ; *
11762                                 ;   iret
11763
11764                                     ; 16/12/2022
11765                                     ; 13/11/2022
```

```
11766                                       ;jmp    short LCritDisk_3
11767                                   ;LCritDisk_iret2:
11768                                       ;iret
11769
11770                                   LCritDisk_3:
11771                                       push    cs ; *
11772                                       ; 13/11/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
11773                                       ;call   LCritDisk_iret2 ; *
11774                                       ;retn
11775                                       ; 16/12/2022
11776                                       call    LCritDisk_iret
11777                                       retn
11778
11779                                   ;EndProc LCritDisk
11780
11781                                   ; ---------------------------
11782
11783                                   ;Procedure   ECritDevice,NEAR
11784
11785                                   ECritDevice:
11786
11787                                   ;SR; Check if critical section is to be entered
11788
11789                                       pushf
11790                                       cmp     byte [ss:redir_patch],0
11791                                       jz      short ECritDevice_2
11792                                       ;popff  ; * (macro)
11793                                   ;   jmp     short ECritDevice_1 ; *
11794                                   ;
11795                                   ;ECritDevice_iret: ; *
11796                                   ;   iret ; *
11797
11798                                       ; 16/12/2022
11799                                       ; 13/11/2022
11800                                       ;jmp    short ECritDevice_1
11801                                   ;ECritDevice_iret:
11802                                       ;iret
11803
11804                                   ECritDevice_1:
11805                                       push    cs ; *
11806                                       call    ECritDevice_iret ; *
11807
11808                                   ECritDevice_0:
11809                                       PUSH    AX
11810                                       ;MOV    AX,8000h+critDevice
11811                                       ;INT    int_IBM
11812                                       mov     ax,8002h
11813                                       int     2Ah     ; Microsoft Networks - BEGIN DOS CRITICAL SECTION
11814                                                       ; AL = critical section number (00h-0Fh)
11815                                       POP     AX
11816                                       retn
11817
11818                                       ; 16/12/2022
11819                                       ; 06/12/2022
11820                                   ECritDevice_iret:  ; 12/05/2019 - Retro DOS v4.0
11821                                   LCritDevice_iret:
11822                                       iret
11823
11824                                   ECritDevice_2:
11825                                       ;;popff ; *
11826                                       ;;retn
11827                                   ;   jmp     short ECritDevice_3 ; *
11828                                   ;ECritDevice_iret2: ; *
11829                                   ;   iret
11830
11831                                       ; 16/12/2022
11832                                       ; 13/11/2022
11833                                       ;jmp    short ECritDevice_3
11834                                   ;ECritDevice_iret2:
11835                                       ;iret
11836
11837                                   ECritDevice_3:
11838                                       push    cs ; *
11839                                       ; 13/11/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
11840                                       ;call   ECritDevice_iret2 ; *
11841                                       ;retn
11842                                       ; 16/12/2022
11843                                       call    ECritDevice_iret
11844                                       retn
11845
11846                                   ;EndProc ECritDevice
11847
11848                                   ; ---------------------------
11849
11850                                   ;Procedure   LCritDevice,NEAR
11851
11852                                   LCritDevice:
11853
11854                                   ;SR; Check if critical section is to be entered
11855
11856                                       pushf
11857                                       cmp     byte [ss:redir_patch],0
11858                                       jz      short LCritDevice_2
11859                                       ;popff  ; * (macro)
11860                                   ;   jmp     short LCritDevice_1 ; *
11861                                   ;
11862                                   ;LCritDevice_iret: ; *
11863                                   ;   iret ; *
11864
11865                                       ; 16/12/2022
11866                                       ; 13/11/2022
11867                                       ;jmp    short LCritDevice_1
11868                                   ;LCritDevice_iret:
11869                                       ;iret
11870
11871                                   LCritDevice_1:
11872                                       push    cs ; *
11873                                       call    LCritDevice_iret ; *
11874
11875                                   LCritDevice_0:
11876                                       PUSH    AX
11877                                       ;MOV    AX,8100h+critDevice
11878                                       ;INT    int_IBM
11879                                       mov     ax,8102h
11880                                       int     2Ah     ; Microsoft Networks - END DOS CRITICAL SECTION
11881                                                       ; AL = critical section number (00h-0Fh)
11882                                       POP     AX
11883                                       retn
11884
11885                                   ;LCritDevice_iret:  ; 12/05/2019 - Retro DOS v4.0
11886                                   ;   iret
11887
11888                                   LCritDevice_2:
11889                                       ;;popff ; *
```

```
                                    ;;retn
                                ;   jmp     short LCritDevice_3 ; *
                                ;LCritDevice_iret2: ; *
                                ;   iret

                                    ; 16/12/2022
                                    ; 13/11/2022
                                    ;jmp    short LCritDevice_3
                                ;LCritDevice_iret2:
                                    ;iret

                                LCritDevice_3:
                                    push    cs ; *
                                    ; 13/11/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
                                    ;call   LCritDevice_iret2 ; *
                                    ;retn
                                    ; 16/12/2022
                                    call    LCritDevice_iret
                                    retn

                                ;EndProc LCritDevice

                                %endif

                                    ; 15/01/2024 - Retro DOS v5.0 (Modified PCDOS 7.1)
                                    ; PCDOS 7.1 IBMDOS.COM - DOSCODE:580Dh

                                ; 15/01/2024 - Retro DOS v5.0
                                %if 1
                                    ;;;
                                ; ----------------------------------------------------------------------------
                                ECritMEM:
                                ECritSFT:
                                ; ----------------------------------------------------------------------------
                                    ; PCDOS 7.1 IBMDOS.COM
                                ECritDisk:
000018E2 51                         push    cx
000018E3 B500                       mov     ch,0
000018E5 368A0E[650D]               mov     cl,[ss:redir_patch]
000018EA E321                       jcxz    ECritDisk_3
000018EC 59                         pop     cx
000018ED 50                         push    ax
000018EE B80180                     mov     ax,8001h        ; BEGIN DOS CRITICAL SECTION
                                                            ; AL = critical section number (01h)
                                ECritDisk_1:
                                ; ----------------------------------------
                                LCritDisk_1:
                                ECritDevice_1:
                                LCritDevice_1:
000018F1 51                         push    cx
000018F2 B500                       mov     ch,0
000018F4 368A0E[5B0F]               mov     cl,[ss:IsWin386]
000018F9 E305                       jcxz    ECritDisk2
000018FB 59                         pop     cx
000018FC CD2A                       int     2Ah             ; Microsoft Networks - BEGIN DOS CRITICAL SECTION
                                                            ; AL = critical section number (00h-0Fh)
000018FE 58                         pop     ax
000018FF C3                         retn

                                ECritDisk2:
00001900 06                         push    es
                                    ;mov    cx,0
                                    ; 15/01/2024
                                    ; cx = 0
00001901 8EC1                       mov     es,cx
00001903 9C                         pushf                   ; simulate INT 2Ah
00001904 26FF1EA800                 call    far [es:00A8h] ; call far (INT 2Ah vector)
00001909 07                         pop     es
0000190A 59                         pop     cx
0000190B 58                         pop     ax
0000190C C3                         retn

                                ECritDisk_3:
                                ; ----------------------------------------
                                LCritDisk_3:
                                ECritDevice_3:
                                LCritDevice_3:
0000190D 59                         pop     cx
0000190E C3                         retn

                                ; ----------------------------------------------------------------------------
                                LCritMEM:
                                LCritSFT:
                                ; ----------------------------------------------------------------------------
                                    ; PCDOS 7.1 IBMDOS.COM
                                LCritDisk:
0000190F 51                         push    cx
00001910 B500                       mov     ch,0
00001912 368A0E[650D]               mov     cl,[ss:redir_patch]
00001917 E3F4                       jcxz    LCritDisk_3
00001919 59                         pop     cx
0000191A 50                         push    ax
0000191B B80181                     mov     ax,8101h        ; END DOS CRITICAL SECTION
                                                            ; AL = critical section number (01h)
0000191E EBD1                       jmp     short LCritDisk_1

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

                                ECritDevice:
00001920 51                         push    cx
00001921 B500                       mov     ch,0
00001923 368A0E[650D]               mov     cl,[ss:redir_patch]
00001928 E3E3                       jcxz    ECritDevice_3
0000192A 59                         pop     cx
0000192B 50                         push    ax
0000192C B80280                     mov     ax,8002h        ; BEGIN DOS CRITICAL SECTION
                                                            ; AL = critical section number (02h)
0000192F EBC0                       jmp     short ECritDevice_1

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

                                LCritDevice:
00001931 51                         push    cx
00001932 B500                       mov     ch,0
00001934 368A0E[650D]               mov     cl,[ss:redir_patch]
00001939 E3D2                       jcxz    LCritDevice_3
0000193B 59                         pop     cx
0000193C 50                         push    ax
0000193D B80281                     mov     ax,8102h        ; END DOS CRITICAL SECTION
                                                            ; AL = critical section number (02h)
00001940 EBAF                       jmp     short LCritDevice_1

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

```
12014                           %endif
12015
12016                           ;==========================================================================
12017                           ; CPMIO.ASM, MSDOS 6.0, 1991
12018                           ;==========================================================================
12019                           ; 20/07/2018 - Retro DOS v3.0
12020
12021                           ;==========================================================================
12022                           ; STDIO.ASM - (MSDOS 2.0)
12023                           ;==========================================================================
12024
12025                           ;
12026                           ; Standard device IO for MSDOS (first 12 function calls)
12027                           ;
12028
12029                           ;.xlist
12030                           ;.xcref
12031                           ;INCLUDE STDSW.ASM
12032                           ;INCLUDE DOSSEG.ASM
12033                           ;.cref
12034                           ;.list
12035
12036                           ;TITLE   STDIO - device IO for MSDOS
12037                           ;NAME    STDIO
12038
12039                           ;INCLUDE IO.ASM
12040
12041                           ; ------------------------------------------------------------------------
12042                           ;
12043                           ; NOTE for Retro DOS v2.0 :  (ERDOGAN TAN - 13/03/2018)
12044                           ;     IO.ASM is missing in MSDOS 2.0 kernel source code files !!!
12045                           ;     INSTEAD of IO.ASM, I have disassembled IBMDOS.COM (MSDOS 2.0)
12046                           ;                       and I have used CPMIO.ASM (MSDOS 6.0 source code)
12047                           ;                       to restore MSDOS 2.0 device IO source code
12048                           ;
12049                           ;          (STRIN.ASM has '$STD_CON_STRING_INPUT' code.)
12050
12051                           ;==========================================================================
12052                           ; STDIO.ASM - (MSDOS 2.0)
12053                           ;==========================================================================
12054
12055                           ;
12056                           ; Standard device IO for MSDOS (first 12 function calls)
12057                           ;
12058
12059                           ;.xlist
12060                           ;.xcref
12061                           ;INCLUDE STDSW.ASM
12062                           ;INCLUDE DOSSEG.ASM
12063                           ;.cref
12064                           ;.list
12065
12066                           ;TITLE   STDIO - device IO for MSDOS
12067                           ;NAME    STDIO
12068
12069                           ;INCLUDE IO.ASM
12070
12071                           ; ------------------------------------------------------------------------
12072                           ;
12073                           ; NOTE for Retro DOS v2.0 :  (ERDOGAN TAN - 13/03/2018)
12074                           ;     IO.ASM is missing in MSDOS 2.0 kernel source code files !!!
12075                           ;     INSTEAD of IO.ASM, I have disassembled IBMDOS.COM (MSDOS 2.0)
12076                           ;                       and I have used CPMIO.ASM (MSDOS 6.0 source code)
12077                           ;                       to restore MSDOS 2.0 device IO source code
12078                           ;
12079                           ;          (STRIN.ASM has '$STD_CON_STRING_INPUT' code.)
12080                           ;
12081                           ;==========================================================================
12082                           ; IO.ASM (MSDOS 2.0) (IBMDOS.COM 2.0) - STRIN.ASM (MSDOS 2.0, 19/08/1983)
12083                           ;==========================================================================
12084                           ; Retro DOS v2.0 by Erdogan Tan, 13/03/2018 - 14/03/2018
12085
12086                           ; (Disassembled code of IBMDOS.COM, 08/03/1983) - Dissassembler: IDA Pro Free
12087                           ; (Comments are from CPMIO.ASM - 1991, MSDOS 6.0)
12088
12089                           ;==========================================================================
12090                           ; CPMIO.ASM (MSDOS 6.0, 1991)
12091                           ;==========================================================================
12092                           ; Retro DOS v4.0 by Erdogan Tan, 04/05/2019
12093
12094                               ; 08/11/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
12095
12096                           ;** Standard device IO for MSDOS (first 12 function calls)
12097                           ;
12098                           ;   TITLE   IBMCPMIO - device IO for MSDOS
12099                           ;   NAME    IBMCPMIO
12100
12101                           ;   Old style CP/M 1-12 system calls to talk to reserved devices
12102                           ;
12103                           ;   $Std_Con_Input_No_Echo
12104                           ;   $Std_Con_String_Output
12105                           ;   $Std_Con_String_Input
12106                           ;   $RawConIO
12107                           ;   $RawConInput
12108                           ;   RAWOUT
12109                           ;   RAWOUT2
12110                           ;
12111
12112                           ; The following routines form the console I/O group (funcs 1,2,6,7,8,9,10,11).
12113                           ; They assume ES and DS NOTHING, while not strictly correct, this forces data
12114                           ; references to be SS or CS relative which is desired.
12115
12116                           ; ------------------------------------------------------------------------
12117
12118                           ;   TITLE   CPMIO2 - device IO for MSDOS
12119                           ;   NAME    CPMIO2
12120
12121                           ;
12122                           ;   Microsoft Confidential
12123                           ;   Copyright (C) Microsoft Corporation 1991
12124                           ;   All Rights Reserved.
12125                           ;
12126
12127                           ;** Old style CP/M 1-12 system calls to talk to reserved devices
12128                           ;
12129                           ;   $Std_Con_Input
12130                           ;   $Std_Con_Output
12131                           ;   OUTT
12132                           ;   TAB
12133                           ;   BUFOUT
12134                           ;   $Std_Aux_Input
12135                           ;   $Std_Aux_Output
12136                           ;   $Std_Printer_Output
12137                           ;   $Std_Con_Input_Status
```

```
                        ;   $Std_Con_Input_Flush
                        ;
                        ;   Revision History:
                        ;
                        ;     AN000  version 4.00 - Jan. 1988

                        ; The following routines form the console I/O group (funcs 1,2,6,7,8,9,10,11).
                        ; They assume ES and DS NOTHING, while not strictly correct, this forces data
                        ; references to be SS or CS relative which is desired.

                        ;DOSCODE SEGMENT
                        ;   ASSUME  SS:DOSDATA,CS:DOSCODE


                        ;hkn;       All the variables use SS override or DS. Therefore there is
                        ;hkn;       no need to specifically set up any seg regs unless SS assumption is
                        ;hkn;       not valid.

                        ; DOSCODE:51BAh (MSDOS 6.21, MSDOS.SYS)
                        ; 08/11/2022
                        ; DOSCODE:51A6h (MSDOS 5.0, MSDOS.SYS)

                        ;
                        ;--------------------------------------------------------------------------
                        ;
                        ; Procedure : $Std_Con_Input_No_Echo
                        ;
                        ;--------------------------------------------------------------------------
                        ;

                        _$STD_CON_INPUT_NO_ECHO:   ;System call 8

                        ; Inputs:
                        ;   None
                        ; Function:
                        ;   Input character from console, no echo
                        ; Returns:
                        ;   AL = character

00001942 1E                         push    ds
00001943 56                         push    si
                        INTEST:
00001944 E86B45                     call    STATCHK
00001947 753A                       jnz     short GET ; 08/09/2018
                                    ;*****************************************************************************
                                    ;hkn; SS override
00001949 36803E[A00A]00             cmp     byte [SS:PRINTER_FLAG],0  ; is printer idle?
0000194F 7505                       jnz     short no_sys_wait
00001951 B405                       mov     ah,5                ; get input status with system wait
00001953 E87237                     call    IOFUNC
                        no_sys_wait:
                                    ;*****************************************************************************
00001956 B484                       MOV     AH,84h          ; (Microsoft Networks - KEYBOARD BUSY LOOP)
00001958 CD2A                       INT     int_IBM ; int 2Ah

                                    ;;; 7/15/86  update the date in the idle loop
                                    ;;; Dec 19, 1986 D.C.L. changed following CMP to Byte Ptr from Word Ptr
                                    ;;;;         to shorten loop in consideration of the PC Convertible

                                    ;hkn; SS override
0000195A 36803E[BD0D]FF             CMP     byte [SS:DATE_FLAG],-1; date is updated may be every
00001960 751A                       JNZ     short NoUpdate        ; 65535 x ? ms if no one calls

00001962 50                         PUSH    AX
00001963 53                         PUSH    BX                  ; following is tricky,
00001964 51                         PUSH    CX                  ; it may be called by critical handler
00001965 52                         PUSH    DX                  ; at that time, DEVCALL is used by
                                                                ; other's READ or WRITE
00001966 1E                         PUSH    DS                  ; save DS = SFT's segment

                                    ;hkn; READTIME must use ds = DOSDATA
                                    ;hkn;       PUSH    CS                  ; READTIME must use DS=CS

00001967 16                         PUSH    SS ; 04/05/2019
00001968 1F                         POP     DS

                                    ;MOV    AX,0                ; therefore, we save DEVCALL
                                    ; 26/06/2024
00001969 31C0                       xor     ax,ax
0000196B E89A02                     CALL    Save_Restore_Packet   ; save DEVCALL packet
                                    ;invoke READTIME              ; readtime
0000196E E819F2                     call    READTIME
00001971 B80100                     MOV     AX,1
00001974 E89102                     CALL    Save_Restore_Packet   ; restore DEVCALL packet

                                    ;   ; MSDOS 3.3 (IBMDOS.COM, Offset 1F8Ch)
                                    ;   ; (MSDOS 6.0 code does not contain IBM DOS FETCHI_TAG check)
                                    ;   push    bx
                                    ;   mov     bx,DATE_FLAG
                                    ;   add     bx,2  ; mov bx,FETCHI_FLAG
                                    ;   cmp     word [cs:bx],5872h
                                    ;   jz      short FETCHI_TAG_chk_ok
                                    ;   call    DOSINIT
                                    ;FETCHI_TAG_chk_ok:
                                    ;   pop     bx

00001977 1F                         POP     DS                  ; restore DS
00001978 5A                         POP     DX
00001979 59                         POP     CX
0000197A 5B                         POP     BX
0000197B 58                         POP     AX
                        NoUpdate:

                                    ;hkn; SS override
0000197C 36FF06[BD0D]               INC     word [SS:DATE_FLAG]

                                    ;;; 7/15/86 ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
00001981 EBC1                       JMP     short INTEST
                        GET:
00001983 30E4                       XOR     AH,AH
00001985 E84037                     call    IOFUNC
00001988 5E                         POP     SI
00001989 1F                         POP     DS
                                    ;;; 7/15/86

                                    ;hkn; SS override
                                    ; MSDOS 6.0
0000198A 36C606[BC0D]00             MOV     BYTE [SS:SCAN_FLAG],0
                                    ;
00001990 3C00                       CMP     AL,0        ; extended code ( AL )
00001992 7505                       JNZ     short noscan

                                    ;hkn; SS override
                                    ;MOV    BYTE [SS:SCAN_FLAG],1 ; set this flag for ALT_Q key
```

```
                                    ; 20/06/2023
00001994 36FE06[BC0D]               inc     byte [SS:SCAN_FLAG]
                                noscan:
00001999 C3                         retn
                                ;
                                ;--------------------------------------------------------------------------
                                ;
                                ;** $STD_CON_STRING_OUTPUT - Console String Output
                                ;
                                ;
                                ;   ENTRY   (DS:DX) Point to output string '$' terminated
                                ;   EXIT    none
                                ;   USES    ALL
                                ;
                                ;--------------------------------------------------------------------------
                                ;

                                _$STD_CON_STRING_OUTPUT:  ;System call 9
0000199A 89D6                       mov     si,dx
                                STRING_OUT1:
0000199C AC                         lodsb
0000199D 3C24                       cmp     al,'$'
0000199F 74F8                       je      short noscan
                                NEXT_STR1:
000019A1 E88D02                     call    OUTT
000019A4 EBF6                       jmp     short STRING_OUT1

                                ;--------------------------------------------------------------------------
                                ;
                                ;** $STD_CON_STRING_INPUT - Input Line from Console
                                ;
                                ;   $STD_CON_STRING_INPUT Fills a buffer from console input until CR
                                ;
                                ;   ENTRY   (ds:dx) = input buffer
                                ;   EXIT    none
                                ;   USES    ALL
                                ;
                                ;--------------------------------------------------------------------------

                                    ; 15/01/2024 - Retro DOS v5.0
                                    ; PCDOS 7.1 IBMDOS.COM - DOSCODE:58D4h

                                _$STD_CON_STRING_INPUT:           ;System call 10

                                    ; 15/01/2024
                                    ;mov    ax,ss
                                    ;mov    es,ax
000019A6 16                         push    ss
000019A7 07                         pop     es

000019A8 89D6                       mov     si,dx
000019AA 30ED                       xor     ch,ch
000019AC AD                         lodsw

                                ;   (AL) = the buffer length
                                ;   (AH) = the template length

000019AD 08C0                           or  al,al
000019AF 74E8                           jz  short noscan   ;Buffer is 0 length!!?
000019B1 88E3                       mov     bl,ah          ;Init template counter
000019B3 88EF                           mov bh,ch          ;Init template counter

                                ;   (BL) = the number of bytes in the template

000019B5 38D8                           cmp al,bl
000019B7 7605                           jbe short NOEDIT   ;If length of buffer inconsistent with contents
000019B9 80380D                         cmp byte [bx+si],c_CR ; 0Dh
000019BC 7402                           jz  short EDITON   ;If CR correctly placed EDIT is OK

                                ; The number of chars in the template is >= the number of chars in buffer or
                                ; there is no CR at the end of the template. This is an inconsistant state
                                ; of affairs. Pretend that the template was empty:
                                ;

                                NOEDIT:
000019BE 88EB                       mov     bl,ch          ;Reset buffer
                                EDITON:
000019C0 88C2                       mov     dl,al
000019C2 4A                         dec     dx             ;DL is # of bytes we can put in the buffer

                                ;   Top level. We begin to read a line in.

                                NEWLIN:
000019C3 36A0[F901]                 mov     al,[SS:CARPOS]
000019C7 36A2[FA01]                 mov     [SS:STARTPOS],al ;Remember position in raw buffer

000019CB 56                         push    si
000019CC BF[FB01]                   mov     di,INBUF ;Build the new line here
000019CF 36882E[7905]               mov     byte [SS:INSMODE],ch ;Insert mode off
000019D4 88EF                       mov     bh,ch          ;No chars from template yet
000019D6 88EE                       mov     dh,ch          ;No chars to new line yet
000019D8 E867FF                     call    _$STD_CON_INPUT_NO_ECHO ;Get first char
000019DB 3C0A                       cmp     al,c_LF        ; 0Ah   ;Linefeed
000019DD 7503                       jnz     short GOTCH

                                ;   This is the main loop of reading in a character and processing it.
                                ;
                                ;   (BH) = the index of the next byte in the template
                                ;   (BL) = the length of the template
                                ;   (DH) = the number of bytes in the buffer
                                ;   (DL) = the length of the buffer

                                GETCH:
000019DF E860FF                     call    _$STD_CON_INPUT_NO_ECHO
                                GOTCH:
                                ;
                                ; Brain-damaged Tim Patterson ignored ^F in case his BIOS did not flush the
                                ; input queue.
                                ;
000019E2 3C06                           cmp al,"F"-"@"  ; CMP AL, 6  ; Ignore ^F
000019E4 74F9                       jz      short GETCH

                                ;   If the leading char is the function-key lead byte

                                    ;cmp    al,[SS:ESCCHAR]

                                    ; 04/05/2019 - Retro DOS v4.0

                                ;hkn;       ESCCHAR is in TABLE seg (DOSCODE)

000019E6 2E3A06[940A]               CMP     AL,[cs:ESCCHAR]
000019EB 7439                           jz  short ESCAPE   ;change reserved keyword DBM 5-7-87

                                ;   Rubout and ^H are both destructive backspaces.
```

```
                                        cmp al,c_DEL ; 7FH
                                        ;jz short BACKSPJ
                                        ; 15/01/2024
000019EF 7466                       je      short BACKSP
000019F1 3C08                       cmp     al,c_BS  ; 8
                                        ;jz short BACKSPJ
                                        ; 15/01/2024
000019F3 7462                       je      short BACKSP

                                    ; 04/05/2019 - MSDOS 6.0, also MSDOS 6.21 has bug (bullshit) here.
                                    ;              Two NOPs -instead of a JMP short, as two bytes-
                                    ;              after CMP and a CMP again!
                                    ;
                                    ;              -It would be better if they use a 'JMP short' to
                                    ;              DOSCODE:5279h from DOSCODE:5271h and leave NOPs
                                    ;              between them. Then, they would be able use a patch
                                    ;              between 5271h and 5279h when if it will be required.
                                    ;              I think Tim Patterson would not do this CMP mistake!-
                                    ;
                                    ; (MSDOS.SYS, from DOSCODE:5271h to DOSCODE:5279h)

                                    ; 08/11/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
                                    ;
                                    ; (Note: nops below might be used for patching code for Windows 3.1)

                                ;DOSCODE:526D      cmp     al, 8
                                ;DOSCODE:526F      jz      short BACKSPJ
                                ;DOSCODE:5271      cmp     al, 17h
                                ;DOSCODE:5273      nop
                                ;DOSCODE:5274      nop
                                ;DOSCODE:5275      cmp     al, 15h
                                ;DOSCODE:5277      nop
                                ;DOSCODE:5278      nop
                                ;DOSCODE:5279      cmp     al, 0Dh
                                ;DOSCODE:527B      jz      short ENDLIN
                                ;DOSCODE:527D      cmp     al, 0Ah
                                ;DOSCODE:527F      jz      short PHYCRLF

                                    ; 08/11/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
                                    ; DOSCODE:525Dh

                                ; 16/12/2022
                                %if 0
                                    ; MSDOS 6.0
                                ;   ^W deletes backward once and then backs up until a letter is before the
                                ;   cursor

                                    CMP     AL,"W"-"@" ; 17h

                                ;   The removal of the comment characters before the jump statement will
                                ;   cause ^W to backup a word.

                                ;***JZ      short WordDel
                                    NOP
                                    NOP

                                    CMP     AL,"U"-"@" ; 15h

                                ;   The removal of the comment characters before the jump statement will
                                ;   cause ^U to clear a line.

                                ;***JZ      short LineDel
                                    NOP
                                    NOP

                                %endif

                                ;   CR terminates the line.

000019F5 3C0D                           cmp al,c_CR ; 0Dh
000019F7 7430                           jz  short ENDLIN

                                ;   LF goes to a new line and keeps on reading.

000019F9 3C0A                           cmp al,c_LF ; 0Ah
000019FB 7442                       jz      short PHYCRLF

                                ;   ^X (or ESC) deletes the line and starts over

                                    ; MSDOS 3.3
                                    ;cmp    al,[ss:CANCHAR] ; 1Bh
                                    ;jz     short KILNEW

                                    ; MSDOS 6.0 (& MSDOS 6.21)

                                ;hkn;       CANCHAR is in TABLE seg (DOSCODE), so CS override

000019FD 2E3A06[930A]               cmp     al,[cs:CANCHAR] ; 1Bh
00001A02 7440                       jz      short KILNEW

                                    ;cmp    al,CANCEL ; 1Bh       ; Retro DOS v3.0
                                    ;jz     short KILNEW

                                ; Otherwise, we save the input character.

                                SAVCH:
                                    cmp     dh,dl
00001A04 38D6                       jnb     short BUFFUL          ; buffer is full.
00001A06 7317                           stosb
00001A08 AA                         inc     dh                      ; increment count in buffer.
00001A09 FEC6                       call    BUFOUT                ; Print control chars nicely
00001A0B E8B702
                                        cmp byte [SS:INSMODE], 0
00001A0E 36803E[7905]00             jnz     short GETCH           ; insertmode => don't advance template
00001A14 75C9                           cmp bh,bl
00001A16 38DF                           jnb short GETCH           ; no more characters in template
00001A18 73C5                           inc si                    ; Skip to next char in template
00001A1A 46                             inc bh                    ; remember position in template
00001A1B FEC7                           jmp short GETCH
00001A1D EBC0

                                    ; 15/01/2024
                                ;BACKSPJ:
                                    ;jmp    short BACKSP

                                BUFFUL:
                                    mov     al, 7                 ; Bell to signal full buffer
00001A1F B007                       call    OUTT
00001A21 E80D02                     jmp     short GETCH
00001A24 EBB9

                                ESCAPE:
                                    ;transfer OEMFunctionKey
00001A26 E996F0                     JMP     OEMFunctionKey        ; let the OEM's handle the key dispatch
```

```
12510                                   ENDLIN:
12511 00001A29 AA                               stosb                           ; Put the CR in the buffer
12512 00001A2A E80402                       call    OUTT                    ; Echo it
12513 00001A2D 5F                           pop di                          ; Get start of user buffer
12514 00001A2E 8875FF                       mov [di-1], dh              ; Tell user how many bytes
12515 00001A31 FEC6                         inc dh                      ; DH is length including CR
12516
12517                                   COPYNEW:
12518                                       ; (IBMDOS.COM, MSDOS 2.0, STRIN.ASM)
12519                                       ;mov    bp, es
12520                                       ;mov    bx, ds
12521                                       ;mov    es, bx
12522                                       ;mov    ds, bp
12523                                       ;mov    si, INBUF
12524                                       ;mov    cl, dh
12525                                       ;rep    movsb
12526                                       ;retn
12527
12528                                       ; CPMIO.ASM (MSDOS 6.0)
12529                                       ; (IBMDOS.COM, MSDOS 3.3, Offset 2061h)
12530                                       ;SAVE   <DS,ES>
12531 00001A33 1E                          PUSH    DS
12532 00001A34 06                          PUSH    ES
12533                                       ;RESTORE <DS,ES>              ; XCHG ES,DS
12534 00001A35 1F                          POP     DS
12535 00001A36 07                          POP     ES
12536
12537                                   ;;hkn; INBUF is in DOSDATA
12538 00001A37 BE[FB01]                         MOV     SI,INBUF
12539 00001A3A 88F1                             MOV     CL,DH                   ; set up count
12540 00001A3C F3A4                             REP     MOVSB                   ; Copy final line to user buffer
12541                                   OLDBAK_RETN:
12542 00001A3E C3                               RETN
12543
12544                                   ;   Output a CRLF to the user screen and do NOT store it into the buffer
12545
12546                                   PHYCRLF:
12547 00001A3F E81B01                      CALL    CRLF
12548 00001A42 EB9B                             JMP short GETCH
12549
12550                                       ; MSDOS 6.0 (& MSDOS 3.3, IBMDOS.COM, 1987)
12551
12552                                   ; DOSCODE:52CAh (MSDOS 621, MSDOS.SYS)
12553
12554                                       ; Note: Following routines were not used in IBMDOS.COM
12555                                       ;       -CRTL+W, CRTL+U is not activated-
12556                                       ;       but they were in the kernel code!?)
12557
12558                                       ; 08/11/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
12559                                       ; DOSCODE:52B6h
12560
12561                                   ;;;;;;;;
12562
12563                                   ; 16/12/2022
12564                                   %if 0
12565                                   ;
12566                                   ; Delete the previous line
12567                                   ;
12568                                   LineDel:
12569                                       OR      DH,DH
12570                                       JZ      short GETCH     ; 06/12/2022
12571                                       Call    BackSpace
12572                                       JMP     short LineDel
12573
12574                                   %endif
12575
12576                                   ;
12577                                   ; delete the previous word.
12578                                   ;
12579                                   WordDel:
12580                                   WordLoop:
12581                                   ;   Call    BackSpace               ; backspace the one spot
12582                                   ;   OR      DH,DH
12583                                   ;   JZ      short GetChj
12584                                   ;   MOV     AL,[ES:DI-1]
12585                                   ;   cmp     al,'0'
12586                                   ;   jb      short GetChj
12587                                   ;   cmp     al,'9'
12588                                   ;   jbe     short WordLoop
12589                                   ;   OR      AL,20h
12590                                   ;   CMP     AL,'a'
12591                                   ;   JB      short GetChj
12592                                   ;   CMP     AL,'z'
12593                                   ;   JBE     short WordLoop
12594                                   ;GetChj:
12595                                   ;   JMP     GETCH
12596
12597                                   ; 16/12/2022
12598                                   %if 0
12599                                       ; 07/12/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
12600                                       ; (Worddel is not called or jumped from anywhere!)
12601                                   WordDel:
12602                                   WordLoop:
12603                                       Call    BackSpace               ; backspace the one spot
12604                                       OR      DH,DH
12605                                       JZ      short GetChj
12606                                       MOV     AL,[ES:DI-1]
12607                                       cmp     al,'0'
12608                                       jb      short GetChj
12609                                       cmp     al,'9'
12610                                       jbe     short WordLoop
12611                                       OR      AL,20h
12612                                       CMP     AL,'a'
12613                                       JB      short GetChj
12614                                       CMP     AL,'z'
12615                                       JBE     short WordLoop
12616                                   GetChj:
12617                                       JMP     GETCH
12618
12619                                   %endif
12620
12621                                   ;;;;;;;;
12622
12623                                   ; DOSCODE:52F3h (MSDOS 621, MSDOS.SYS)
12624
12625                                   ; The user wants to throw away what he's typed in and wants to start over.
12626                                   ; We print the backslash and then go to the next line and tab to the correct
12627                                   ; spot to begin the buffered input.
12628
12629                                   KILNEW:
12630 00001A44 B05C                            mov al,'\'
12631 00001A46 E8E801                          call    OUTT            ;Print the CANCEL indicator
12632 00001A49 5E                              pop si              ;Remember start of edit buffer
12633                                   PUTNEW:
```

```
12634 00001A4A E81001                   call    CRLF            ;Go to next line on screen
12635 00001A4D 36A0[FA01]               mov     al,[SS:STARTPOS]
12636 00001A51 E85102                   call    TAB             ;Tab over
12637 00001A54 E96CFF                       JMP     NEWLIN              ;Start over again
12638
12639                                   ;   Destructively back up one character position
12640
12641                                   BACKSP:
12642                                       ; 09/09/2018
12643 00001A57 E80800                   Call    BackSpace
12644 00001A5A EB83                     JMP     short GETCH   ; 15/01/2024
12645
12646                                       ; 15/01/2024
12647                                   ;User really wants an ESC character in his line
12648                                   TWOESC:
12649 00001A5C 2EA0[940A]               mov     al,[cs:ESCCHAR] ; 10/06/2019
12650 00001A60 EBA2                     jmp     short SAVCH
12651
12652                                   BackSpace:
12653 00001A62 08F6                     or      dh,dh
12654 00001A64 7419                     jz      short OLDBAK   ;No chars in line, do nothing to line
12655 00001A66 E85800                   call    BACKUP          ;Do the backup
12656 00001A69 268A05                   mov     al,[es:di]     ;Get the deleted char
12657 00001A6C 3C20                         cmp al,20h  ; ' '
12658 00001A6E 730F                     jnb     short OLDBAK   ;Was a normal char
12659 00001A70 3C09                         cmp al,c_HT ; 9
12660 00001A72 741B                     jz      short BAKTAB   ;Was a tab, fix up users display
12661                                   ;; 9/27/86 fix for ctrl-U backspace
12662 00001A74 3C15                     CMP     AL,"U"-"@" ; 15h ; ctrl-U is a section symbol not ^U
12663 00001A76 7407                     JZ      short OLDBAK
12664 00001A78 3C14                             CMP     AL,"T"-"@" ; 14h ; ctrl-T is a paragraphs symbol not ^T
12665 00001A7A 7403                     JZ      short OLDBAK
12666                                   ;; 9/27/86 fix for ctrl-U backspace
12667 00001A7C E84500                           call       BACKMES          ;Was a control char, zap the '^'
12668                                   OLDBAK:
12669 00001A7F 36803E[7905]00                   cmp byte [SS:INSMODE], 0
12670 00001A85 75B7                     jnz     short OLDBAK_RETN ;In insert mode, done
12671 00001A87 08FF                     or      bh,bh
12672 00001A89 74B3                             jz  short OLDBAK_RETN
12673                                                               ;Not advanced in template, stay where we are
12674 00001A8B FECF                         dec     bh              ;Go back in template
12675 00001A8D 4E                               dec si
12676 00001A8E C3                           retn
12677                                   BAKTAB:
12678 00001A8F 57                               push       di
12679 00001A90 4F                               dec di              ;Back up one char
12680 00001A91 FD                               std                 ;Go backward
12681 00001A92 88F1                             mov cl,dh           ;Number of chars currently in line
12682 00001A94 B020                             mov al,20h  ; ' '
12683 00001A96 53                               push       bx
12684 00001A97 B307                             mov bl,7            ;Max
12685 00001A99 E30E                             jcxz       FIGTAB           ;At start, do nothing
12686                                   FNDPOS:
12687 00001A9B AE                               scasb                       ;Look back
12688 00001A9C 7609                         jbe     short CHKCNT
12689 00001A9E 26807D0109                   cmp     byte [es:di+1],9
12690 00001AA3 7409                         jz      short HAVTAB   ;Found a tab
12691 00001AA5 FECB                         dec     bl              ;Back one char if non tab control char
12692                                   CHKCNT:
12693 00001AA7 E2F2                             loop       FNDPOS
12694                                   FIGTAB:
12695 00001AA9 362A1E[FA01]                 sub     bl,[SS:STARTPOS]
12696                                   HAVTAB:
12697 00001AAE 28F3                         sub     bl,dh
12698 00001AB0 00D9                         add     cl,bl
12699 00001AB2 80E107                       and     cl,7            ;CX has correct number to erase
12700 00001AB5 FC                           cld                     ;Back to normal
12701 00001AB6 5B                           pop     bx
12702 00001AB7 5F                           pop     di
12703 00001AB8 74C5                         jz      short OLDBAK   ;Nothing to erase
12704                                   TABBAK:
12705 00001ABA E80700                       call    BACKMES
12706 00001ABD E2FB                         loop    TABBAK          ;Erase correct number of chars
12707 00001ABF EBBE                         jmp     short OLDBAK
12708
12709                                   BACKUP:
12710 00001AC1 FECE                             dec dh              ;Back up in line
12711 00001AC3 4F                               dec di
12712                                   BACKMES:
12713 00001AC4 B008                             mov al,c_BS ; 8     ;Backspace
12714 00001AC6 E86801                           call       OUTT
12715 00001AC9 B020                             mov al,20h ; ' '    ;Erase
12716 00001ACB E86301                           call       OUTT
12717 00001ACE B008                             mov al,c_BS ; 8     ;Backspace
12718 00001AD0 E95E01                       jmp     OUTT            ;Done
12719
12720                                       ; 15/01/2024
12721                                   ;User really wants an ESC character in his line
12722                                   ;TWOESC:
12723                                   ;   mov     al,[cs:ESCCHAR] ; 10/06/2019
12724                                   ;   jmp     SAVCH
12725
12726                                   ;Copy the rest of the template
12727                                   COPYLIN:
12728 00001AD3 88D9                             mov cl,bl           ;Total size of template
12729 00001AD5 28F9                         sub     cl,bh           ;Minus position in template, is number to move
12730 00001AD7 EB07                             jmp short COPYEACH
12731
12732                                   COPYSTR:
12733 00001AD9 E83200                       call    FINDOLD         ;Find the char
12734 00001ADC EB02                         jmp     short COPYEACH  ;Copy up to it
12735
12736                                   ;Copy one char from template to line
12737                                   COPYONE:
12738 00001ADE B101                             mov cl,1
12739                                   ;Copy CX chars from template to line
12740                                   COPYEACH:
12741 00001AE0 36C606[7905]00                   mov byte [SS:INSMODE],0   ;All copies turn off insert mode
12742 00001AE6 38D6                         cmp     dh,dl
12743 00001AE8 740F                             jz  short GETCH2          ;At end of line, can't do anything
12744 00001AEA 38DF                             cmp bh,bl
12745 00001AEC 740B                             jz  short GETCH2          ;At end of template, can't do anything
12746 00001AEE AC                               lodsb
12747 00001AEF AA                               stosb
12748 00001AF0 E8D201                       call    BUFOUT
12749 00001AF3 FEC7                             inc bh                  ;Ahead in template
12750 00001AF5 FEC6                             inc dh                  ;Ahead in line
12751 00001AF7 E2E7                             loop       COPYEACH
12752                                   GETCH2:
12753 00001AF9 E9E3FE                           jmp GETCH
12754
12755                                   ;Skip one char in template
12756                                   SKIPONE:
12757 00001AFC 38DF                         cmp     bh,bl
```

```
00001AFE 74F9                       jz      short GETCH2            ;At end of template
00001B00 FEC7                       inc     bh                      ;Ahead in template
00001B02 46                         inc     si
                                        ;jmp       GETCH
                                    ; 15/01/2024
00001B03 EBF4                       jmp     short GETCH2

                                SKIPSTR:
00001B05 E80600                     call    FINDOLD                   ;Find out how far to go
00001B08 01CE                           add si,cx                   ;Go there
00001B0A 00CF                           add bh,cl
                                        ;jmp       GETCH
                                    ; 15/01/2024
00001B0C EBEB                       jmp     short GETCH2

                                ;Get the next user char, and look ahead in template for a match
                                ;CX indicates how many chars to skip to get there on output
                                ;NOTE: WARNING: If the operation cannot be done, the return
                                ;       address is popped off and a jump to GETCH is taken.
                                ;       Make sure nothing extra on stack when this routine
                                ;       is called!!! (no PUSHes before calling it).

                                FINDOLD:
00001B0E E831FE                         call       _$STD_CON_INPUT_NO_ECHO

                                    ; STRIN.ASM (MSDOS 2.11, 19/07/2018)

                                    ;CMP     AL,[SS:ESCCHAR]
                                    ;JNZ     SHORT FINDSETUP

                                    ; CPMIO.ASM (MSDOS 6.0, 04/05/2019 - Retro DOS v4.0)

                                ;hkn; ESCCHAR is in TABLE seg (DOSCODE), so CS override

00001B11 2E3A06[940A]               CMP     AL,[CS:ESCCHAR]             ; did he type a function key?
00001B16 7505                       JNZ     SHORT FINDSETUP             ; no, set up for scan

00001B18 E827FE                     CALL    _$STD_CON_INPUT_NO_ECHO     ; eat next char
00001B1B EB1D                           JMP SHORT NOTFND                ; go try again
                                FINDSETUP:
00001B1D 88D9                       mov     cl,bl
00001B1F 28F9                           sub cl,bh           ;CX is number of chars to end of template
00001B21 7417                       jz      short NOTFND    ;At end of template
00001B23 49                             dec cx              ;Cannot point past end, limit search
00001B24 7414                           jz  short NOTFND    ;If only one char in template, forget it
00001B26 06                         push    es
00001B27 1E                         push    ds
00001B28 07                         pop     es
00001B29 57                         push    di
00001B2A 89F7                       mov     di,si           ;Template to ES:DI
00001B2C 47                         inc     di
00001B2D F2AE                       repne   scasb           ;Look
00001B2F 5F                         pop     di
00001B30 07                         pop     es
00001B31 7507                       jnz     short NOTFND    ;Didn't find the char
00001B33 F6D1                           not cl              ;Turn how far to go into how far we went
00001B35 00D9                           add cl,bl           ;Add size of template
00001B37 28F9                           sub cl,bh           ;Subtract current pos, result distance to skip
                                FINDOLD_RETN:
00001B39 C3                         retn

                                NOTFND:
00001B3A 5D                         pop     bp               ;Chuck return address
                                    ;jmp    GETCH
                                    ; 15/01/2024
                                GETCH2_j:
00001B3B EBBC                       jmp     short GETCH2

                                REEDIT:
00001B3D B040                       mov     al,'@'          ;Output re-edit character
00001B3F E8EF00                     call    OUTT
00001B42 5F                         pop     di
00001B43 57                         push    di
00001B44 06                         push    es
00001B45 1E                         push    ds
00001B46 E8EAFE                     call    COPYNEW         ;Copy current line into template
00001B49 1F                         pop     ds
00001B4A 07                         pop     es
00001B4B 5E                         pop     si
00001B4C 88F3                       mov     bl,dh           ;Size of line is new size template
00001B4E E9F9FE                     jmp     PUTNEW          ;Start over again

                                EXITINS:
                                ENTERINS:
00001B51 36F616[7905]               not     byte [SS:INSMODE]
                                    ;jmp    GETCH
                                    ; 15/01/2024
00001B56 EBE3                       jmp     short GETCH2_j

                                ;Put a real live ^Z in the buffer (embedded)
                                CTRLZ:
00001B58 B01A                       mov     al,"Z"-"@" ; 1Ah
00001B5A E9A7FE                         jmp SAVCH

                                ;Output a CRLF
                                CRLF:
00001B5D B00D                       mov     al,c_CR ; 0Dh
00001B5F E8CF00                     call    OUTT
00001B62 B00A                       mov     al,c_LF ; 0Ah
00001B64 E9CA00                     jmp     OUTT

                                ;
                                ;--------------------------------------------------------------------------
                                ;
                                ;** $RAW_CON_IO - Do Raw Console I/O
                                ;
                                ;   Input or output raw character from console, no echo
                                ;
                                ;   ENTRY   DL = -1 if input
                                ;              =  output character if output
                                ;   EXIT    (AL) = input character if input
                                ;   USES    all
                                ;
                                ;--------------------------------------------------------------------------
                                ; 20/07/2018 - Retro DOS v3.0

                                ; 04/05/2019 - Retro DOS v4.0
                                ; DOSCODE:541Ch (MSDOS 6.21, MSDOS.SYS)

                                ; 08/11/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
                                ; DOSCODE:5408h (MSDOS 5.0, MSDOS.SYS)

                                _$RAW_CON_IO:                   ; System call 6
```

```
12882 00001B67 88D0                           MOV AL,DL
12883 00001B69 3CFF                           CMP AL,-1
12884 00001B6B 7541                       JNZ     SHORT RAWOUT ; 16/12/2022
12885                                     ; 08/11/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
12886                                     ;jz     short rci1
12887                                     ;jmp    short RAWOUT
12888                                     ; 16/12/202
12889                                     ; 07/12/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
12890                                     ;nop
12891                                 rci1:
12892                                             ; Get pointer to register save area
12893 00001B6D 36C43E[8405]               LES     DI,[SS:USER_SP] ; 12/03/2018
12894 00001B72 31DB                       XOR     BX,BX
12895                                     ;CALL   GET_IO_FCB      ; MSDOS 2.11 (Retro DOS v2.0)
12896 00001B74 E85223                     CALL    GET_IO_SFT      ; MSDOS 3.3 & MSDOS 6.0
12897                                         ;JC SHORT RET17
12898 00001B77 72C0                           jc  short FINDOLD_RETN
12899 00001B79 B401                       MOV     AH,1
12900 00001B7B E84A35                     CALL    IOFUNC
12901 00001B7E 750B                       JNZ     SHORT RESFLG
12902 00001B80 E80143                     CALL    SPOOLINT
12903                                     ;OR     BYTE [ES:DI+16H],40H
12904 00001B83 26804D1640                 OR      BYTE [ES:DI+user_env.user_F],40H ; Set user's zero flag
12905 00001B88 30C0                       XOR     AL,AL
12906                                 RET17:
12907 00001B8A C3                         RETN
12908
12909                                 RESFLG:
12910                                     ;AND    BYTE [ES:DI+16H],0FFH-40H  ; 0BFh
12911 00001B8B 26806516BF                 AND     BYTE [ES:DI+user_env.user_F],0FFH-40H
12912                                                     ; Reset user's zero flag
12913                                 ;RILP:
12914                                 rci0:
12915 00001B90 E8F142                             CALL    SPOOLINT
12916                                 ;
12917                                 ;--------------------------------------------------------------------------
12918                                 ;
12919                                 ;** $Raw_CON_INPUT - Raw Console Input
12920                                 ;
12921                                 ;   Input raw character from console, no echo
12922                                 ;
12923                                 ;   ENTRY   none
12924                                 ;   EXIT    (al) = character
12925                                 ;   USES    all
12926                                 ;
12927                                 ;--------------------------------------------------------------------------
12928                                 ;
12929
12930                                 ;rci0:      invoke  SPOOLINT
12931
12932                                     ;entry  $RAW_CON_INPUT
12933
12934                                     ; 04/05/2019 - Retro DOS v4.0
12935
12936                                 ; DOSCODE:544Bh (MSDOS 6.21, MSDOS.SYS)
12937
12938                                     ; 15/01/2024 - Retro DOS v5.0
12939
12940                                 ; DOSCODE:5ACBh (PCDOS 7.1, IBMDOS.COM)
12941
12942                                 _$RAW_CON_INPUT:            ; System call 7
12943 00001B93 53                         push    bx
12944 00001B94 31DB                       XOR     BX,BX
12945                                     ;CALL   GET_IO_FCB      ; MSDOS 2.11 (Retro DOS v2.0)
12946 00001B96 E83023                     CALL    GET_IO_SFT      ; MSDOS 3.3 & MSDOS 6.0
12947 00001B99 5B                         pop     bx
12948 00001B9A 72EE                       JC      SHORT RET17
12949 00001B9C B401                       MOV     AH,1
12950 00001B9E E82735                     CALL    IOFUNC
12951                                     ;JZ     SHORT RILP      ; MSDOS 2.11
12952                                     ;XOR    AH,AH
12953                                         ;CALL       IOFUNC
12954                                         ;RETN
12955 00001BA1 7506                       jnz     short rci5      ; MSDOS 3.3 & MSDOS 6.0
12956 00001BA3 B484                       MOV     AH,84h
12957 00001BA5 CD2A                       INT     int_IBM  ; int 2Ah
12958 00001BA7 EBE7                       JMP     short rci0
12959                                 rci5:
12960 00001BA9 30E4                       XOR     AH,AH
12961                                     ;CALL   IOFUNC
12962                                     ;RETN
12963                                     ; 18/12/2022
12964 00001BAB E91A35                     jmp     IOFUNC
12965
12966                                 ;       Output the character in AL to stdout
12967                                 ;
12968                                     ;entry  RAWOUT
12969                                 RAWOUT:
12970 00001BAE 53                         PUSH    BX
12971 00001BAF BB0100                     MOV     BX,1
12972
12973                                     ;CALL   GET_IO_FCB      ; MSDOS 2.11 (Retro DOS v2.0)
12974 00001BB2 E81423                     CALL    GET_IO_SFT      ; MSDOS 3.3 & MSDOS 6.0
12975 00001BB5 7224                       JC      SHORT RAWRET1
12976
12977                                     ;
12978                                     ; MSDOS 2.11
12979                                         ;TEST       BYTE [SI+18H],080H    ; output to file?
12980                                         ;JZ SHORT RAWNORM         ; if so, do normally
12981                                         ;PUSH       DS
12982                                         ;PUSH       SI
12983                                         ;LDS        SI,[SI+19H]           ; output to special?
12984                                     ;TEST   BYTE [SI+4],ISSPEC
12985                                     ;POP    SI
12986                                     ;
12987
12988                                     ; MSDOS 3.3 & MSDOS 6.0
12989                                     ;mov    bx,[si+5]
12990 00001BB7 8B5C05                     MOV     BX,[SI+SF_ENTRY.sf_flags] ;hkn; DS set up by get_io_sft
12991                                   ;
12992                                   ; If we are a network handle OR if we are not a local device then go do the
12993                                   ; output the hard way.
12994                                   ;
12995                                     ;and    bx,8080h
12996 00001BBA 81E38080                   AND     BX,sf_isnet+devid_device
12997                                     ;cmp    bx,80h
12998 00001BBE 81FB8000                   CMP     BX,devid_device
12999 00001BC2 7519                       jnz      short RAWNORM
13000 00001BC4 1E                         push     ds
13001                                     ;lds    bx,[si+7]
13002 00001BC5 C55C07                     LDS     BX,[SI+SF_ENTRY.sf_devptr] ; output to special?
13003                                     ;test   byte [bx+4],10h
13004 00001BC8 F6470410                   TEST    BYTE [BX+SYSDEV.ATT],ISSPEC
13005                                     ;
```

```
13006
13007 00001BCC 1F                         POP     DS
13008 00001BCD 740E                       JZ      SHORT RAWNORM         ; if not, do normally
13009
13010                                     ; 15/01/2024
13011                                     ;INT    int_fastcon  ; int 29h; quickly output the char
13012
13013                                     ; 26/06/2024
13014 00001BCF 1E                         push    ds ; *
13015
13016                                     ; 12/04/2024 - Retro DOS v5.0
13017                                     ; (PCDOS 7.1 IBMBOS.COM)
13018 00001BD0 31DB                       xor     bx,bx
13019 00001BD2 8EDB                       mov     ds,bx ; 0
13020
13021                                     ; 15/01/2024
13022 00001BD4 9C                         pushf                   ; simulate INT 29h
13023 00001BD5 FF1EA400                   call    far [29h*4]   ; call far [00A4h]
13024
13025                                     ; 26/06/2024
13026 00001BD9 1F                         pop     ds ; *
13027
13028                                     ;JMP    SHORT RAWRET
13029                                   ;RAWNORM:
13030                                   ;   CALL    RAWOUT3
13031                                   RAWRET:
13032 00001BDA F8                         CLC
13033                                   RAWRET1:
13034 00001BDB 5B                         POP     BX
13035                                   RAWRET2:
13036 00001BDC C3                         RETN
13037                                   RAWNORM:
13038 00001BDD E80700                     CALL    RAWOUT3
13039 00001BE0 EBF8                       jmp     short RAWRET
13040
13041                                   ;   Output the character in AL to handle in BX
13042                                   ;
13043                                   ;   entry   RAWOUT2
13044
13045                                   RAWOUT2:
13046                                       ;CALL   GET_IO_FCB     ; MSDOS 2.11 (Retro DOS v2.0)
13047                                       ;JC     SHORT RET18
13048 00001BE2 E8E422                     CALL    GET_IO_SFT     ; MSDOS 3.3 & MSDOS 6.0
13049 00001BE5 72F5                       JC      SHORT RAWRET2
13050                                   RAWOUT3:
13051 00001BE7 50                         PUSH    AX
13052 00001BE8 EB0C                       JMP     SHORT RAWOSTRT
13053                                   ROLP:
13054 00001BEA E89742                     CALL    SPOOLINT
13055
13056                                     ; 01/05/2019 - Retro DOS v4.0
13057
13058                                     ; MSDOS 6.0
13059                                     ;OR     word [ss:DOS34_FLAG],CTRL_BREAK_FLAG ; 0010000000000b
13060                                     ; 17/12/2022
13061 00001BED 36800E[1206]02             or      byte [ss:DOS34_FLAG+1],(CTRL_BREAK_FLAG>>8) ; 02h
13062                                     ;or     word [ss:DOS34_FLAG],200h
13063                                                             ;AN002; set control break
13064                                     ;invoke DSKSTATCHK
13065 00001BF3 E80D42                     call    DSKSTATCHK     ;AN002; check control break
13066                                   RAWOSTRT:
13067 00001BF6 B403                       MOV     AH,3
13068 00001BF8 E8CD34                     CALL    IOFUNC
13069 00001BFB 74ED                       JZ      SHORT ROLP
13070
13071                                     ; MSDOS 6.0
13072                                   ;SR;
13073                                   ; IOFUNC now returns ax = 0ffffh if there was an I24 on a status call and
13074                                   ;the user failed. We do not send a char if this happens. We however return
13075                                   ;to the caller with carry clear because this DOS call does not return any
13076                                   ;status.
13077                                   ;
13078 00001BFD 40                         inc     ax              ;fail on I24 if ax = -1
13079 00001BFE 58                         POP     AX
13080 00001BFF 7405                       jz      short nosend    ;yes, do not send char
13081 00001C01 B402                       MOV     AH,2
13082 00001C03 E8C234                     call    IOFUNC
13083                                   nosend:
13084 00001C06 F8                         CLC                     ; Clear carry indicating successful
13085 00001C07 C3                         retn
13086
13087                                     ; MSDOS 3.3 & MSDOS 2.11
13088                                     ;POP    AX
13089                                     ;MOV    AH,2
13090                                        ;CALL      IOFUNC
13091                                     ;CLC                    ; Clear carry indicating successful
13092                                   ;RET18:
13093                                     ;RETN
13094
13095                                   ;;10/08/2018
13096                                   ; 20/07/2018 - Retro DOS v3.0
13097                                   ; ----------------------------------------------------------------------------
13098                                   ; Retro DOS v2.0 (MSDOS 2.11) - OUTMES
13099                                   ; ----------------------------------------------------------------------------
13100
13101                                   ; This routine is called at DOS init
13102
13103                                   ;;  ;procedure OUTMES,NEAR ; String output for internal messages
13104                                   ;;OUTMES:
13105                                   ;;  ;LODS   CS:BYTE PTR [SI]
13106                                   ;;  CS      LODSB
13107                                   ;;  CMP     AL,"$" ; 24h
13108                                   ;;  JZ      SHORT RET18
13109                                   ;;  CALL    OUTT
13110                                   ;;  JMP     SHORT OUTMES
13111
13112                                   ; ----------------------------------------------------------------------------
13113
13114                                   ; 20/07/2018 - Retro DOS v3.0
13115
13116                                   ; IBMDOS.COM (MSDOS 3.3 kernel) - Offset 2252h
13117
13118                                   ;
13119                                   ;----------------------------------------------------------------------------
13120                                   ;
13121                                   ; Inputs:
13122                                   ;   AX=0 save the DEVCALL request packet
13123                                   ;     =1 restore the DEVCALL request packet
13124                                   ; Function:
13125                                   ;   save or restore the DEVCALL packet
13126                                   ; Returns:
13127                                   ;   none
13128                                   ;
13129                                   ;----------------------------------------------------------------------------
```

```
                                ;
                                ; 04/05/2019 - Retro DOS v4.0
                                ; DOSCODE:54B9h (MSDOS 6.21, MSDOS.SYS)
                                ; 08/11/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
                                ; DOSCODE:54A5h (MSDOS 5.0, MSDOS.SYS)
                                ; 12/05/2019
                                    ; 15/01/2024 - Retro DOS v5.0
                                    ; DOSCODE:5B42h (PCDOS 7.1, IBMDOS.COM)
                                Save_Restore_Packet:
00001C08 1E                         PUSH    DS
00001C09 06                         PUSH    ES
00001C0A 56                         PUSH    SI
00001C0B 57                         PUSH    DI
                                    ; 16/12/2022
                                    ; 08/11/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
                                    ; 09/09/2018
00001C0C BF[A00D]                   mov     di,FAKE_STACK_2F
00001C0F BE[5A03]                   mov     si,DEVCALL
                                    ;
                                    ; 21/09/2023
00001C12 09C0                       or      ax,ax
                                    ;CMP    AX,0            ; save packet
00001C14 7402                       JZ      short save_packet ; 16/12/2022
                                    ;je     short set_seg
                                    ; MSDOS 6.0
                                restore_packet:
                                ;   MOV     SI,OFFSET DOSDATA:Packet_Temp ;source
                                ;   MOV     DI,OFFSET DOSDATA:DEVCALL     ;destination
                                    ; MSDOS 3.3
                                    ;mov    si,FAKE_STACK_2F ; DOS_TEMP ; Packed_Temp
                                    ;mov    di,DEVCALL  ; 09/09/2018
                                    ;
                                    ;JMP    short set_seg
                                    ; 16/12/2022
                                    ; 09/09/2018
00001C16 87F7                       xchg    si,di  ; DI = offset DEVCALL, SI = offset FAKE_STACK_2F
                                ; 16/12/2022
                                %if 0
                                    ; 08/11/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
                                    cmp     ax,0            ; save packet
                                    jz      short save_packet
                                    mov     si,FAKE_STACK_2F ; 07/12/2022
                                    mov     di,DEVCALL
                                    jmp     short set_seg
                                    ; MSDOS 6.0
                                save_packet:
                                ;   MOV     DI,OFFSET DOSDATA:Packet_Temp ;destination
                                ;   MOV     SI,OFFSET DOSDATA:DEVCALL     ;source
                                    ; 09/09/2018
                                    ; MSDOS 3.3
                                    ;mov    di,FAKE_STACK_2F ; DOS_TEMP ; Packed_Temp
                                    ;mov    si,DEVCALL ; 09/09/2018
                                    ; 07/12/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
                                    mov     di,FAKE_STACK_2F ; DOS_TEMP ; Packed_Temp
                                    mov     si,DEVCALL
                                %endif
                                ; 16/12/2022
                                save_packet:
                                ;set_seg:
                                    ; MSDOS 3.3
                                    ;mov    ax,cs
                                    ; MSDOS 6.0
                                    ;MOV    AX,SS           ; set DS,ES to DOSDATA
                                    ;MOV    DS,AX
                                    ;MOV    ES,AX
                                    ; 15/01/2024
00001C18 16                         push    ss
00001C19 1F                         pop     ds
00001C1A 1E                         push    ds
00001C1B 07                         pop     es
00001C1C B90B00                     MOV     CX,11           ; 11 words to move
00001C1F F3A5                       REP     MOVSW
00001C21 5F                         POP     DI
00001C22 5E                         POP     SI
00001C23 07                         POP     ES
00001C24 1F                         POP     DS
00001C25 C3                         retn
                                ;=============================================================================
                                ; CPMIO2.ASM, MSDOS 6.0, 1991
                                ;=============================================================================
                                ; 20/07/2018 - Retro DOS v3.0
                                ; 01/05/2019 - Retro DOS v4.0
                                ;hkn;       All the variables use SS override or DS. Therefore there is
                                ;hkn;       no need to specifically set up any seg regs unless SS assumption is
                                ;hkn;       not valid.
                                ;
                                ;--------------------------------------------------------------------------
                                ;
                                ;** $STD_CON_INPUT - System Call 1
                                ;
                                ;   Input character from console, echo
                                ;
                                ;   ENTRY   none
                                ;   EXIT    (al) = character
                                ;   USES    ALL
                                ;
                                ;--------------------------------------------------------------------------
                                ;
                                _$STD_CON_INPUT:   ;System call 1
00001C26 E819FD                     CALL    _$STD_CON_INPUT_NO_ECHO
00001C29 50                         PUSH    AX
00001C2A E80400                     CALL    OUTT
00001C2D 58                         POP     AX
                                CON_INPUT_RETN:
```

```
00001C2E C3                             RETN

                                ;
                                ;--------------------------------------------------------------------------
                                ;
                                ;** $STD_CON_OUTPUT - System Call 2
                                ;
                                ;   Output character to console
                                ;
                                ;   ENTRY   (dl) = character
                                ;   EXIT    none
                                ;   USES    all
                                ;
                                ;--------------------------------------------------------------------------
                                ;

                                ; DOSCODE:54E9h (MSDOS 6.21, MSDOS.SYS)

                                ; 08/11/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
                                ; DOSCODE:54D5h (MSDOS 5.0, MSDOS.SYS)

                                ; 15/01/2024 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMDOS.COM)
                                ; DOSCODE:5B70h (PCDOS 7.1, IBMDOS.COM)

                                _$STD_CON_OUTPUT:  ;System call 2

00001C2F 88D0                       MOV     AL,DL
                                OUTT:
00001C31 3C20                       CMP     AL,20H ; " "
00001C33 725C                       JB      SHORT CTRLOUT
00001C35 3C7F                       CMP     AL,c_DEL ; 7Fh
00001C37 7405                       JZ      SHORT OUTCH
                                OUTCHA:
                                    ;INC    BYTE PTR [CARPOS]
00001C39 36FE06[F901]               INC     BYTE [SS:CARPOS]
                                OUTCH:
00001C3E 1E                         PUSH    DS
00001C3F 56                         PUSH    SI
                                    ;INC    BYTE PTR [CHARCO]           ;invoke statchk...
                                    ;AND    BYTE PTR [CHARCO],00111111B  ;AN000; every 64th char
00001C40 36FE06[0003]               INC     BYTE [SS:CHARCO]
                                    ;AND    BYTE [SS:CHARCO],00111111B
                                    ; 01/05/2019 - Retro DOS v4.0
00001C45 368026[0003]3F             and     byte [SS:CHARCO],3Fh
00001C4B 7505                       JNZ     SHORT OUTSKIP

00001C4D 50                         PUSH    AX
00001C4E E86142                     CALL    STATCHK
00001C51 58                         POP     AX
                                OUTSKIP:
00001C52 E859FF                     CALL    RAWOUT                      ;output the character

00001C55 5E                         POP     SI
00001C56 1F                         POP     DS

                                    ;TEST   BYTE PTR [PFLAG],-1
                                    ;retz
00001C57 36F606[FE02]FF             TEST    BYTE [SS:PFLAG],0FFh
00001C5D 74CF                       JZ      SHORT CON_INPUT_RETN

00001C5F 53                         PUSH    BX
00001C60 1E                         PUSH    DS
00001C61 56                         PUSH    SI
00001C62 BB0100                     MOV     BX,1
                                    ; 20/07/2018 - Retro DOS v3.0
                                    ; MSDOS 3.3
                                    ; MSDOS 6.0 (CPMIO2.ASM)
00001C65 E86122                     CALL    GET_IO_SFT              ;hkn; GET_IO_SFT will set up DS:SI
                                                                    ;hkn; to sft entry
00001C68 7224                       JC      SHORT TRIPOPJ

                                    ; 01/05/2019 - Retro DOS v4.0

                                    ;mov    bx,[si+5]
00001C6A 8B5C05                     MOV     BX,[SI+SF_ENTRY.sf_flags]
                                    ;test   bx,8000h
                                    ;TEST   BX,sf_isnet    ; 8000h      ; output to NET?
00001C6D F6C780                     test    bh,(sf_isnet>>8) ; 80h
00001C70 751C                       JNZ     short TRIPOPJ               ; if so, no echo
                                    ;;test  bx,80h
                                    ;TEST   BX,devid_device             ; output to file?
00001C72 F6C380                     test    bl,devid_device ; 80h
00001C75 7417                       JZ      SHORT TRIPOPJ               ; if so, no echo
                                    ; 14/03/2018
                                    ;call   GET_IO_FCB          ; IBMDOS.COM, MSDOS 2.11
                                    ;jc     short TRIPOPJ
                                    ; MSDOS 2.11
                                    ;test   byte [SI+18H], 80h
                                    ;jz     short TRIPOPJ
00001C77 BB0400                     MOV     BX,4
00001C7A E84C22                     CALL    GET_IO_SFT
00001C7D 720F                       JC      SHORT TRIPOPJ
                                    ;;test  word [si+5], 800h
                                    ;TEST   word [SI+SF_ENTRY.sf_flags],sf_net_spool ; 800H
                                    ;test   byte [si+6],8 ; 08/11/2022
00001C7F F6440608                   test    byte [SI+SF_ENTRY.sf_flags+1],(sf_net_spool>>8) ; 8
                                                                    ; StdPrn redirected?
                                    ;;JZ    SHORT LISSTRT2J                 ; No, OK to echo
                                    ;jz     LISSTRT2 ; 10/08/2018
                                    ; 16/12/2022
00001C83 7503                       jnz     short outch1
00001C85 E98700                     jmp     LISSTRT2
                                    ; 08/11/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
                                    ;jz     short LISSTRT2J
                                outch1:
                                    ;MOV    BYTE [PFLAG],0
00001C88 36C606[FE02]00             MOV     BYTE [SS:PFLAG],0           ; If a spool, NEVER echo
                                    ; MSDOS 2.11
                                    ;mov    bx,4
                                    ;jmp    short LISSTRT2

                                TRIPOPJ:
                                    ; 20/07/2018
00001C8E E98100                     JMP     TRIPOP

                                    ; 16/12/2022
                                    ; 08/11/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
                                ;LISSTRT2J:
                                ;   JMP     LISSTRT2

                                CTRLOUT:
00001C91 3C0D                       CMP     AL,c_CR ; 0Dh
00001C93 7420                       JZ      SHORT ZERPOS
00001C95 3C08                       CMP     AL,c_BS ; 8
```

```
00001C97 7424                       JZ      SHORT BACKPOS
00001C99 3C09                       CMP     AL,c_HT ; 9
00001C9B 75A1                       JNZ     SHORT OUTCH
                                    ;MOV    AL,[CARPOS]
00001C9D 36A0[F901]                 MOV     AL,[SS:CARPOS]
00001CA1 0CF8                       OR      AL,0F8H
00001CA3 F6D8                       NEG     AL
                                TAB:
00001CA5 51                         PUSH    CX
00001CA6 88C1                       MOV     CL,AL
00001CA8 B500                       MOV     CH,0
00001CAA E307                       JCXZ    POPTAB
                                TABLP:
00001CAC B020                       MOV     AL," "
00001CAE E880FF                     CALL    OUTT
00001CB1 E2F9                       LOOP    TABLP
                                POPTAB:
00001CB3 59                         POP     CX

00001CB4 C3                         RETN

                                ZERPOS:
                                    ;MOV    BYTE PTR [CARPOS],0
00001CB5 36C606[F901]00             MOV     BYTE [SS:CARPOS],0
                                    ; 10/08/2018
00001CBB EB81                       JMP     short OUTCH ; 04/05/2019

                                    ; 18/12/2022
                                ;OUTJ:
                                    ;JMP    OUTT

                                BACKPOS:
                                    ;DEC    BYTE PTR [CARPOS]
00001CBD 36FE0E[F901]               DEC     BYTE [SS:CARPOS]
00001CC2 E979FF                     JMP     OUTCH

                                BUFOUT:
00001CC5 3C20                       CMP     AL," "
00001CC7 7315                       JAE     SHORT OUTJ              ;Normal char
00001CC9 3C09                       CMP     AL,9
00001CCB 7411                       JZ      SHORT OUTJ              ;OUT knows how to expand tabs
                                    ;DOS 3.3  7/14/86
00001CCD 3C15                       CMP     AL,"U"-"@" ; 15h        ; turn ^U to section symbol
00001CCF 740D                       JZ      short CTRLU
00001CD1 3C14                       CMP     AL,"T"-"@" ; 14h        ; turn ^T to paragraph symbol
00001CD3 7409                       JZ      short CTRLU
                                NOT_CTRLU:
                                    ;DOS 3.3  7/14/86
00001CD5 50                         PUSH    AX
00001CD6 B05E                       MOV     AL,"^"
00001CD8 E856FF                     CALL    OUTT            ;Print '^' before control chars
00001CDB 58                         POP     AX
00001CDC 0C40                       OR      AL,40H          ;Turn it into Upper case mate
                                CTRLU:
                                    ;CALL   OUTT
                                    ; 18/12/2022
                                OUTJ:
00001CDE E950FF                     jmp     OUTT
                                ;BUFOUT_RETN:
                                    ;RETN

                                ;
                                ;--------------------------------------------------------------------------
                                ;
                                ;** $STD_AUX_INPUT - System Call 3
                                ;
                                ;   $STD_AUX_INPUT returns a character from Aux Input
                                ;
                                ;   ENTRY   none
                                ;   EXIT    (al) = character
                                ;   USES    all
                                ;
                                ;--------------------------------------------------------------------------
                                ;

                                    ; 08/11/2022 Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)

                                _$STD_AUX_INPUT:   ;System call 3

00001CE1 E8CE41                     CALL    STATCHK
00001CE4 BB0300                     MOV     BX,3
00001CE7 E8DF21                     CALL    GET_IO_SFT      ; 20/07/2018 - MSDOS 3.3 (MSDOS 6.0)
                                    ;CALL   GET_IO_FCB      ; 14/03/2018 - MSDOS 2.11
                                    ;retc
                                    ; 16/12/2022
                                    ; 08/11/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
                                    ;JC     SHORT BUFOUT_RETN
                                    ;JMP    SHORT TAISTRT
                                    ; 07/12/2022
00001CEA 7304                       jnc     SHORT TAISTRT
00001CEC C3                         retn

                                AUXILP:
00001CED E89441                     CALL    SPOOLINT
                                TAISTRT:
00001CF0 B401                       MOV     AH,1
00001CF2 E8D333                     CALL    IOFUNC
00001CF5 74F6                       JZ      SHORT AUXILP
00001CF7 30E4                       XOR     AH,AH
                                    ; 16/12/2022
                                    ;CALL   IOFUNC
                                    ;RETN
                                    ; 07/12/2022
00001CF9 E9CC33                     jmp     IOFUNC

                                ;
                                ;--------------------------------------------------------------------------
                                ;
                                ;** $STD_AUX_OUTPUT - Output character to AUX
                                ;
                                ;   ENTRY   (dl) = character
                                ;   EXIT    none
                                ;   USES    all
                                ;
                                ;--------------------------------------------------------------------------
                                ;

                                _$STD_AUX_OUTPUT:  ;System call 4

00001CFC 53                         PUSH    BX
00001CFD BB0300                     MOV     BX,3
00001D00 EB04                       JMP     SHORT SENDOUT

                                ;
```

```
                                ;--------------------------------------------------------------------------
                                ;
                                ;** $STD_PRINTER_OUTPUT - Output character to printer
                                ;
                                ;   ENTRY   (dl) = character
                                ;   EXIT    none
                                ;   USES    all
                                ;
                                ;--------------------------------------------------------------------------
                                ;

                                _$STD_PRINTER_OUTPUT:     ;System call 5

00001D02 53                         PUSH    BX
00001D03 BB0400                     MOV     BX,4
                                SENDOUT:
00001D06 88D0                       MOV     AL,DL
00001D08 50                         PUSH    AX
00001D09 E8A641                     CALL    STATCHK
00001D0C 58                         POP     AX
00001D0D 1E                         PUSH    DS
00001D0E 56                         PUSH    SI

                                LISSTRT2:
00001D0F E8D0FE                     CALL    RAWOUT2
                                TRIPOP:
00001D12 5E                         POP     SI
00001D13 1F                         POP     DS
00001D14 5B                         POP     BX

                                SCIS_RETN:  ; 20/07/2018
00001D15 C3                         RETN
                                ;
                                ;--------------------------------------------------------------------------
                                ;
                                ;** $STD_CON_INPUT_STATUS - System Call 11
                                ;
                                ;   Check console input status
                                ;
                                ;   ENTRY   none
                                ;   EXIT    AL = -1 character available, = 0 no character
                                ;   USES    all
                                ;
                                ;--------------------------------------------------------------------------
                                ;

                                _$STD_CON_INPUT_STATUS:           ;System call 11

00001D16 E89941                     CALL    STATCHK
00001D19 B000                       MOV     AL,0            ; no xor!!
                                    ;retz
00001D1B 74F8                       JZ      SHORT SCIS_RETN ; 15/04/2018
                                    ;OR     AL,-1
                                    ; 15/01/2024 (PCDOS 7.1 IBMDOS.COM)
00001D1D 48                         dec     ax ; al = -1
                                ;SCIS_RETN:
00001D1E C3                         RETN
                                ;
                                ;--------------------------------------------------------------------------
                                ;
                                ;** $STD_CON_INPUT_FLUSH - System Call 12
                                ;
                                ;   Flush console input buffer and perform call in AL
                                ;
                                ;   ENTRY   (AL) = DOS function to be called after flush (1,6,7,8,10)
                                ;   EXIT    (al) = 0 iff (al) was not one of the supported fcns
                                ;           return arguments for the fcn supplied in (AL)
                                ;   USES    all
                                ;
                                ;--------------------------------------------------------------------------
                                ;

                                _$STD_CON_INPUT_FLUSH:            ;System call 12

00001D1F 50                         PUSH    AX
00001D20 52                         PUSH    DX
00001D21 31DB                       XOR     BX,BX
00001D23 E8A321                     CALL    GET_IO_SFT      ; 20/07/2018 - MSDOS 3.3 (MSDOS 6.0)
                                    ;CALL   GET_IO_FCB      ; 14/03/2018 - MSDOS 2.11
00001D26 7205                       JC      SHORT BADJFNCON
00001D28 B404                       MOV     AH,4
00001D2A E89B33                     CALL    IOFUNC

                                BADJFNCON:
00001D2D 5A                         POP     DX
00001D2E 58                         POP     AX
00001D2F 88C4                       MOV     AH,AL
00001D31 3C01                       CMP     AL,1
00001D33 7413                       JZ      SHORT REDISPJ
00001D35 3C06                       CMP     AL,6
00001D37 740F                       JZ      SHORT REDISPJ
00001D39 3C07                       CMP     AL,7
00001D3B 740B                       JZ      SHORT REDISPJ
00001D3D 3C08                       CMP     AL,8
00001D3F 7407                       JZ      SHORT REDISPJ
00001D41 3C0A                       CMP     AL,10
00001D43 7403                       JZ      SHORT REDISPJ
00001D45 B000                       MOV     AL,0
00001D47 C3                         RETN

                                REDISPJ:
00001D48 FA                         CLI
                                    ;transfer REDISP
00001D49 E929E6                     JMP     REDISP

                                ;==========================================================================
                                ; FCBIO.ASM, MSDOS 6.0, 1991
                                ;==========================================================================
                                ; 20/07/2018 - Retro DOS v3.0
                                ; 17/05/2019 - Retro DOS v4.0

                                ;** FCBIO.ASM - Ancient 1.0 1.1 FCB system calls
                                ;
                                ;   $GET_FCB_POSITION
                                ;   $FCB_DELETE
                                ;   $GET_FCB_FILE_LENGTH
                                ;   $FCB_CLOSE
                                ;   $FCB_RENAME
                                ;   SaveFCBInfo
                                ;   ResetLRU
                                ;   SetOpenAge
                                ;   LRUFCB
                                ;   FCBRegen
                                ;   BlastSFT
                                ;   CheckFCB
                                ;   SFTFromFCB
```

```
                                ;   FCBHardErr
                                ;
                                ;   Revision history:
                                ;
                                ;           Created: ARR 4 April 1983"
                                ;                   MZ  6 June  1983 completion of functions
                                ;                   MZ 15 Dec   1983 Brain damaged programs close FCBs multiple
                                ;                                    times.  Change so successive closes work by
                                ;                                    always returning OK.Also, detect I/O to
                                ;                                    already closed FCB and return EOF.
                                ;                   MZ 16 Jan   1984 More braindamage.  Need to separate info
                                ;                                    out of sft into FCB for reconnection
                                ;
                                ;           A000    version 4.00  Jan. 1988

                               ;Break <$Get_FCB_Position - set random record fields to current pos>
                               ;-----------------------------------------------------------------------------
                               ;
                               ;   $Get_FCB_Position - look at an FCB, retrieve the current position from the
                               ;   extent and next record field and set the random record field to point
                               ;   to that record
                               ;
                               ;   Inputs: DS:DX point to a possible extended FCB
                               ;   Outputs:       The random record field of the FCB is set to the current record
                               ;   Registers modified: all
                               ;
                               ;-----------------------------------------------------------------------------
                               ;

                               _$GET_FCB_POSITION:
00001D4C E8FB04                    call    GetExtended           ; point to FCB
00001D4F E8CC04                    call    GetExtent             ; DX:AX is current record
                                   ;mov    [si+21h],ax
00001D52 894421                    MOV     [SI+SYS_FCB.RR],AX    ; drop in low order piece
                                   ;mov    [si+23h],dl
00001D55 885423                    MOV     [SI+SYS_FCB.RR+2],DL  ; drop in high order piece
                                   ;cmp    word [si+0Eh],64
00001D58 837C0E40                  CMP     word [SI+SYS_FCB.RECSIZ],64
00001D5C 7303                      JAE     short GetFCBBye
                                   ;mov    [si+24h],dh
00001D5E 887424                    MOV     [SI+SYS_FCB.RR+2+1],DH; Set 4th byte only if record size < 64
                               GoodPath:   ; 16/12/2022
                               GetFCBBye:
00001D61 E926E9                    jmp     FCB_RET_OK

                               ;Break <$FCB_Delete - remove several files that match the input FCB>
                               ;-----------------------------------------------------------------------------
                               ;
                               ;** $FCB_Delete - Delete from FCB Template
                               ;
                               ;   given an FCB, remove all directory entries in the current
                               ;   directory that have names that match the FCB's ?  marks.
                               ;
                               ;   ENTRY   (DS:DX) = address of FCB
                               ;   EXIT    entries matching the FCB are deleted
                               ;           (al) = ff iff no entries were deleted
                               ;   USES    all
                               ;
                               ;-----------------------------------------------------------------------------
                               ;
                                   ; 08/11/2022 Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)

                               _$FCB_DELETE:              ; System call 19
                                                          ; OpenBuf is in DOSDATA
00001D64 BF[BE03]                  MOV     DI,OPENBUF            ; appropriate place

00001D67 E8925E                    call    TransFCB              ; convert FCB to path
00001D6A 7207                      JC      short BadPath         ; signal no deletions

00001D6C 16                        push    SS
00001D6D 1F                        pop     DS                    ; SS is DOSDATA

00001D6E E8F30D                    call    DOS_DELETE            ; wham
                                   ;JC     short BadPath
                                   ; 16/12/2022
00001D71 73EE                      jnc     short GoodPath
                               ;GoodPath:
                               ;   ;jmp    FCB_RET_OK            ; do a good return
                               ;   ; 08/11/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
                               ;   jmp     short GetFCBBye

                               BadPath:
                                   ; Error code is in AX

00001D73 E917E9                    jmp     FCB_RET_ERR           ; let someone else signal the error

                               ;Break <$Get_FCB_File_Length - return the length of a file>
                               ;-----------------------------------------------------------------------------
                               ;
                               ;   $Get_FCB_File_Length - set the random record field to the length of the
                               ;   file in records (rounded up if partial).
                               ;
                               ;   Inputs: DS:DX - point to a possible extended FCB
                               ;   Outputs:       Random record field updated to reflect the number of records
                               ;   Registers modified: all
                               ;
                               ;-----------------------------------------------------------------------------
                               ;
                                   ; 15/01/2024 - Retrodos v5.0 (Modified PCDOS 7.1 IBMDOS.COM)

                               _$GET_FCB_FILE_LENGTH:

00001D76 E8D104                    call    GetExtended           ; get real FCB pointer
                                                                 ; DX points to Input FCB

                                                                 ; OpenBuf is in DOSDATA
00001D79 BF[BE03]                  MOV     DI,OPENBUF            ; appropriate buffer

00001D7C 1E                        push    ds                    ; save pointer to true FCB
00001D7D 56                        push    si
00001D7E E87B5E                    call    TransFCB              ; Trans name DS:DX, sets SATTRIB
00001D81 5E                        pop     si
00001D82 1F                        pop     ds
00001D83 72EE                      JC      short BadPath
00001D85 1E                        push    ds                    ; save pointer
00001D86 56                        push    si
00001D87 16                        push    ss
00001D88 1F                        pop     ds
00001D89 E83712                    call    GET_FILE_INFO         ; grab the info
00001D8C 5E                        pop     si                    ; get pointer back
00001D8D 1F                        pop     ds
00001D8E 72E3                      JC      short BadPath         ; invalid something
                                   ; 15/01/2024
                                   ;MOV    DX,BX (*)             ; get high order size
```

```
13750                                   ;MOV    AX,DI (**)              ; get low order size
13751 00001D90 89D8                     mov     ax,bx ; hw of file size
13752                                   ;
13753                                   ;mov    bx,[si+0Eh]
13754 00001D92 8B5C0E                   MOV     BX,[SI+SYS_FCB.RECSIZ]; get his record size
13755 00001D95 09DB                     OR      BX,BX                   ; empty record => 0 size for file
13756 00001D97 7502                     JNZ     short GetSize           ; not empty
13757                                   ;MOV    BX,128
13758 00001D99 B380                     mov     bl,128  ; 15/01/2024
13759                                GetSize:
13760                                   ; 15/01/2024
13761                                   ;MOV    DI,AX                   ; save low order word
13762                                   ;MOV    AX,DX                   ; move high order for divide
13763                                   ;xchg   ax,dx ; (*)
13764                                   ; ax = hw of file size
13765
13766 00001D9B 31D2                     XOR     DX,DX                   ; clear out high
13767 00001D9D F7F3                     DIV     BX                      ; wham
13768 00001D9F 50                       PUSH    AX                      ; save dividend
13769 00001DA0 89F8                     MOV     AX,DI ; (**)            ; get low order piece
13770 00001DA2 F7F3                     DIV     BX                      ; wham
13771 00001DA4 89D1                     MOV     CX,DX                   ; save remainder
13772 00001DA6 5A                       POP     DX                      ; get high order dividend
13773 00001DA7 E306                     JCXZ    LengthStore             ; no roundup
13774 00001DA9 83C001                   ADD     AX,1
13775 00001DAC 83D200                   ADC     DX,0                    ; 32-bit increment
13776                                LengthStore:
13777                                   ;mov    [si+21h],ax
13778 00001DAF 894421                   MOV     [SI+SYS_FCB.RR],AX      ; store low order
13779                                   ;mov    [si+23h],dl
13780 00001DB2 885423                   MOV     [SI+SYS_FCB.RR+2],DL    ; store high order
13781 00001DB5 08F6                     OR      DH,DH
13782 00001DB7 74A8                     JZ      short GoodPath          ; not storing insignificant zero
13783                                   ;mov    [si+24h],dh
13784 00001DB9 887424                   MOV     [SI+SYS_FCB.RR+3],DH    ; save that high piece
13785                                   ; 16/12/2022
13786                                GoodRet:
13787                                   ;jmp    FCB_RET_OK
13788 00001DBC EBA3                     jmp     short GoodPath
13789
13790                                ;Break <$FCB_Close - close a file>
13791                                ;--------------------------------------------------------------------------
13792                                ;
13793                                ;   $FCB_Close - given an FCB, look up the SFN and close it. Do not free it
13794                                ;   as the FCB may be used for further I/O
13795                                ;
13796                                ;   Inputs: DS:DX point to FCB
13797                                ;   Outputs:       AL = FF if file was not found on disk
13798                                ;   Registers modified: all
13799                                ;
13800                                ;--------------------------------------------------------------------------
13801                                ;
13802                                   ; 16/01/2024 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMDOS.COM)
13803
13804                                _$FCB_CLOSE:            ; System call 16
13805
13806 00001DBE 30C0                     XOR     AL,AL                   ; default search attributes
13807 00001DC0 E88704                   call    GetExtended             ; DS:SI point to real FCB
13808 00001DC3 7403                     JZ      short NoAttr            ; not extended
13809 00001DC5 8A44FF                   MOV     AL,[SI-1]               ; get attributes
13810                                NoAttr:
13811                                                                   ; SS override
13812 00001DC8 36A2[6B05]               MOV     [SS:ATTRIB],AL          ; stash away found attributes
13813 00001DCC E8CE03                   call    SFTFromFCB
13814 00001DCF 72EB                     JC      short GoodRet           ; MZ 16 Jan Assume death
13815
13816                                   ; If the sharer is present, then the SFT is not regenable. Thus,
13817                                   ; there is no need to set the SFT's attribute.
13818
13819                                   ;;; 9/8/86 F.C. save SFT attribute and restore it back when close is
13820                                   ;;; done
13821
13822                                   ;mov    al,[es:di+4]
13823 00001DD1 268A4504                 MOV     AL,[ES:DI+SF_ENTRY.sf_attr]
13824 00001DD5 30E4                     XOR     AH,AH
13825 00001DD7 50                       PUSH    AX
13826
13827                                   ;;; 9/8/86 F.C. save SFT attribute and restore it back when close is
13828                                   ;;; done
13829
13830 00001DD8 E85F66                   call    CheckShare
13831 00001DDB 7508                     JNZ     short NoStash
13832 00001DDD 36A0[6B05]               MOV     AL,[SS:ATTRIB]
13833                                   ;mov    [es:di+4],al
13834 00001DE1 26884504                 MOV     [ES:DI+SF_ENTRY.sf_attr],AL ; attempted attribute for close
13835                                NoStash:
13836
13837                                ; 16/01/2024
13838                                %if 0
13839                                   ;mov    ax,[si+14h]
13840                                   MOV     AX,[SI+SYS_FCB.FDATE] ; move in the time and date
13841                                   ;mov    [es:di+0Fh],ax
13842                                   MOV     [ES:DI+SF_ENTRY.sf_date],AX
13843                                   ;mov    ax,[si+16h]
13844                                   MOV     AX,[SI+SYS_FCB.FTIME]
13845                                   ;mov    [es:di+0Dh],ax
13846                                   MOV     [ES:DI+SF_ENTRY.sf_time],AX
13847                                   ;mov    ax,[si+10h]
13848                                   MOV     AX,[SI+SYS_FCB.FILSIZ]
13849                                   ;mov    [es:di+11h],ax
13850                                   MOV     [ES:DI+SF_ENTRY.sf_size],AX
13851                                   ;mov    ax,[si+12h]
13852                                   MOV     AX,[SI+SYS_FCB.FILSIZ+2]
13853                                   ;mov    [es:di+13h],ax
13854                                   MOV     [ES:DI+SF_ENTRY.sf_size+2],AX
13855                                   ;or     word [es:di+5],4000h
13856                                   ; 17/12/2022
13857                                   or      byte [ES:DI+SF_ENTRY.sf_flags+1],(sf_close_nodate>>8) ; 40h
13858                                   ;OR     word [ES:DI+SF_ENTRY.sf_flags],sf_close_nodate
13859                                %else
13860                                   ; 16/01/2024 (PCDOS 7.1 IBMDOS.COM)
13861 00001DE5 1E                       push    ds
13862                                   ;lds    ax,[si+14h]
13863 00001DE6 C54414                   lds     ax,[si+SYS_FCB.FDATE] ; move in the time and date
13864                                   ;mov    [es:di+0Fh],ax
13865 00001DE9 2689450F                 mov     [es:di+SF_ENTRY.sf_date],ax
13866                                   ;mov    [es:di+0Dh],ds
13867 00001DED 268C5D0D                 mov     [es:di+SF_ENTRY.sf_time],ds
13868 00001DF1 1F                       pop     ds
13869                                   ;lds    ax,[si+10h]
13870 00001DF2 C54410                   lds     ax,[si+SYS_FCB.FILSIZ]
13871                                   ;mov    [es:di+11h],ax
13872 00001DF5 26894511                 mov     [es:di+SF_ENTRY.sf_size],ax
13873                                   ;mov    [es:di+13h],ds
```

```
13874 00001DF9 268C5D13                 mov     [es:di+SF_ENTRY.sf_size+2],ds
13875                                   ; 16/01/2024
13876                                   ;;or    word [es:di+5],4000h
13877                                   ;or     word [es:di+SF_ENTRY.sf_flags],sf_close_nodate
13878 00001DFD 26804D0640              or      byte [es:di+SF_ENTRY.sf_flags+1],(sf_close_nodate>>8) ; 40h
13879                                 %endif
13880
13881 00001E02 16                       push    ss
13882 00001E03 1F                       pop     ds
13883 00001E04 E82119                   call    DOS_CLOSE       ; wham
13884 00001E07 C43E[9E05]               LES     DI,[THISSFT]
13885
13886                                   ;;; 9/8/86 F.C. restore SFT attribute
13887 00001E0B 59                       POP     CX
13888                                   ;mov    [es:di+4],cl
13889 00001E0C 26884D04                 MOV     [ES:DI+SF_ENTRY.sf_attr],CL
13890                                   ;;; 9/8/86 F.C. restore SFT attribute
13891
13892 00001E10 9C                       PUSHF
13893                                   ;test   word [es:di],0FFFFh
13894                                   ;cmp    word [ES:DI+SF_ENTRY.sf_ref_count],0
13895                                                   ; zero ref count gets blasted
13896 00001E11 26833D00                 cmp     word [ES:DI],0
13897 00001E15 7507                     jnz      short CloseOK
13898 00001E17 50                       PUSH    AX
13899 00001E18 B04D                     MOV     AL,'M' ; 4Dh
13900 00001E1A E8FB02                   call    BlastSFT
13901 00001E1D 58                       POP     AX
13902                                 CloseOK:
13903 00001E1E 9D                       POPF
13904 00001E1F 739B                     JNC     short GoodRet
13905                                   ;cmp    al,6
13906 00001E21 3C06                     CMP     AL,error_invalid_handle
13907 00001E23 7497                     JZ      short GoodRet
13908                                   ;mov    al,2
13909 00001E25 B002                     MOV     AL,error_file_not_found
13910                                 fren90:
13911                                   ; 16/12/2022
13912                                 fcb_close_err:
13913 00001E27 E963E8                   jmp     FCB_RET_ERR
13914
13915                                 ;
13916                                 ;--------------------------------------------------------------------------
13917                                 ;
13918                                 ;** $FCB_Rename - Rename a File
13919                                 ;
13920                                 ;   $FCB_Rename - rename a file in place within a directory. Renames
13921                                 ;   multiple files copying from the meta characters.
13922                                 ;
13923                                 ;   ENTRY   DS:DX point to an FCB. The normal name field is the source
13924                                 ;               name of the files to be renamed. Starting at offset 11h
13925                                 ;               in the FCB is the destination name.
13926                                 ;   EXIT    AL = 0 -> no error occurred and all files were renamed
13927                                 ;           AL = FF -> some files may have been renamed but:
13928                                 ;                   rename to existing file or source file not found
13929                                 ;   USES    ALL
13930                                 ;
13931                                 ;--------------------------------------------------------------------------
13932                                 ;
13933                                   ; 08/11/2022 Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
13934                                   ; 16/01/2024 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMDOS.COM)
13935
13936                                 _$FCB_RENAME:             ; System call 23
13937
13938 00001E2A E81D04                   call    GetExtended             ; get pointer to real FCB
13939 00001E2D 52                       push    dx
13940 00001E2E 8A04                     MOV     AL,[SI]                 ; get drive byte
13941 00001E30 83C610                   ADD     SI,10h                  ; point to destination
13942
13943                                                                   ; RenBuf is in DOSDATA
13944 00001E33 BF[3E04]                 MOV     DI,RENBUF               ; point to destination buffer
13945 00001E36 FF34                     push    word [SI]
13946 00001E38 1E                       push    ds
13947                                   ;push   di                      ; save source pointer for TransFCB
13948                                   ; 16/01/2024 (Retro DOS v4 BugFix!)
13949 00001E39 56                       push    si
13950 00001E3A 8804                     MOV     [SI],AL                 ; drop in real drive
13951 00001E3C 89F2                     MOV     DX,SI                   ; let TransFCB know where the FCB is
13952 00001E3E E8BB5D                   call    TransFCB                ; munch this pathname
13953 00001E41 5E                       pop     si
13954 00001E42 1F                       pop     ds
13955 00001E43 8F04                     pop     WORD [SI]               ; get path back
13956 00001E45 5A                       pop     dx                      ; Original FCB pointer
13957 00001E46 72DF                     JC      short fren90            ; bad path -> error
13958
13959                                                                   ; SS override for WFP_Start & Ren_WFP
13960 00001E48 368B36[B205]             MOV     SI,[ss:WFP_START]       ; get pointer
13961 00001E4D 368936[B405]             MOV     [ss:REN_WFP],SI                 ; stash it
13962
13963                                                                   ; OpenBuf is in DOSDATA
13964 00001E52 BF[BE03]                 MOV     DI,OPENBUF              ; appropriate spot
13965 00001E55 E8A45D                   call    TransFCB                ; wham
13966                                                                   ; NOTE that this call is pointing
13967                                                                   ;  back to the ORIGINAL FCB so
13968                                                                   ;  SATTRIB gets set correctly
13969 00001E58 72CD                     JC      short fren90            ; error
13970                                   ;;;
13971                                   ; 16/01/2024 - Retro DOS 5.0
13972                                   ; (PCDOS 7.1 IBMDOS.COM)
13973 00001E5A 36C606[5411]43           mov     byte [ss:PATHNAMELEN], 67 ; DIRSTRLEN = 67
13974                                   ;mov    word [ss:PATHNAMELEN], 67 ; set pathname length to 67
13975                                   ;;;
13976 00001E60 E8140F                   call    DOS_RENAME
13977 00001E63 72C2                     JC      short fren90
13978                                   ; 16/12/2022
13979 00001E65 E922E8                   jmp     FCB_RET_OK
13980
13981                                 ;   Error -
13982                                 ;
13983                                 ;   (al) = error code
13984
13985                                   ; 16/12/2022
13986                                 ;fren90:
13987                                 ;   ;jmp    FCB_RET_ERR
13988                                 ;   ; 08/11/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
13989                                 ;   jmp     short fcb_close_err
13990
13991                                 ;Break <Misbehavior fixers>
13992                                 ;
13993                                 ;   FCBs suffer from several problems. First, they are maintained in the
13994                                 ;   user's space so he may move them at will. Second, they have a small
13995                                 ;   reserved area that may be used for system information. Third, there was
13996                                 ;   never any "rules for behavior" for FCBs; there was no protocol for their
13997                                 ;   usage.
```

```
13998                                   ;
13999                                   ;   This results in the following misbehavior:
14000                                   ;
14001                                   ;   infinite opens of the same file:
14002                                   ;
14003                                   ;   While (TRUE) {                While (TRUE) {
14004                                   ;       FCBOpen (FCB);                FCBOpen (FCB);
14005                                   ;       Read (FCB);                   Write (FCB);
14006                                   ;       }                             }
14007                                   ;
14008                                   ;   infinite opens of different files:
14009                                   ;
14010                                   ;   While (TRUE) {                While (TRUE) {
14011                                   ;       FCBOpen (FCB[i++]);           FCBOpen (FCB[i++]);
14012                                   ;       Read (FCB);                   Write (FCB);
14013                                   ;       }                             }
14014                                   ;
14015                                   ;   multiple closes of the same file:
14016                                   ;
14017                                   ;   FCBOpen (FCB);
14018                                   ;   while (TRUE)
14019                                   ;       FCBClose (FCB);
14020                                   ;
14021                                   ;   I/O after closing file:
14022                                   ;
14023                                   ;   FCBOpen (FCB);
14024                                   ;   while (TRUE) {
14025                                   ;       FCBWrite (FCB);
14026                                   ;       FCBClose (FCB);
14027                                   ;       }
14028                                   ;
14029                                   ;   The following is am implementation of a methodology for emulating the
14030                                   ;   above with the exception of I/O after close. We are NOT attempting to
14031                                   ;   resolve that particular misbehavior. We will enforce correct behaviour in
14032                                   ;   FCBs when they refer to a network file or when there is file sharing on
14033                                   ;   the local machine.
14034                                   ;
14035                                   ;   The reserved fields of the FCB (10 bytes worth) is divided up into various
14036                                   ;   structures depending on the file itself and the state of operations of the
14037                                   ;   OS. The information contained in this reserved field is enough to
14038                                   ;   regenerate the SFT for the local non-shared file. It is assumed that this
14039                                   ;   regeneration procedure may be expensive. The SFT for the FCB is
14040                                   ;   maintained in a LRU cache as the ONLY performance inprovement.
14041                                   ;
14042                                   ;   No regeneration of SFTs is attempted for network FCBs.
14043                                   ;
14044                                   ;   To regenerate the SFT for a local FCB, it is necessary to determine if the
14045                                   ;   file sharer is working. If the file sharer is present then the SFT is not
14046                                   ;   regenerated.
14047                                   ;
14048                                   ;   Finally, if there is no local sharing, the full name of the file is no
14049                                   ;   longer available. We can make up for this by using the following
14050                                   ;   information:
14051                                   ;
14052                                   ;   The Drive number (from the DPB).
14053                                   ;   The physical sector of the directory that contains the entry.
14054                                   ;   The relative position of the entry in the sector.
14055                                   ;   The first cluster field.
14056                                   ;   The last used SFT.
14057                                   ;      OR In the case of a device FCB
14058                                   ;   The low 6 bits of sf_flags (indicating device type)
14059                                   ;   The pointer to the device header
14060                                   ;
14061                                   ;   We read in the particular directory sector and examine the indicated
14062                                   ;   directory entry. If it matches, then we are kosher; otherwise, we fail.
14063                                   ;
14064                                   ;   Some key items need to be remembered:
14065                                   ;
14066                                   ;   Even though we are caching SFTs, they may contain useful sharing
14067                                   ;   information. We enforce good behavior on the FCBs.
14068                                   ;
14069                                   ;   Network support must not treat FCBs as impacting the ref counts on
14070                                   ;   open VCs. The VCs may be closed only at process termination.
14071                                   ;
14072                                   ;   If this is not an installed version of the DOS, file sharing will
14073                                   ;   always be present.
14074                                   ;
14075                                   ;   We MUST always initialize lstclus to = firclus when regenerating a
14076                                   ;   file. Otherwise we start allocating clusters up the wazoo.
14077                                   ;
14078                                   ;   Always initialize, during regeneration, the mode field to both isFCB
14079                                   ;   and open_for_both. This is so the FCB code in the sharer can find the
14080                                   ;   proper OI record.
14081                                   ;
14082                                   ;   The test bits are:
14083                                   ;
14084                                   ;   00 -> local file
14085                                   ;   40 -> sharing local
14086                                   ;   80 -> network
14087                                   ;   C0 -> local device
14088                                   ;
14089                                   ;Break     <SaveFCBInfo - store pertinent information from an SFT into the FCB>
14090                                   ;--------------------------------------------------------------------------
14091                                   ;
14092                                   ;   SaveFCBInfo - given an FCB and its associated SFT, copy the relevant
14093                                   ;   pieces of information into the FCB to allow for subsequent
14094                                   ;   regeneration. Poke LRU also.
14095                                   ;
14096                                   ;   Inputs: ThisSFT points to a complete SFT.
14097                                   ;           DS:SI point to the FCB (not an extended one)
14098                                   ;   Outputs:       The relevant reserved fields in the FCB are filled in.
14099                                   ;           DS:SI preserved
14100                                   ;           ES:DI point to sft
14101                                   ;   Registers modified: All
14102                                   ;
14103                                   ;
14104                                   ;--------------------------------------------------------------------------
14105                                   ;
14106
14107                                       ; 08/11/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
14108                                       ; 20/01/2024 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMDOS.COM)
14109
14110                                   SaveFCBInfo:
14111
14112 00001E68 36C43E[9E05]               LES     DI,[SS:THISSFT]             ; SS override
14113 00001E6D E8D2F9                     call    IsSFTNet
14114 00001E70 740B                       JZ      short SaveLocal             ; if not network then save local info
14115                                   ;
14116                                   ;----- In net support -----
14117                                   ;
14118                                       ; 17/05/2019 - Retro DOS v4.0
14119
14120                                       ; MSDOS 3.3
14121                                       ;;mov   ax,[es:di+1Dh]
```

```
                                    ;mov    ax,[es:di+SF_ENTRY.sf_dirsec]
                                    ;;mov   [si+1Ah],ax
                                    ;mov    [si+fcb_net_handle],ax
                                    ;push   es
                                    ;push   di
                                    ;;les   di,[es:di+19h]
                                    ;LES    DI,[ES:DI+sf_netid]
                                    ;;mov   [si+1Ch],di
                                    ;MOV    [SI+fcb_netID],DI     ; save net ID
                                    ;;mov   [si+1Eh],es
                                    ;MOV    [SI+fcb_netID+2],ES
                                    ;pop    di
                                    ;pop    es

                                    ; MSDOS 6.0
                                    ;mov    ax,[es:di+0Bh]
00001E72 268B450B                   MOV     AX,[ES:DI+sf_serial_ID] ;AN000;;IFS. save IFS ID
                                    ;mov    [si+1Ch],ax
00001E76 89441C                     MOV     [SI+fcb_netID],ax     ;AN000;;IFS.

                                    ;mov    bl,80h
00001E79 B380                       MOV     BL,FCBNETWORK
                                ;
                                ;----- END In net support -----
                                ;
00001E7B EB63                       jmp     SHORT SaveSFN

                                SaveLocal:
                                    ;IF     Installed
00001E7D E8BA65                     call    CheckShare
                                    ;JZ     short SaveNoShare     ; no sharer
                                    ;JMP    short SaveShare               ; sharer present
                                    ; 16/12/2022
                                    ; 28/07/2019
00001E80 7559                       jnz     short SaveShare
                                    ; 08/11/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
                                    ;JZ     short SaveNoShare     ; no sharer
                                    ;JMP    short SaveShare               ; sharer present

                                SaveNoShare:
                                    ;;test  word [es:di+5],80h
                                    ;TEST   word [ES:DI+SF_ENTRY.sf_flags],devid_device
00001E82 26F6450580                 test    byte [ES:DI+SF_ENTRY.sf_flags],devid_device ; 80h
00001E87 7542                       JNZ     short SaveNoShareDev  ; Device

                                    ; Save no sharing local file information

                                    ;;mov   ax,[es:di+1Dh]  ; MSDOS 3.3
                                    ;mov    ax,[es:di+1Bh]  ; MSDOS 6.0
00001E89 268B451B                   MOV     AX,[ES:DI+SF_ENTRY.sf_dirsec] ; get directory sector F.C.
                                    ;mov    [si+1Dh],ax
00001E8D 89441D                     MOV     [SI+fcb_nsl_dirsec],AX

                                    ; MSDOS 6.0

                                    ;SR; Store high byte of directory sector
                                    ;mov    ax,[es:di+1Dh]
00001E90 268B451D                   mov     ax,[es:di+SF_ENTRY.sf_dirsec+2] ; get high word

                                    ; SR;
                                    ; We have to store the read-only and archive attributes of the file.
                                    ; We extract it from the SFT and store it in the top two bits of the
                                    ; sector number ( sector number == 22 bits only )

                                    ;mov    bl,[es:di+4]
00001E94 268A5D04                   mov     bl,[es:di+SF_ENTRY.sf_attr]
00001E98 88DF                       mov     bh,bl
00001E9A D0CB                       ror     bl,1
00001E9C D0E7                       shl     bh,1
00001E9E 08FB                       or      bl,bh
00001EA0 80E3C0                     and     bl,0C0h
00001EA3 08D8                       or      al,bl
                                    ;mov    [si+18h],al ; 08/11/2022
00001EA5 884418                     mov     [si+fcb_sfn],al       ; sector number = 22 bits

                                    ; MSDOS 6.0 (& MSDOS 3.3)
                                    ;mov    al,[es:di+1Fh]
00001EA8 268A451F                   MOV     AL,[ES:DI+SF_ENTRY.sf_dirpos] ; location in sector
                                    ;mov    [si+1Fh],al
00001EAC 88441F                     MOV     [SI+fcb_nsl_dirpos],AL

                                    ; 20/01/2024 - Retro DOS v5.0 (PCDOS 7.1 IBMDOS.COM)
                                    ;;;
                                    ;;mov   ax,[es:di+0Bh] ; .sf_firclus:
                                    ;MOV    AX,[ES:DI+SF_ENTRY.sf_firclus] ; first cluster
                                    ; 20/01/2024
                                    ; (PCDOS 7.1 IBMDOS.COM - DOSCODE:5DF5h)
                                    ; (Windows ME IO.SYS - BIOSCODE:5D60h)
00001EAF 268B452B                   mov     ax,[es:di+2Bh]  ; .sf_chain !!! (MSDOS 6.22)
                                    ;mov    ax,[es:di+SF_ENTRY.sf_chain] ; first cluster (32 bit) !?
                                    ;;;
                                    ;mov    [si+1Bh],ax
00001EB3 89441B                     MOV     [SI+fcb_nsl_firclus],AX
00001EB6 B300                       MOV     BL,0

                                    ; Create the bits field from the dirty/device bits of the flags word
                                    ; and the mode byte

                                SetFCBBits:
                                    ;mov    ax,[es:di+5]
00001EB8 268B4505                   MOV     AX,[ES:DI+SF_ENTRY.sf_flags]
00001EBC 24C0                       AND     AL,0C0h               ; mask off drive bits
                                    ;or     al,[es:di+2]
00001EBE 260A4502                   OR      AL,[ES:DI+SF_ENTRY.sf_mode] ; stick in open mode
                                    ;mov    [si+1Ah], al
00001EC2 88441A                     MOV     [SI+fcb_nsl_bits],AL  ; save dirty info

                                    ; MSDOS 6.0

                                    ; SR;
                                    ; Check if we came here for local file or device. If for local file,
                                    ; skip setting of SFT index

00001EC5 08DB                       or      bl,bl
00001EC7 7428                       jz      short SaveNoSFN               ; do not save SFN if local file

00001EC9 EB15                       JMP     short SaveSFN         ; go and save SFN

                                    ; Save no sharing local device information

                                SaveNoShareDev:
                                    ; 20/01/2024
                                    ;;mov   ax,[es:di+7]
                                    ;MOV    AX,[ES:DI+SF_ENTRY.sf_devptr]
```

```
                                    ;;mov   [si+1Ah],ax
                                    ;MOV    [SI+fcb_nsld_drvptr],AX
                                    ;;mov   ax,[es:di+9]
                                    ;MOV    AX,[ES:DI+SF_ENTRY.sf_devptr+2]
                                    ;MOV    [SI+fcb_nsld_drvptr+2],AX
                                    ; 20/01/2024 (PCDOS 7.1 IBMDOS.COM)
00001ECB 06                         push    es
00001ECC 26C44507                   les     ax,[es:di+SF_ENTRY.sf_devptr]
00001ED0 89441A                     mov     [si+fcb_nsld_drvptr],ax
00001ED3 8C441C                     mov     [si+fcb_nsld_drvptr+2],es
00001ED6 07                         pop     es

                                    ;mov    bl,40h
00001ED7 B340                       MOV     BL,FCBDEVICE
                                    ; 28/12/2022
00001ED9 EBDD                       JMP     short SetFCBBits      ; go and save SFN

                                SaveShare:
                                    ;ENDIF

                                ;----- In share support -----

                                    ;call   far [ss:ShSave]
00001EDB 36FF1E[B800]               Call    far [ss:JShare+(10*4)] ; 10 = ShSave ; SS Override

                                ;----- end in share support -----

                                    ; 17/05/2019

                                SaveSFN:
                                    ;lea    ax,[di-6]
00001EE0 8D45FA                     LEA     AX,[DI-SFT.SFTable]

                                    ; Adjust for offset to table.

00001EE3 362B06[4000]               SUB     AX,[SS:SFTFCB]        ; SS override for SftFCB

00001EE8 53                         push    bx                    ;bx = FCB type (net/Share or local)
                                    ;;mov   bl,53 ; MSDOS 3.3
                                    ;mov    bl,59 ; MSDOS 6.0
00001EE9 B33B                       MOV     BL,SF_ENTRY.size
00001EEB F6F3                       DIV     BL
                                    ;mov    [si+18h],al
00001EED 884418                     MOV     [SI+fcb_sfn],AL               ; last used SFN
00001EF0 5B                         pop     bx                    ;restore bx

                                SaveNoSFN:
                                    ;mov    ax,[es:di+5]
00001EF1 268B4505                   MOV     AX,[ES:DI+SF_ENTRY.sf_flags]
00001EF5 243F                       AND     AL,3Fh                ; get real drive
00001EF7 08D8                       OR      AL,BL
                                    ;mov    [si+19h],al
00001EF9 884419                     MOV     [SI+fcb_l_drive],AL

00001EFC 36A1[1000]                 MOV     AX,[SS:FCBLRU]        ; get lru count
00001F00 40                         INC     AX
                                    ;mov    [es:di+15h],ax
00001F01 26894515                   MOV     [ES:DI+sf_LRU],AX
00001F05 7506                       JNZ     short SimpleStuff

                                    ; lru flag overflowed. Run through all FCB sfts and adjust:
                                    ; LRU < 8000H get set to 0. Others -= 8000h. This LRU = 8000h

                                    ;mov    bx,15h
00001F07 BB1500                     MOV     BX,SF_ENTRY.sf_position
00001F0A E80500                     call    ResetLRU

                                    ; Set new LRU to AX
                                SimpleStuff:
00001F0D 36A3[1000]                 MOV     [SS:FCBLRU],AX
00001F11 C3                         retn

                                ;Break      <ResetLRU - reset overflowed lru counts>
                                ;-----------------------------------------------------------------------------
                                ;
                                ;   ResetLRU - during lru updates, we may wrap at 64K. We must walk the
                                ;   entire set of SFTs and subtract 8000h from their lru counts and truncate
                                ;   at 0.
                                ;
                                ;   Inputs: BX is offset into SFT field where lru firld is kept
                                ;           ES:DI point to SFT currently being updated
                                ;   Outputs:       All FCB SFTs have their lru fields truncated
                                ;           AX has 8000h
                                ;   Registers modified: none
                                ;
                                ;-----------------------------------------------------------------------------
                                ;

                                    ; 17/05/2019 - Retro DOS v4.0
                                ResetLRU:
                                    ; ResetLRU is only called from fcbio.asm. So SS can be assumed to be
                                    ; DOSDATA

00001F12 B80080                     MOV     AX,8000h
00001F15 06                         push    es
00001F16 57                         push    di
                                    ;LES    DI,[CS:SFTFCB]        ; get pointer to head
00001F17 36C43E[4000]               LES     DI,[SS:SFTFCB] ; MSDOS 6.0
                                    ;mov    cx,[es:di+4]
00001F1C 268B4D04                   MOV     CX,[ES:DI+SFT.SFCount]
                                    ;lea    di,[di+6]
00001F20 8D7D06                     LEA     DI,[DI+SFT.SFTable]   ; point at table
                                ovScan:
00001F23 262901                     SUB     [ES:DI+BX],AX         ; decrement lru count
00001F26 7703                       JA      short ovLoop
00001F28 268901                     MOV     [ES:DI+BX],AX         ; truncate at 0
                                ovLoop:
                                    ;;add   di,53   ; MSDOS 3.3
                                    ;add    di,59   ; MSDOS 6.0
00001F2B 83C73B                     ADD     DI,SF_ENTRY.size      ; advance to next
00001F2E E2F3                       LOOP    ovScan
00001F30 5F                         pop     di
00001F31 07                         pop     es
00001F32 268901                     MOV     [ES:DI+BX],AX
00001F35 C3                         retn

                                ;IF  0  ; We dont need this routine any more.
                                ;
                                ;Break      <SetOpenAge - update the open age of a SFT>
                                ;-----------------------------------------------------------------------------
                                ;
                                ;   SetOpenAge - In order to maintain the first N open files in the FCB cache,
                                ;   we keep the 'open age' or an LRU count based on opens. We update the
                                ;   count here and fill in the appropriate field.
```

```
                        ;
                        ;   Inputs: ES:DI point to SFT
                        ;   Outputs:       ES:DI has the open age field filled in.
                        ;           If open age has wraparound, we will have subtracted 8000h
                        ;                from all open ages.
                        ;   Registers modified: AX
                        ;
                        ;--------------------------------------------------------------------------
                        ;
                        ;SetOpenAge:
                        ;   ; 20/07/2018 - Retro DOS v3.0
                        ;   ; MSDOS 3.3 - IBMDOS.COM, Offset 2597h
                        ;   ; (& MSDOS 6.0, FCBIO.ASM)
                        ;
                        ;   ; SetOpenAge is called from fcbio2.asm. SS can be assumed to be valid.
                        ;
                        ;   MOV     AX,[CS:OpenLRU]       ; SS override
                        ;   INC     AX
                        ;   ;mov    [es:di+17h],ax
                        ;   MOV     [ES:DI+sf_OpenAge],AX
                        ;   JNZ     short SetDone
                        ;   ;mov    bx,17h
                        ;   MOV     BX,SF_ENTRY.sf_position+2 ; mov bx,sf_OpenAge
                        ;   call    ResetLRU
                        ;SetDone:
                        ;   MOV     [CS:OpenLRU],AX
                        ;   retn
                        ;
                        ;ENDIF      ; SetOpenAge no longer needed

                        ; 21/07/2018 - Retro DOS v3.0
                        ; LRUFCB for MSDOS 6.0 !

                        ;Break      <LRUFCB - perform LRU on FCB sfts>
                        ;--------------------------------------------------------------------------
                        ;
                        ;   LRUFCB - find LRU fcb in cache. Set ThisSFT and return it. We preserve
                        ;   the first keepcount sfts if they are network sfts or if sharing is
                        ;   loaded.  If carry is set then NO BLASTING is NECESSARY.
                        ;
                        ;   Inputs: none
                        ;   Outputs:       ES:DI point to SFT
                        ;           ThisSFT points to SFT
                        ;           SFT is zeroed
                        ;           Carry set of closes failed
                        ;   Registers modified: none
                        ;
                        ;--------------------------------------------------------------------------
                        ;
                        ; MSDOS 6.0
                        ;IF 0       ; rewritten this routine
                        ;
                        ;LRUFCB: ; MSDOS 3.3 - IBMDOS.COM (1987) - Offset 25ADh
                        ;   call    save_world
                        ;
                        ; Find nth oldest NET/SHARE FCB. We want to find its age for the second scan
                        ; to find the lease recently used one that is younger than the open age.  We
                        ; operate be scanning the list n times finding the least age that is greater
                        ; or equal to the previous minimum age.
                        ;
                        ;   BP is the count of times we need to go through this loop.
                        ;   AX is the current acceptable minimum age to consider
                        ;
                        ;   mov     bp,[CS:KEEPCOUNT]     ; k = keepcount;
                        ;   XOR     AX,AX                 ; low = 0;
                        ;
                        ; If we've scanned the table n times, then we are done.
                        ;
                        ;lru1:
                        ;   CMP     bp,0                  ; while (k--) {
                        ;   JZ      short lru75
                        ;   DEC     bp
                        ;
                        ; Set up for scan.
                        ;
                        ;   AX is the minimum age for consideration
                        ;   BX is the minimum age found during the scan
                        ;   SI is the position of the entry that corresponds to BX
                        ;
                        ;   MOV     BX,-1                 ;     min = 0xffff;
                        ;   MOV     si,BX                 ;     pos = 0xffff;
                        ;   LES     DI,[CS:SFTFCB]        ;     for (CX=FCBCount; CX>0; CX--)
                        ;   ;mov    cx,[es:di+4]
                        ;   MOV     CX,[ES:DI+SFT.SFCount]
                        ;   ;lea    di,[di+6]
                        ;   LEA     DI,[DI+SFT.SFTable]
                        ;
                        ; Innermost loop.  If the current entry is free, then we are done.  Or, if the
                        ; current entry is busy (indicating a previous aborted allocation), then we
                        ; are done.  In both cases, we use the found entry.
                        ;
                        ;lru2:
                        ;   cmp     word [es:di],0
                        ;   ;cmp    word [es:di+SF_ENTRY.sf_ref_count],0
                        ;   jz      short lru25
                        ;   ;cmp    word [es:di],-1
                        ;   ;cmp    word [es:di+SF_ENTRY.sf_ref_count],sf_busy
                        ;   cmp     word [es:di],sf_busy
                        ;   jnz     short lru3
                        ;
                        ; The entry is usable without further scan.  Go and use it.
                        ;
                        ;lru25:
                        ;   MOV     si,DI                 ;             pos = i;
                        ;   JMP     short lru11           ;             goto got;
                        ;
                        ; See if the entry is for the network or for the sharer.
                        ;
                        ;  If for the sharer or network then
                        ;   if the age < current minimum AND >= allowed minimum then
                        ;       this entry becomes current minimum
                        ;
                        ;lru3:
                        ;   ;test   word [es:di+5],8000h
                        ;   TEST    word [ES:DI+SF_ENTRY.sf_flags],sf_isnet
                        ;                                 ;         if (!net[i]
                        ;   JNZ     short lru35
                        ;if installed
                        ;   call    CheckShare            ;              && !sharing)
                        ;   JZ      short lru5            ;         else
                        ;ENDIF
                        ;
                        ; This SFT is for the net or is for the sharer. See if it less than the
                        ; current minimum.
```

```
14494                                   ;
14495                                   ;lru35:
14496                                   ;   ;mov    dx,[es:di+17h]
14497                                   ;   MOV     DX,[ES:DI+sf_OpenAge]
14498                                   ;   CMP     DX,AX                   ;        if (age[i] >= low &&
14499                                   ;   JB      short lru5
14500                                   ;   CMP     DX,BX
14501                                   ;   JAE     short lru5              ;            age[i] < min) {
14502                                   ;
14503                                   ; entry is new minimum.  Remember his age.
14504                                   ;
14505                                   ;   mov     bx,DX                   ;            min = age[i];
14506                                   ;   mov     si,di                   ;            pos = i;
14507                                   ;
14508                                   ; End of loop.     gp back for more
14509                                   ;
14510                                   ;lru5:
14511                                   ;   ;add    di,53
14512                                   ;   add     di,SF_ENTRY.size
14513                                   ;   loop    lru2                    ;            }
14514                                   ;
14515                                   ; The scan is complete. If we have successfully found a new minimum (pos != -1)
14516                                   ; set then threshold value to this new minimum + 1. Otherwise, the scan is
14517                                   ; complete.  Go find LRU.
14518                                   ;
14519                                   ;lru6:
14520                                   ;   cmp     si,-1                   ; position not -1?
14521                                   ;   jz      short lru75             ; no, done with everything
14522                                   ;   lea     ax,[bx+1]               ; set new threshold age
14523                                   ;   jmp     short lru1              ; go and loop for more
14524                                   ;lru65:
14525                                   ;   stc
14526                                   ;   jmp     short LRUDead           ;        return -1;
14527                                   ;
14528                                   ; Main loop is done. We have AX being the age+1 of the nth oldest sharer or
14529                                   ; network entry. We now make a second pass through to find the LRU entry
14530                                   ; that is local-no-share or has age >= AX
14531                                   ;
14532                                   ;lru75:
14533                                   ;   mov     bx,-1                   ; min = 0xffff;
14534                                   ;   mov     si,bx                   ; pos = 0xffff;
14535                                   ;   LES     DI,[CS:SFTFCB]          ; for (CX=FCBCount; CX>0; CX--)
14536                                   ;   ;mov    cx,[es:di+4]
14537                                   ;   MOV     CX,[ES:DI+SFT.SFCount]
14538                                   ;   ;lea    di,[di+6]
14539                                   ;   LEA     DI,[DI+SFT.SFTable]
14540                                   ;
14541                                   ; If this is is local-no-share then go check for LRU else if age >= threshold
14542                                   ; then check for lru.
14543                                   ;
14544                                   ;lru8:
14545                                   ;   ;test   word [es:di+5],8000h
14546                                   ;   TEST    word [ES:DI+SF_ENTRY.sf_flags],sf_isnet
14547                                   ;   jnz     short lru85             ; is for network, go check age
14548                                   ;   call    CheckShare              ; sharer here?
14549                                   ;   jz      short lru86             ; no, go check lru
14550                                   ;
14551                                   ; Network or sharer.  Check age
14552                                   ;
14553                                   ;lru85:
14554                                   ;   ;cmp    [es:di+17h],ax
14555                                   ;   cmp     [es:di+sf_OpenAge],ax
14556                                   ;   jb      short lru9              ; age is before threshold, skip it
14557                                   ;
14558                                   ; Check LRU
14559                                   ;
14560                                   ;lru86:
14561                                   ;   ;cmp    [es:di+15h],bx
14562                                   ;   cmp     [es:di+sf_LRU],bx       ; is LRU less than current LRU?
14563                                   ;   jae     short lru9              ; no, skip this
14564                                   ;   mov     si,di                   ; remember position
14565                                   ;   ;mov    bx,[es:di+15h]
14566                                   ;   mov     bx,[es:di+sf_LRU]       ; remember new minimum LRU
14567                                   ;
14568                                   ; Done with this entry, go back for more.
14569                                   ;
14570                                   ;lru9:
14571                                   ;   ;add    di, 53
14572                                   ;   add     di,SF_ENTRY.size
14573                                   ;   loop    lru8
14574                                   ;
14575                                   ; Scan is complete. If we found NOTHING that satisfied us then we bomb
14576                                   ; out. The conditions here are:
14577                                   ;
14578                                   ;  No local-no-shares AND all net/share entries are older than threshold
14579                                   ;
14580                                   ;lru10:
14581                                   ;   cmp     si,-1                   ; if no one f
14582                                   ;   jz      short lru65             ;     return -1;
14583                                   ;lru11:
14584                                   ;   mov     di,si
14585                                   ;   MOV     [CS:THISSFT],DI              ; set thissft
14586                                   ;   MOV     [CS:THISSFT+2],ES
14587                                   ;
14588                                   ; If we have sharing or thisSFT is a net sft, then close it until ref count
14589                                   ; is 0.
14590                                   ;
14591                                   ;   ;test   word [es:di+5],8000h
14592                                   ;   TEST    word [ES:DI+SF_ENTRY.sf_flags],sf_isnet
14593                                   ;   JNZ     short LRUClose
14594                                   ;IF INSTALLED
14595                                   ;   call    CheckShare
14596                                   ;   JZ      short LRUDone
14597                                   ;ENDIF
14598                                   ;
14599                                   ; Repeat close until ref count is 0
14600                                   ;
14601                                   ;LRUClose:
14602                                   ;   push    ss
14603                                   ;   pop     ds
14604                                   ;   LES     DI,[THISSFT]
14605                                   ;   cmp     word [es:di],0
14606                                   ;   ;CMP    word [ES:DI+SFT.sf_ref_count],0 ; is ref count still <> 0?
14607                                   ;   JZ      short LRUDone           ; nope, all done
14608                                   ;   call    DOS_CLOSE
14609                                   ;   jnc     short LRUClose          ; no error => clean up
14610                                   ;   ;cmp    al,6
14611                                   ;   cmp     al,error_invalid_handle
14612                                   ;   jz      short LRUClose
14613                                   ;   stc
14614                                   ;   JMP     short LRUDead
14615                                   ;LRUDone:
14616                                   ;   XOR     AL,AL
14617                                   ;   call    BlastSFT                ; fill SFT with 0 (AL), 'C' cleared
```

```
                                ;
                                ;LRUDead:
                                ;   call    restore_world
                                ;   LES      DI,[CS:THISSFT]
                                ;   jnc     short LRUFCB_retn
                                ;LRUFCB_err:
                                ;   ; mov   al, 23h
                                ;   MOV     AL,error_FCB_unavailable
                                ;LRUFCB_retn:
                                ;   retn:
                                ;
                                ;ENDIF      ; LRUFCB has been rewritten below.

                                ; 17/05/2019 - Retro DOS v4.0
                                ; LRUFCB for MSDOS 6.0 !
                                ;--------------------------------------------------------------------------
                                ;
                                ; LruFCB -- allocate the LRU SFT from the SFT Table. The LRU scheme
                                ; maintains separate counts for net/Share and local SFTs. We allocate a
                                ; net/Share SFT only if we do not find a local SFT. This helps keep
                                ; net/Share SFTs which cannot be regenerated for as long as possible. We
                                ; optimize regeneration operations by keeping track of the current local
                                ; SFT. This avoids scanning of the SFTs as long as we have at least one
                                ; local SFT in the SFT Block.
                                ;
                                ; Inputs: al = 0 => Regenerate SFT operation
                                ;       = 1 => Allocate new SFT for Open/Create
                                ;
                                ; Outputs: Carry clear
                                ;          es:di = Address of allocated SFT
                                ;          ThisSFT = Address of allocated SFT
                                ;
                                ;     carry set if closes of net/Share files failed
                                ;          al = error_FCB_unavailable
                                ;
                                ; Registers affected: None
                                ;
                                ;--------------------------------------------------------------------------
                                ;LruFCB     PROC    NEAR
                                LRUFCB:
                                    ; 17/05/2019 - Retro DOS v4.0
                                    ; DOSCODE:5805h (MSDOS 6.21, MSDOS.SYS)

                                    ; 08/11/2022 Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
                                    ; DOSCODE:57F1h (MSDOS 5.0, MSDOS.SYS)

                                    ; 20/01/2024 Retro DOS v5.0 (Modified PCDOS 7.1 IBMMDOS.COM)
                                    ; DOSCODE:5E7Ch (PCDOS 7.1, IBMDOS.COM)

00001F36 06                         push    es      ; * (MSDOS 6.21)

00001F37 E81EE5                     call    save_world

                                    ;getdseg <ds>          ;ds = DOSDATA
00001F3A 2E8E1E[0700]               mov     ds,[cs:DosDSeg]

00001F3F 08C0                       or      al,al          ;Check if regenerate allocation
00001F41 7516                       jnz     short lru1     ;Try to find SFT to use

                                    ; This is a regen call. If LocalSFT contains the address of a valid
                                    ; local SFT, just return that SFT to reuse

                                    ; 20/01/2024
                                    ;mov    di,[LocalSFT]
                                    ;or     di,[LocalSFT+2]       ;is address == 0?
                                    ;jz     short lru1     ;invalid local SFT, find one

                                    ; We have found a valid local SFT. Recycle this SFT

00001F43 C43E[760F]                 les     di,[LocalSFT]

                                    ; 20/01/2024 (PCDOS 7.1 IBMDOS.COM)
00001F47 8CC1                       mov     cx,es
00001F49 09F9                       or      cx,di          ; is address == 0?
00001F4B 740C                       jz      short lru1     ; invalid local SFT, find one

                                gotlocalSFT:
00001F4D 893E[9E05]                 mov     [THISSFT],di
00001F51 8C06[A005]                 mov     [THISSFT+2],es
00001F55 F8                         clc
00001F56 E9A900                     jmp     LRUDone        ;clear up SFT and return

                                lru1:
00001F59 C43E[4000]                 les     di,[SFTFCB]    ;es:di = SF Table for FCBs
                                    ;mov    cx,[es:di+4]
00001F5D 268B4D04                   mov     cx,[es:di+SFT.SFCount];cx = number of SFTs
                                    ;lea    di,[di+6]
00001F61 8D7D06                     lea     di,[di+SFT.SFTable]   ;es:di = first SFT

                                    ; We scan through all the SFTs scanning for a free one. It also
                                    ; remembers the LRU SFT for net/Share SFTs and local SFTs separately.
                                    ; bx = min. LRU for local SFTs
                                    ; si = pos. of local SFT with min. LRU
                                    ; dx = min. LRU for net/Share SFTs
                                    ; bp = pos. of net/Share SFT with min. LRU

00001F64 BBFFFF                     mov     bx,-1          ; init. to 0xffff ( max. LRU value )
00001F67 89DE                       mov     si,bx
00001F69 89DA                       mov     dx,bx
00001F6B 89DD                       mov     bp,bx

                                findSFT:
                                    ;See if this SFT is a free one. If so, return it
00001F6D 26830D00                   or      word [es:di],0
                                    ;or     word [es:di+SF_ENTRY.sf_ref_count],0 ;reference count = 0 ?
00001F71 744C                       jz      short gotSFT   ;yes, SFT is free
                                    ;;cmp   word [es:di],-1
                                    ;cmp    word [es:di+SF_ENTRY.sf_ref_count],sf_busy ;Is it busy?
00001F73 26833DFF                   cmp     word [es:di],sf_busy ; -1
00001F77 7446                       jz      short gotSFT   ;no, can use it

                                    ; Check if this SFT is local and store its address in LocalSFT. Can be
                                    ; used for a later regen.

                                    ; 16/12/2022
                                    ; 08/11/2022
                                    ;test   byte [es:di+6],80h
00001F79 26F6450680                 test    byte [es:di+SF_ENTRY.sf_flags+1],(sf_isnet>>8) ; 80h
                                    ; 08/11/2022 Retro DOS v4.0 (MSDOS 5.0 MSDOS.SYS compatibility)
                                    ;;test  word [es:di+5],8000h
                                    ;test   word [ES:DI+SF_ENTRY.sf_flags],sf_isnet ; network SFT?
00001F7E 7531                       jnz     short lru5     ;yes, get net/Share LRU
```

```
                                ;IF installed
00001F80 E8B764                     call    CheckShare      ;Share present?
                                ;ENDIF
00001F83 752C                       jnz     short lru5      ;yes, get net/Share LRU

                                    ;Local SFT, register its address

                                    ; !!HACK!!!
                                    ; There is a slightly dirty hack out here in a desperate bid to save
                                    ; code space. There is similar code duplicated at label 'gotSFT'. We
                                    ; enter from there if al = 0, update the LocalSFT variable, and since
                                    ; al = 0, we jump out of the loop to the exit point. I have commented
                                    ; out the code that previously existed at label 'gotSFT'

                                hackpoint:
00001F85 893E[760F]                 mov     [LocalSFT],di
00001F89 8C06[780F]                 mov     [LocalSFT+2],es       ;store local SFT address

00001F8D 08C0                       or      al,al           ;Is operation = REGEN?
00001F8F 74BC                       jz      short gotlocalSFT ;yes, return this SFT for reuse

                                    ;Get LRU for local files

                                    ;cmp    [es:di+15h],bx
00001F91 26395D15                   cmp     [es:di+sf_LRU],bx ;SFT.LRU < min?
00001F95 7306                       jae     short lru4      ;no, skip

                                    ;mov    bx,[es:di+15h]
00001F97 268B5D15                   mov     bx,[es:di+sf_LRU] ;yes, store new minimum
00001F9B 89FE                       mov     si,di           ;store SFT position
                                lru4:
                                    ;add    di,59
00001F9D 83C73B                     add     di,SF_ENTRY.size ;go to next SFT
00001FA0 E2CB                       loop    findSFT

                                    ; 20/01/2024
00001FA2 49                         dec     cx ; -1

                                    ; Check whether we got a net/Share or local SFT. If local SFT
                                    ; available, we will reuse it instead of net/Share LRU

00001FA3 89F7                       mov     di,si
                                    ;cmp    si,-1           ;local SFT available?
00001FA5 39CE                       cmp     si,cx ; 20/01/2024
00001FA7 7516                       jnz     short gotSFT    ;yes, return it

                                    ;No local SFT, see if we got a net/Share SFT

00001FA9 89EF                       mov     di,bp

00001FAB 39CD                       cmp     bp,cx ; -1 ; 20/01/2024
                                    ;cmp    bp,-1           ;net/Share SFT available?
00001FAD 752D                       jnz     short gotnetSFT       ;yes, return it
                                noSFT:
                                    ; NB: This error should never occur. We always must have an LRU SFT.
                                    ; This error can occur only if the SFT has been corrupted or the LRU
                                    ; count is not maintained properly.

00001FAF EB4E                       jmp     short errorbadSFT ;error, no FCB available.

                                    ; Handle the LRU for net/Share SFTs
                                lru5:
                                    ;cmp    [es:di+15h],dx
00001FB1 26395515                   cmp     [es:di+sf_LRU],dx ;SFT.LRU < min?
00001FB5 73E6                       jae     short lru4      ;no, skip

                                    ;mov    dx,[es:di+15h]
00001FB7 268B5515                   mov     dx,[es:di+sf_LRU] ;yes, store new minimum

00001FBB 89FD                       mov     bp,di           ;store SFT position
00001FBD EBDE                       jmp     short lru4      ;continue with next SFT

                                gotSFT:
00001FBF 08C0                       or      al,al
00001FC1 74C2                       jz      short hackpoint       ;save es:di in LocalSFT

                                    ; HACK!!!
                                    ; The code here differs from the code at 'hackpoint' only in the
                                    ; order of the check for al. If al = 0, we can jump to 'hackpoint'
                                    ; and then from there jump out to 'gotlocalSFT'. The original code
                                    ; has been commented out below and replaced by the code just above.

                                ;If regen, then this SFT can be registered as a local one ( even if free ).
                                ;
                                ;   or      al,al           ;Regen?
                                ;   jnz     short notlocaluse ;yes, register it and return
                                ;
                                ;Register this SFT as a local one
                                ;
                                ;   mov     [LocalSFT],di
                                ;   mov     [LocalSFT+2],es
                                ;   jmp     gotlocalSFT    ;return to caller
                                ;
                                ;notlocaluse:

                                    ; The caller is probably going to use this SFT for a net/Share file.
                                    ; We will come here only on a Open/Create when the caller($FCB_OPEN)
                                    ; does not really know whether it is a local file or not. We
                                    ; invalidate LocalSFT if the SFT we are going to use was previously
                                    ; registered as a local SFT that can be recycled.

00001FC3 8CC0                       mov     ax,es
00001FC5 393E[760F]                 cmp     [LocalSFT],di         ;Offset same?
00001FC9 750E                       jne     short notinvalid
00001FCB 3906[780F]                 cmp     [LocalSFT+2],ax               ;Segments same?
                                    ;je     short zerolocalSFT    ;no, no need to invalidate
                                    ; 20/01/2024 (PCDOS 7.1 IBMDOS.COM)
00001FCF 7508                       jne     short notinvalid
                                zerolocalSFT:
00001FD1 31C0                       xor     ax,ax ; 0
00001FD3 A3[760F]                   mov     [LocalSFT],ax
00001FD6 A3[780F]                   mov     [LocalSFT+2],ax

                                notinvalid:
00001FD9 E971FF                     jmp     gotlocalSFT

                                    ; The SFT we are going to use was registered in the LocalSFT variable.
                                    ; Invalidate this variable i.e LocalSFT = NULL

                                ;zerolocalSFT:
                                    ;xor    ax,ax ; 0
                                    ;mov    [LocalSFT],ax
                                    ;mov    [LocalSFT+2],ax
                                    ;
```

```
                                    ;jmp    gotlocalSFT
                                
                                gotnetSFT:
                                    ; We have an SFT that is currently net/Share. If it is going to be
                                    ; used for a regen, we know it has to be a local SFT. Update the
                                    ; LocalSFT variable
                                
00001FDC 08C0                       or      al,al
00001FDE 7508                       jnz     short closenet
                                
00001FE0 893E[760F]                 mov     [LocalSFT],di
00001FE4 8C06[780F]                 mov     [LocalSFT+2],es       ;store local SFT address
                                closenet:
00001FE8 893E[9E05]                 mov     [THISSFT],di   ; set thissft
00001FEC 8C06[A005]                 mov     [THISSFT+2],es
                                
                                    ; If we have sharing or thisSFT is a net sft, then close it until ref
                                    ; count is 0.
                                    ; NB: We come here only if it is a net/Share SFT that is going to be
                                    ; recycled -- no need to check for this.
                                
                                LRUClose:
00001FF0 26833D00                   cmp     word [es:di],0
                                    ;cmp    word [es:di+SF_ENTRY.sf_ref_count],0 ; is ref count still <> 0?
00001FF4 740C                       jz      short LRUDone  ; nope, all done
                                
00001FF6 E82F17                     call    DOS_CLOSE
00001FF9 73F5                       jnc     short LRUClose ; no error => clean up
                                
                                    ; Bugbug: I dont know why we are trying to close after we get an
                                    ; error closing. Seems like we could have a potential infinite loop
                                    ; here. This has to be verified.
                                
00001FFB 3C06                       cmp     al,error_invalid_handle ; 6
00001FFD 74F1                       je      short LRUClose
                                errorbadSFT:
00001FFF F9                         stc
00002000 EB05                       JMP     short LRUDead
                                LRUDone:
00002002 30C0                       XOR     AL,AL
00002004 E81101                     call    BlastSFT              ; fill SFT with 0 (AL), 'C' cleared
                                
                                LRUDead:
00002007 E837E4                     call    restore_world         ; use macro
                                
0000200A 07                         pop     es ; * (MSDOS 6.21)
                                
                                    ;getdseg <es>
0000200B 2E8E06[0700]               mov     es,[cs:DosDSeg]
00002010 26C43E[9E05]               les     di,[es:THISSFT]              ;es:di points at allocated SFT
                                
                                    ;;retnc
                                    ;jc     short LruFCB_err
                                    ;retn
                                
                                    ; 16/12/2022
                                    ; 08/11/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
00002015 7302                       jnc     short LruFCB_retn
                                    ;jc     short LruFCB_err
                                    ;retn
                                
                                LruFCB_err:
00002017 B023                       MOV     AL,error_FCB_unavailable ; 23h
                                LruFCB_retn:
00002019 C3                         retn
                                
                                ;LruFCB     ENDP
                                
                                ; 17/05/2019 - Retro DOS v4.0
                                
                                ; DOSCODE:58F3h (MSDOS 6.21, MSDOS.SYS)
                                
                                ; 26/06/2024
                                %if 0
                                
                                ; ---------------------------------------------------------------------------
                                ;**** RegenCopyName -- This function copies the filename from the FCB to
                                ; SFT and also to DOS local buffers. There was duplicate code in FCBRegen
                                ; to copy the name to different destinations
                                ;
                                ; Inputs: ds:si = source string
                                ;    es:di = destination string
                                ;    cx = length of string
                                ;
                                ; Outputs: String copied to destination
                                ;
                                ; Registers affected: cx,di,si
                                ; ---------------------------------------------------------------------------
                                
                                RegenCopyName:
                                CopyName:
                                    lodsb                 ;load character
                                    call    UCase         ; convert char to upper case
                                StuffChar2:
                                    STOSB                 ;store converted character
                                    LOOP    CopyName      ;
                                DoneName:
                                    retn
                                
                                %endif
                                
                                ; ---------------------------------------------------------------------------
                                
                                    ; 09/11/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
                                    ; 21/01/2024 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMDOS.COM)
                                
                                FCBRegen:
                                    ; called from SFTFromFCB. SS already DOSDATA
                                
                                    ; General data filling. Mode is sf_isFCB + open_for_both, date/time
                                    ; we do not fill, size we do no fill, position we do not fill,
                                    ; bit 14 of flags = TRUE, other bits = FALSE
                                
                                    ;mov    al,[si+19h]
0000201A 8A4419                     MOV     AL,[SI+fcb_l_drive]
                                
                                    ; We discriminate based on the first two bits in the reserved field.
                                
                                    ;test   al,80h
0000201D A880                       test    AL,FCBSPECIAL         ; check for no sharing test
0000201F 741C                       JZ      short RegenNoSharing  ; yes, go regen from no sharing
                                
                                    ; The FCB is for a network or a sharing based system. At this point
                                    ; we have already closed the SFT for this guy and reconnection is
```

```
                                    ; impossible.
                                    ;
                                    ; Remember that he may have given us a FCB with bogus information in
                                    ; it. Check to see if sharing is present or if the redir is present.
                                    ; If either is around, presume that we have cycled out the FCB and
                                    ; give the hard error. Otherwise, just return with carry set.

00002021 E81664                     call    CheckShare              ; test for sharer
00002024 7509                       JNZ     short RegenFail                 ; yep, fail this.

                                    ;mov    ax,1100h
00002026 B80011                     MOV     AX,MultNET<<8           ; install check on multnet
00002029 CD2F                       int      2Fh    ; Multiplex - NETWORK REDIRECTOR - INSTALLATION CHECK
                                                    ; Return: AL = 00h  not installed, OK to install
                                                    ; 01h  not installed, not OK to install
                                                    ; FFh  installed
0000202B 08C0                       OR      AL,AL                   ; is it there?
0000202D 740C                       JZ      short RegenDead                 ; no, just fail the operation
                                RegenFail:
                                    ; 17/05/2019 - Retro DOS v4.0
                                    ;MOV    AX,[CS:USER_IN_AX]      ; SS override
0000202F 36A1[3A03]                 mov     ax,[SS:USER_IN_AX] ; MSDOS 6.0

                                    ;cmp    ah,10h
00002033 80FC10                     cmp     AH,FCB_CLOSE
00002036 7403                       jz      short RegenDead
00002038 E89801                     call    FCBHardErr              ; massive hard error.
                                RegenDead:
0000203B F9                         STC                             ; carry set
                                FCBRegen_retn:
0000203C C3                         retn

                                    ; Local FCB without sharing. Check to see if sharing is loaded. If
                                    ; so fail the operation.

                                RegenNoSharing:
0000203D E8FA63                     call    CheckShare              ; Sharing around?
00002040 75ED                       JNZ     short RegenFail

                                    ; Find an SFT for this guy.

                                    ; 17/05/2019 - Retro DOS v4.0

                                    ; MSDOS 3.3
                                    ;call   LRUFCB
                                    ;jc     short FCBRegen_retn

                                    ; MSDOS 6.0
00002042 50                         push    ax
00002043 B000                       mov     al,0                    ;indicate it is a regen operation
00002045 E8EEFE                     call    LRUFCB
00002048 58                         pop     ax
00002049 72F1                       jc      short FCBRegen_retn

                                    ;mov    word [es:di+2],8002h
0000204B 26C745020280               MOV     word [ES:DI+SF_ENTRY.sf_mode],sf_isFCB+open_for_both+SHARING_COMPAT
00002051 243F                       AND     AL,3Fh                  ; get drive number for flags
00002053 98                         CBW
                                    ;or     ax,4000h
00002054 0D0040                     OR      AX,sf_close_nodate      ; normal FCB operation

                                    ; The bits field consists of the upper two bits (dirty and device)
                                    ; from the SFT and the low 4 bits from the open mode.

                                    ;mov    cl,[si+1Ah]
00002057 8A4C1A                     MOV     CL,[SI+fcb_nsl_bits]    ; stick in dirty bits.
0000205A 88CD                       MOV     CH,CL
0000205C 80E5C0                     AND     CH,0C0h                 ; mask off the dirty/device bits
0000205F 08E8                       OR      AL,CH
                                    ;and    cl,0Fh
00002061 80E10F                     AND     CL,access_mask          ; get the mode bits
                                    ;mov    [es:di+2],cl
00002064 26884D02                   MOV     [ES:DI+SF_ENTRY.sf_mode],CL
                                    ;mov    [es:di+5],ax
00002068 26894505                   MOV     [ES:DI+SF_ENTRY.sf_flags],AX ; initial flags
                                    ;MOV    AX,[CS:PROC_ID]                 ; SS override
0000206C 36A1[3C03]                 mov     ax,[ss:PROC_ID] ; MSDOS 6.0
                                    ;mov    [es:di+31h],ax
00002070 26894531                   MOV     [ES:DI+SF_ENTRY.sf_PID],AX
00002074 1E                         push    ds
00002075 56                         push    si
00002076 06                         push    es
00002077 57                         push    di
00002078 16                         push    ss
00002079 07                         pop     es
0000207A BF[4B05]                   MOV     DI,NAME1                ; NAME1 is in DOSDATA

0000207D B90800                     MOV     CX,8
00002080 46                         INC     SI                      ; Skip past drive byte to name in FCB

                                    ; MSDOS 3.3
                                ;RegenCopyName:
                                    ;lodsb
                                    ;call   UCase
                                    ;stosb
                                    ;loop   RegenCopyName

                                    ; MSDOS 6.0
00002081 E88C00                     call    RegenCopyName           ;copy the name to NAME1

00002084 16                         push    ss      ; SS is DOSDATA
00002085 1F                         pop     ds

                                    ;mov    byte [ATTRIB],16h
00002086 C606[6B05]16               MOV     byte [ATTRIB],attr_hidden+attr_system+attr_directory
                                                                    ; Must set this to something interesting
                                                                    ; to call DEVNAME.
0000208B E8462D                     call    DEVNAME                 ; check for device
0000208E 5E                         pop     si
0000208F 07                         pop     es
00002090 5E                         pop     si
00002091 1F                         pop     ds
00002092 7219                       JC      short RegenFileNoSharing ; not found on device list => file

                                    ; Device found. We can ignore disk-specific info

                                    ;mov    [es:di+5],bh
00002094 26887D05                   MOV     [ES:DI+SF_ENTRY.sf_flags],BH ; device parms
                                    ;mov    byte [es:di+4],0
00002098 26C6450400                 MOV     byte [ES:DI+SF_ENTRY.sf_attr],0 ; attribute
                                                                    ; SS override
                                    ;LDS    SI,[CS:DEVPT]           ; get device driver
0000209D 36C536[9A05]               lds     si,[ss:DEVPT] ; MSDOS 6.0
                                regen_save_dpb:     ; 26/06/2024
```

```
15114                                   ;mov    [es:di+7],si
15115 000020A2 26897507                 MOV     [ES:DI+SF_ENTRY.sf_devptr],SI
15116                                   ;mov    [es:di+9],ds
15117 000020A6 268C5D09                 MOV     [ES:DI+SF_ENTRY.sf_devptr+2],DS
15118 000020AA C3                       retn                            ; carry is clear
15119
15120                                 RegenDeadJ:
15121 000020AB EB8E                     JMP     short RegenDead
15122
15123                                   ; File found. Just copy in the remaining pieces.
15124
15125                                 RegenFileNoSharing:
15126                                   ;mov    ax,[es:di+5]
15127 000020AD 268B4505                 MOV     AX,[ES:DI+SF_ENTRY.sf_flags]
15128 000020B1 83E03F                   AND     AX,03Fh
15129 000020B4 1E                       push    ds
15130 000020B5 56                       push    si
15131 000020B6 E8075A                   call    FIND_DPB
15132                                   ;;mov   [es:di+7],si
15133                                   ;MOV    [ES:DI+SF_ENTRY.sf_devptr],SI
15134                                   ;;mov   [es:di+9],ds
15135                                   ;MOV    [ES:DI+SF_ENTRY.sf_devptr+2],DS
15136                                   ; 26/06/2024 (PCDOS 7.1 IBMDOS.COM)
15137 000020B9 E8E6FF                   call    regen_save_dpb
15138 000020BC 5E                       pop     si
15139 000020BD 1F                       pop     ds
15140 000020BE 72EB                     jc      short RegenDeadJ        ; if find DPB fails, then drive
15141                                                                   ; indicator was bogus
15142                                   ;mov    ax,[si+1Dh]
15143 000020C0 8B441D                   MOV     AX,[SI+fcb_nsl_dirsec]
15144                                   ;;mov   [es:di+1Dh],ax ; MSDOS 3.3
15145                                   ;mov    [es:di+1Bh],ax ; MSDOS 6.0
15146 000020C3 2689451B                 MOV     [ES:DI+SF_ENTRY.sf_dirsec],AX
15147
15148                                   ; MSDOS 6.0
15149
15150                                   ; SR;
15151                                   ; Extract the read-only and archive bits from the top 2 bits of the sector
15152                                   ; number
15153
15154                                   ;mov    al,[si+18h]
15155 000020C7 8A4418                   mov     al,[si+fcb_sfn]
15156 000020CA 24C0                     and     al,0C0h         ;get the 2 attribute bits
15157 000020CC 88C4                     mov     ah,al
15158 000020CE D0C4                     rol     ah,1
15159 000020D0 D0E8                     shr     al,1
15160 000020D2 08E0                     or      al,ah
15161 000020D4 243F                     and     al,03Fh         ;mask off unused bits
15162                                   ;mov    [es:di+4],al
15163 000020D6 26884504                 mov     [es:di+SF_ENTRY.sf_attr],al
15164
15165                                   ; SR;
15166                                   ; Update the higher word of the directory sector from the FCB
15167
15168                                   ;;mov   al,[si+18h]
15169 000020DA 8A4418                   mov     al,[si+fcb_sfn]
15170 000020DD 243F                     and     al,03Fh         ;mask off top 2 bits -- attr bits
15171 000020DF 28E4                     sub     ah,ah
15172                                   ;mov    [es:di+1Dh],ax
15173 000020E1 2689451D                 mov     [es:di+SF_ENTRY.sf_dirsec+2],ax ;update high word
15174
15175                                   ; 21/01/2024
15176                                   ; MSDOS 6.0 (& MSDOS 3.3)
15177                                   ;mov    ax,[si+1Bh]
15178 000020E5 8B441B                   MOV     AX,[SI+fcb_nsl_firclus]
15179                                   ;;mov   [es:di+0Bh],ax
15180                                   ;MOV    [ES:DI+SF_ENTRY.sf_firclus],AX
15181                                   ;;;
15182                                   ; 21/01/2024 (PCDOS 7.1 IBMDOS.COM)
15183 000020E8 2689452B                 mov     [es:di+2Bh],ax ; .sf_chain !!! (MSDOS 6.22)
15184                                   ;mov    [es:di+SF_ENTRY.sf_chain],ax ; first cluster (32 bit) !?
15185                                   ;;;
15186
15187                                   ;;mov   [es:di+1Bh],ax ; MSDOS 3.3
15188                                   ;mov    [es:di+35h],ax ; MSDOS 6.0
15189 000020EC 26894535                 MOV     [ES:DI+SF_ENTRY.sf_lstclus],AX
15190
15191                                   ;;;
15192                                   ; 21/01/2024 (PCDOS 7.1 IBMDOS.COM)
15193 000020F0 31C0                     xor     ax,ax   ; 0
15194 000020F2 2689452D                 mov     [es:di+2Dh], ax ; 0
15195                                   ;mov    [es:di+SF_ENTRY.sf_chain+2],ax
15196                                                   ; .sf_chain ! (MSDOS 6.22)
15197                                                   ; high word of first cluster (32 bit) !?
15198 000020F6 26894537                 mov     [es:di+37h],ax ; 0
15199                                   ;mov    [es:di+SF_ENTRY.sf_lstclus+2],ax ; hw of last cluster
15200                                   ;;;
15201
15202                                   ;mov    al,[si+1Fh]
15203 000020FA 8A441F                   MOV     AL,[SI+fcb_ns]_dirpos]
15204                                   ;mov    [es:di+1Fh],al
15205 000020FD 2688451F                 MOV     [ES:DI+SF_ENTRY.sf_dirpos],AL
15206                                   ;INC    word [ES:DI+SF_ENTRY.sf_ref_count]
15207 00002101 26FF05                   inc     word [ES:DI]            ; Increment reference count.
15208                                                                   ; Existing FCB entries would be
15209                                                                   ; flushed unnecessarily because of
15210                                                                   ; check in CheckFCB of the ref_count.
15211                                                                   ; July 22/85 - BAS
15212                                   ;;;
15213                                   ; 21/01/2024 (PCDOS 7.1 IBMDOS.COM)
15214 00002104 E81129                   call    set_sftfcb_entry ; put SFT entry number in the SFTFCB table
15215                                                            ; as FCB index number
15216                                   ;;;
15217
15218                                   ;lea    si,[si+1]
15219 00002107 8D7401                   LEA     SI,[SI+SYS_FCB.name]
15220                                   ;lea    di,[di+20h]
15221 0000210A 8D7D20                   LEA     DI,[DI+SF_ENTRY.sf_name]
15222                                   ;mov    cx,11
15223 0000210D B90B00                   MOV     CX,SYS_FCB.EXTENT-SYS_FCB.name ; 12-1
15224
15225                                   ; 26/06/2024
15226                                   ; MSDOS 6.0
15227                                   ;call   RegenCopyName  ;copy name to SFT
15228                                   ; 26/06/2024
15229                                   ; cf = 0 (at the result of the 'test' instruction)
15230
15231                                   ; MSDOS 3.3
15232                                 ;RegenCopyName2:
15233                                   ;lodsb
15234                                   ;call    UCase
15235                                   ;stosb
15236                                   ;loop    RegenCopyName2
15237
```

```
                                    ; 26/06/2024
                                    ; cf = 0
                                    ;clc
                                    ;retn

                                ; 26/06/2024
                                %if 1

                                ; ----------------------------------------------------------------------------
                                ;**** RegenCopyName -- This function copies the filename from the FCB to
                                ; SFT and also to DOS local buffers. There was duplicate code in FCBRegen
                                ; to copy the name to different destinations
                                ;
                                ; Inputs: ds:si = source string
                                ;    es:di = destination string
                                ;    cx = length of string
                                ;
                                ; Outputs: String copied to destination
                                ;
                                ; Registers affected: cx,di,si
                                ; ----------------------------------------------------------------------------

                                RegenCopyName:
                                CopyName:
15262 00002110 AC                   lodsb                   ;load character
15263 00002111 E8833C               call   UCase ; *        ; convert char to upper case
15264                           StuffChar2:
15265 00002114 AA                   STOSB                   ;store converted character
15266 00002115 E2F9                 LOOP   CopyName         ;
15267                               ; 26/06/2024
15268                               ; cf= 0 ; *
15269                           DoneName:
15270 00002117 C3                   retn

15272                           %endif

15274                           ; 17/05/2019 - Retro DOS v4.0
15275                           ; 21/01/2024 - Retro DOS v5.0
15276
15277                           ;** BlastSFT - FIll SFT with Garbage
15278                           ; ----------------------------------------------------------------------------
15279                           ;   BlastSFT is used when an SFT is no longer needed; it's called with
15280                           ;   various garbage values to put into the SFT.  I don't know why,
15281                           ;   presumably to help with debugging (jgl).  We clear the few fields
15282                           ;   necessary to show that the SFT is free after filling it.
15283                           ;
15284                           ;   ENTRY   (es:di) = address of SFT
15285                           ;           (al) = fill character
15286                           ;   EXIT    (ax) = -1
15287                           ;           'C' clear
15288                           ;   USES    AX, CX, Flags
15289
15290                           BlastSFT:
15291                               ; 21/01/2024 (PCDOS 7.1 IBMDOS.COM)
15292                               ;;;
15293 00002118 E8D029             call    SFT_FREE
15294                               ;;;
15295 0000211B 57                   push    di
15296                               ;mov    cx,53 ; MSDOS 3.3
15297                               ;mov    cx,59 ; MSDOS 6.0
15298 0000211C B93B00               mov     cx,SF_ENTRY.size
15299 0000211F F3AA                 rep     stosb
15300 00002121 5F                   pop     di
15301 00002122 29C0                 sub     ax,ax   ; 0             ; clear 'C'-----------------;
15302 00002124 268905               mov     [es:di],ax
15303                               ;mov    [es:di+SF_ENTRY.sf_ref_count],ax ; set ref count    ;
15304                               ;mov    [es:di+15h],ax
15305 00002127 26894515             mov     [es:di+sf_LRU],ax       ; set lru                   ;
15306 0000212B 48                   dec     ax      ; -1                                        ;
15307                               ;mov    [es:di+17h],ax ; 0FFFFh ; -1
15308 0000212C 26894517             mov     [es:di+sf_OpenAge],ax ; set open age to -1      ;
15309                           BlastSFT_retn:
15310 00002130 C3                   retn                            ; return with 'C' clear     ;
15311
15312                           ;Break      <CheckFCB - see if the SFT pointed to by the FCB is still OK>
15313                           ; ----------------------------------------------------------------------------
15314                           ;
15315                           ;   CheckFCB - examine an FCB and its contents to see if it needs to be
15316                           ;   regenerated.
15317                           ;
15318                           ;   Inputs: DS:SI point to FCB (not extended)
15319                           ;           AL is SFT index
15320                           ;   Outputs:        Carry Set - FCB needs to be regened
15321                           ;           Carry clear - FCB is OK. ES:DI point to SFT
15322                           ;   Registers modified: AX and BX
15323                           ;
15324                           ; ----------------------------------------------------------------------------
15325
15326                               ; 09/11/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
15327                               ; DOSCODE:59F0h (MSDOS 5.0, MSDOS.SYS)
15328
15329                               ; 21/01/2024 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMDOS.COM)
15330                               ; DOSCODE:607Eh (PCDOS 7.1 IBMDOS.COM)
15331                           CheckFCB:
15332
15333                               ; called from $fcb_open and sftfromfcb. SS already set up to DOSDATA
15334
15335                               ; MSDOS 3.3
15336
15337                               ; LES   DI,[CS:SFTFCB]
15338
15339                               ; MSDOS 6.0
15340
15341                               ; SR;
15342                               ; We check if the given FCB is for a local file. If so, we return a
15343                               ; bad SFT status forcing the caller to regenerate the SFT.
15344
15345                               ;test   byte [si+19h],0C0h
15346 00002131 F64419C0             test    byte [si+fcb_l_drive],FCBNETWORK|FCBSHARE|FCBDEVICE
15347 00002135 7464                 jz      short BadSFT            ;Local file, return bad SFT
15348 00002137 36C43E[4000]         LES     DI,[SS:SFTFCB]          ; SS override
15349
15350                               ; MSDOS 6.0 (& MSDOS 3.3)
15351                               ;cmp    [es:di+4],al
15352 0000213C 26384504             CMP     [ES:DI+SFT.SFCount],AL
15353 00002140 7259                 JC      short BadSFT
15354                               ;;mov   bl,53 ; MSDOS 3.3
15355                               ;mov    bl,59 ; MSDOS 6.0
15356 00002142 B33B                 MOV     BL,SF_ENTRY.size
15357 00002144 F6E3                 MUL     BL
15358                               ;lea    di,[di+6]
15359 00002146 8D7D06               LEA     DI,[DI+SFT.SFTable]
15360 00002149 01C7                 ADD     DI,AX
15361                               ;MOV    AX,[CS:PROC_ID]         ; MSDOS 3.3
```

```
15362 0000214B 36A1[3C03]                mov     ax,[SS:PROC_ID] ; MSDOS 6.0  ; SS override
15363                                    ;cmp    [es:di+31h],ax
15364 0000214F 26394531                  CMP     [ES:DI+SF_ENTRY.sf_PID],AX
15365 00002153 7546                      JNZ     short BadSFT            ; must match process
15366 00002155 26833D00                  cmp     word [es:di],0
15367                                    ;CMP    word [ES:DI+SF_ENTRY.sf_ref_count],0
15368 00002159 7440                      JZ      short BadSFT            ; must also be in use
15369                                    ;mov    al,[si+19h]
15370 0000215B 8A4419                    MOV     AL,[SI+fcb_1_drive]
15371                                    ;test   al,80h
15372 0000215E A880                      test    AL,FCBSPECIAL           ; a special FCB?
15373 00002160 7427                      JZ      short CheckNoShare      ; No. try local or device
15374
15375                                    ; Since we are a special FCB, try NOT to use a bogus test instruction.
15376                                    ; FCBSHARE is a superset of FCBNETWORK.
15377
15378 00002162 50                         PUSH    AX
15379                                    ;and    al,0C0h
15380 00002163 24C0                      AND     AL,FCBMASK
15381                                    ;cmp    al,0C0h
15382 00002165 3CC0                      CMP     AL,FCBSHARE             ; net FCB?
15383 00002167 58                        POP     AX
15384 00002168 7515                      JNZ     short CheckNet          ; no
15385                                                                    ; yes
15386                                ;
15387                                ;----- In share support -----
15388                                ;
15389                                    ;call   far [cs:JShare+(11*4)]
15390 0000216A 36FF1E[BC00]              Call    far [ss:JShare+(11*4)] ; 11 = ShChk ; SS Override
15391 0000216F 722A                      JC      short BadSFT
15392
15393                                ; 21/01/2024
15394                                %if 0
15395                                    JMP     SHORT CheckD
15396                                ;
15397                                ;----- End in share support -----
15398                                ;
15399                                    ; 09/11/2022
15400                                    ; (There is not any procedure/sub
15401                                    ;  which calls or jumps to CheckFirClus here)
15402                                    ;;;
15403                                CheckFirClus:
15404                                    ;cmp     bx,[es:di+0Bh]
15405                                    ; 07/12/2022
15406                                    CMP     BX,[ES:DI+SF_ENTRY.sf_firclus]
15407                                    JNZ     short BadSFT
15408                                    ;;;
15409                                %endif
15410
15411                                CheckD:
15412 00002171 243F                      AND     AL,3Fh
15413                                    ;mov    ah,[es:di+5]
15414 00002173 268A6505                  MOV     AH,[ES:DI+SF_ENTRY.sf_flags]
15415 00002177 80E43F                    AND     AH,3Fh
15416 0000217A 38C4                      CMP     AH,AL
15417                                    ; 26/06/2024
15418                                    ; 16/12/2022
15419                                    ;jz     short BlastSFT_retn   ; carry is clear
15420                                    ; 09/11/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
15421 0000217C 751D                      jnz     short BadSFT
15422                                CheckD_retn:
15423 0000217E C3                        retn
15424
15425                                ; 26/06/2024
15426                                ;BadSFT:
15427                                ;   STC
15428                                ;   retn
15429
15430                                CheckNet:
15431                                    ; 17/05/2019 - Retro DOS v4.0
15432
15433                                ;----- In net support -----
15434
15435                                    ; MSDOS 3.3
15436                                    ;;mov   ax,[si+1Ah]
15437                                    ;mov    ax,[si+fcb_net_handle]
15438                                    ;;cmp   ax,[es:di+1Dh]
15439                                    ;cmp    ax,[ES:DI+SF_ENTRY.sf_dirsec]
15440                                    ;jnz    short BadSFT
15441                                    ;;cmp   ax,[es:di+19h]
15442                                    ;cmp    ax,[ES:DI+sf_netid]
15443                                    ;jnz     short BadSFT
15444                                    ;;mov   ax,[si+1Eh]
15445                                    ;mov    ax,[si+fcb_l_attr]
15446                                    ;;cmp   ax,[es:di+1Bh]
15447                                    ;cmp    ax,[es:di+SF_ENTRY.sf_lstclus]
15448                                    ;jnz     short BadSFT
15449
15450                                    ; MSDOS 6.0
15451                                    ;mov    ax,[si+1Ch]
15452 0000217F 8B441C                    MOV     AX,[SI+fcb_netID]     ;AN000;IFS.DOS 4.00
15453                                    ; 09/11/2022
15454                                    ;cmp    ax,[es:di+0Bh]
15455 00002182 263B450B                  CMP     AX,[ES:DI+sf_serial_ID]       ;AN000;IFS.DOS 4.00
15456 00002186 7513                      JNZ     short BadSFT
15457
15458                                ;----- END In net support -----
15459
15460                                CheckNet_retn:
15461 00002188 C3                        retn
15462
15463                                CheckNoShare:
15464
15465                                ; 16/12/2022
15466                                ;   ; 09/11/2022 (following test instruction is nonsense!)
15467                                ;   ; (I am leaving it here for MSDOS 5.0 MSDOS.SYS compatibility)
15468                                ;   ;test   al,40h
15469                                ;   test    AL,FCBDEVICE            ; Device?
15470                                ;   ;jnz    short $+2 ; 09/11/2022
15471                                ;   JNZ     short CheckNoShareDev ; Yes
15472
15473                                    ; MSDOS 3.3 - IBMDOS.COM - Offset 27EFh
15474                                    ;;mov   bx,[si+1Dh]
15475                                    ;MOV    BX,[SI+fcb_nsl_dirsec]
15476                                    ;;cmp   bx,[es:di+1Dh]
15477                                    ;cmp    bx,[ES:DI+SF_ENTRY.sf_dirsec]
15478                                    ;jnz    short BadSFT
15479                                    ;;mov   bl,[si+1Fh]
15480                                    ;MOV    bl,[SI+fcb_nsl_dirpos]
15481                                    ;;cmp   bl,[es:di+1Fh]
15482                                    ;cmp    bl,[ES:DI+SF_ENTRY.sf_dirpos]
15483                                    ;jnz    short BadSFT
15484                                    ;;mov   bl,[si+1Ah]
15485                                    ;MOV    bl,[SI+fcb_nsl_bits]
```

```
                                    ;;mov   bh,[es:di+5]
                                    ;MOV    bh,[ES:DI+SF_ENTRY.sf_flags]
                                    ;xor    bh,bl
                                    ;and    bh,0C0h
                                    ;jnz    short BadSFT
                                    ;;xor   bl,[es:di+2]
                                    ;xor    bl,[ES:DI+SF_ENTRY.sf_mode]
                                    ;and    bl,0Fh
                                    ;jnz    short BadSFT
                                    ;push   di
                                    ;push   si
                                    ;;lea   di,[di+20h]  ; MSDOS 3.3
                                    ;LEA    DI,[DI+SF_ENTRY.sf_name]
                                    ;;lea   si,[si+1]
                                    ;LEA    SI,[SI+SYS_FCB.name]
                                    ;;mov   cx,11
                                    ;MOV    CX,SYS_FCB.EXTENT-SYS_FCB.name ; 12-1
                                    ;repe   cmpsb
                                    ;pop    si
                                    ;pop    di
                                    ;jnz    short BadSFT
                                    ;;mov   bx,[si+1Bh]
                                    ;MOV    bX,[SI+fcb_nsl_firclus]
                                    ;jmp    short CheckFirClus

                                    ; MSDOS 6.0

                                    ; SR;
                                    ; The code below to match a local FCB with its SFT can no longer be
                                    ; used. We just return a no-match status. This check is done right
                                    ; at the top.

                                CheckNoShareDev:
                                    ;mov    bx,[si+1Ah]
00002189 8B5C1A                     MOV     BX,[SI+fcb_nsld_drvptr]
                                    ;cmp    bx,[es:di+7]
0000218C 263B5D07                   CMP     BX,[ES:DI+SF_ENTRY.sf_devptr]
00002190 7509                       JNZ     short BadSFT
                                    ;mov    bx,[si+1Ch]
00002192 8B5C1C                     MOV     BX,[SI+fcb_nsld_drvptr+2]
                                    ;cmp    bx,[es:di+9]
00002195 263B5D09                   CMP     BX,[ES:DI+SF_ENTRY.sf_devptr+2]
                                    ;JNZ    short BadSFT
                                    ;JMP    short CheckD
                                    ; 26/06/2024
00002199 74D6                       jz      short CheckD

                                ; 26/06/2024
                                BadSFT:
0000219B F9                         STC
0000219C C3                         retn

                                ;Break      <SFTFromFCB - take a FCB and obtain a SFT from it>
                                ;-----------------------------------------------------------------------------
                                ;
                                ;   SFTFromFCB - the workhorse of this compatability crap. Check to see if
                                ;   the SFT for the FCB is Good. If so, make ThisSFT point to it. If not
                                ;   good, get one from the cache and regenerate it. Overlay the LRU field
                                ;   with PID
                                ;
                                ;   Inputs: DS:SI point to FCB
                                ;   Outputs:       ThisSFT point to appropriate SFT
                                ;           Carry clear -> OK ES:DI -> SFT
                                ;           Carry set -> error in ax
                                ;   Registers modified: ES,DI, AX
                                ;
                                ;-----------------------------------------------------------------------------

                                SFTFromFCB:
                                    ; called from fcbio and $fcb_close. SS already set up to DOSDATA

                                    ; 17/05/2019 - Retro DOS v4.0

0000219D 50                         push    ax
0000219E 53                         push    bx
                                    ;mov    al,[si+18h]
0000219F 8A4418                     MOV     AL,[SI+fcb_sfn]       ; set SFN for check
000021A2 E88CFF                     call    CheckFCB
000021A5 5B                         pop     bx
000021A6 58                         pop     ax
                                    ;MOV    [CS:THISSFT],DI             ; SS override
                                    ;MOV    [CS:THISSFT+2],ES
000021A7 36893E[9E05]               MOV     [SS:THISSFT],DI             ; SS override
000021AC 368C06[A005]               MOV     [SS:THISSFT+2],ES
000021B1 7311                       JNC     short Set_SFT         ; no problems, just set thissft

                                    ; 09/11/2022 (MSDOS 5.0)
                                    ; 31/05/2019
000021B3 06                         push    es ; * (MSDOS 6.21) & (MSDOS 5.0)
000021B4 E8A1E2                     call    save_world
000021B7 E860FE                     call    FCBRegen
000021BA E884E2                     call    restore_world         ; use macro restore world
000021BD 07                         pop     es ; * (MSDOS 6.21) ; 31/05/2019 ; 09/11/2022 (MSDOS 5.0)

                                    ;MOV    AX,[CS:EXTERR]        ; SS override
000021BE 36A1[2403]                 MOV     AX,[SS:EXTERR]        ; SS override
000021C2 72C4                       jc      short CheckNet_retn

                                Set_SFT:
                                    ;LES    DI,[CS:THISSFT]              ; SS override for THISSFT & PROC_ID
000021C4 36C43E[9E05]               les     di,[ss:THISSFT]
                                    ;PUSH   word [CS:PROC_ID]     ; set process id
000021C9 36FF36[3C03]               push    word [ss:PROC_ID]
                                    ;pop    word [es:di+31h]
000021CE 268F4531                   POP      word [ES:DI+SF_ENTRY.sf_PID]
000021D2 C3                         retn                          ; carry is clear

                                ;Break      <FCBHardErr - generate INT 24 for hard errors on FCBS>
                                ;-----------------------------------------------------------------------------
                                ;
                                ;   FCBHardErr - signal to a user app that he is trying to use an
                                ;   unavailable FCB.
                                ;
                                ;   Inputs: none.
                                ;   Outputs:       none.
                                ;   Registers modified: all
                                ;
                                ;-----------------------------------------------------------------------------

                                    ; 21/01/2024 - Retro DOS v5.0
                                FCBHardErr:
                                    ; 17/05/2019 - Retro DOS v4.0
000021D3 2E8E06[0700]               mov     es,[cs:DosDSeg]
                                    ;
```

```
                                    ;mov    ax,23h
000021D8 B82300                     MOV     AX,error_FCB_unavailable
                                    ;;mov   byte [cs:ALLOWED],8
                                    ;MOV    byte [CS:ALLOWED],Allowed_FAIL
000021DB 26C606[4B03]08             mov     byte [es:ALLOWED],Allowed_FAIL

                                    ;LES    BP,[CS:THISDPB]
000021E1 26C42E[8A05]               les     bp,[es:THISDPB]

000021E6 BF0100                     MOV     DI,1                    ; Fake some registers
000021E9 89F9                       MOV     CX,DI
                                    ;;;
                                    ; 21/01/2024 (PCDOS 7.1 IBMDOS.COM)
000021EB 31D2                       xor     dx,dx ; 0
                                    ;cmp    [es:bp+0Fh],dx
000021ED 2639560F                   cmp     [es:bp+DPB.FAT_SIZE],dx ; 0
000021F1 740B                       jz      short fcbharderr_fat32 ; FAT32
000021F3 268916[0706]               mov     [es:HIGH_SECTOR],dx ; 0
                                    ;mov    dx,[es:bp+0Bh]
000021F8 268B560B                   mov     dx,[es:bp+DPB.FIRST_SECTOR]
000021FC EB0D                       jmp     short fcbharderr_fat
                                fcbharderr_fat32:
                                    ;mov    dx,[es:bp+2Bh]
000021FE 268B562B                   mov     dx,[es:bp+DPB.FCLUS_FSECTOR+2]
00002202 268916[0706]               mov     [es:HIGH_SECTOR],dx
                                    ;mov    dx,[es:bp+29h]
00002207 268B5629                   mov     dx,[es:bp+DPB.FCLUS_FSECTOR]
                                fcbharderr_fat:
                                    ;;;
                                    ; 21/01/2024
                                    ;;mov   dx,[es:bp+0Bh]
                                    ;MOV    DX,[ES:BP+DPB.FIRST_SECTOR]

0000220B E8463E                     call    HARDERR
0000220E F9                         STC
0000220F C3                         retn

                                ;=============================================================================
                                ; FCBIO2.ASM, MSDOS 6.0, 1991
                                ;=============================================================================
                                ; 21/07/2018 - Retro DOS v3.0
                                ; 17/05/2019 - Retro DOS v4.0

                                ;** FCBIO2.ASM - Ancient 1.0 1.1 FCB system calls
                                ;
                                ;   GetRR
                                ;   GetExtent
                                ;   SetExtent
                                ;   GetExtended
                                ;   GetRecSize
                                ;   FCBIO
                                ;   $FCB_OPEN
                                ;   $FCB_CREATE
                                ;   $FCB_RANDOM_WRITE_BLOCK
                                ;   $FCB_RANDOM_READ_BLOCK
                                ;   $FCB_SEQ_READ
                                ;   $FCB_SEQ_WRITE
                                ;   $FCB_RANDOM_READ
                                ;   $FCB_RANDOM_WRITE
                                ;
                                ;   Revision history:
                                ;
                                ;          Created: ARR 4 April 1983
                                ;                   MZ  6 June  1983 completion of functions
                                ;                   MZ 15 Dec   1983 Brain damaged programs close FCBs multiple
                                ;                           times.  Change so successive closes work by
                                ;                           always returning OK.Also, detect I/O to
                                ;                           already closed FCB and return EOF.
                                ;           MZ 16 Jan   1984 More braindamage.  Need to separate info
                                ;                           out of sft into FCB for reconnection
                                ;
                                ;       A000   version 4.00      Jan. 1988

                                ; Defintions for FCBOp flags

                                RANDOM      equ 2                       ; random operation
                                FCBREAD equ 4                           ; doing a read
                                BLOCK       equ 8                       ; doing a block I/O

                                ;Break <GetRR - return the random record field in DX:AX>
                                ;----------------------------------------------------------------------------
                                ;
                                ;   GetRR - correctly load DX:AX with the random record field (3 or 4 bytes)
                                ;   from the FCB pointed to by DS:SI
                                ;
                                ;   Inputs: DS:SI point to an FCB
                                ;           BX has record size
                                ;   Outputs:       DX:AX contain the contents of the random record field
                                ;   Registers modified: none
                                ;----------------------------------------------------------------------------

                                GetRR:
                                    ;mov    ax,[si+21h]
00002210 8B4421                     MOV     AX,[SI+SYS_FCB.RR]    ; get low order part
                                    ;mov    dx,[si+23h]
00002213 8B5423                     MOV     DX,[SI+SYS_FCB.RR+2]  ; get high order part
00002216 83FB40                     CMP     BX,64                 ; ignore MSB of RR if recsiz > 64
00002219 7202                       JB      short GetRRBye
                                GetExtent_bye:     ; 21/01/2024
0000221B 30F6                       XOR     DH,DH
                                GetRRBye:
0000221D C3                         retn

                                ;Break <GetExtent - retrieve next location for sequential IO>
                                ;----------------------------------------------------------------------------
                                ;
                                ;   GetExtent - Construct the next record to perform I/O from the EXTENT and
                                ;   NR fields in the FCB.
                                ;
                                ;   Inputs: DS:SI - point to FCB
                                ;   Outputs:       DX:AX contain the contents of the random record field
                                ;   Registers modified: none
                                ;----------------------------------------------------------------------------

                                GetExtent:
                                    ;mov    al,[si+20h]
0000221E 8A4420                     MOV     AL,[SI+SYS_FCB.NR]    ; get low order piece
                                    ;mov    dx,[si+0Ch]
00002221 8B540C                     MOV     DX,[SI+SYS_FCB.EXTENT]; get high order piece
00002224 D0E0                       SHL     AL,1
00002226 D1EA                       SHR     DX,1
00002228 D0D8                       RCR     AL,1    ; move low order bit of DL to high order of AH
0000222A 88D4                       MOV     AH,DL
0000222C 88F2                       MOV     DL,DH
```

```
                                    ; 21/01/2024 (PCDOS 7.1 IBMDOS.COM)
                                    ;XOR   DH,DH
                                    ;retn
0000222E EBEB                       jmp     short GetExtent_bye

                                ;Break <SetExtent - update the extent/NR field>
                                ;--------------------------------------------------------------------------
                                ;
                                ;   SetExtent - change the position of an FCB by filling in the extent/NR
                                ;   fields
                                ;
                                ;   Inputs: DS:SI point to FCB
                                ;           DX:AX is a record location in file
                                ;   Outputs:       Extent/NR fields are filled in
                                ;   Registers modified: CX
                                ;--------------------------------------------------------------------------

                                SetExtent:
00002230 50                         push    ax
00002231 52                         push    dx
00002232 89C1                       MOV     CX,AX
00002234 247F                       AND     AL,7FH                  ; next rec field
                                    ;mov    [si+20h],al
00002236 884420                     MOV     [SI+SYS_FCB.NR],AL
00002239 80E180                     AND     CL,80H                  ; save upper bit
0000223C D1E1                       SHL     CX,1
0000223E D1D2                       RCL     DX,1                    ; move high bit of CX to low bit of DX
00002240 88E8                       MOV     AL,CH
00002242 88D4                       MOV     AH,DL
                                    ;mov    [si+0Ch], ax
00002244 89440C                     MOV     [SI+SYS_FCB.EXTENT],AX; all done
00002247 5A                         pop     dx
00002248 58                         pop     ax
00002249 C3                         retn

                                ;Break <GetExtended - find FCB in potential extended fcb>
                                ;--------------------------------------------------------------------------
                                ;
                                ;   GetExtended - Make DS:SI point to FCB from DS:DX
                                ;
                                ;   Inputs: DS:DX point to a possible extended FCB
                                ;   Outputs:       DS:SI point to the FCB part
                                ;           zeroflag set if not extended fcb
                                ;   Registers modified: SI
                                ;--------------------------------------------------------------------------

                                GetExtended:
0000224A 89D6                       MOV     SI,DX                   ; point to Something
0000224C 803CFF                     CMP     BYTE [SI],-1            ; look for extention
0000224F 7503                       JNZ     short GetBye            ; not there
00002251 83C607                     ADD     SI,7                    ; point to FCB
                                GetBye:
00002254 39D6                       CMP     SI,DX                   ; set condition codes
                                getextd_retn:
00002256 C3                         retn

                                ;Break <GetRecSize - return in BX the FCB record size>
                                ;--------------------------------------------------------------------------
                                ;
                                ;   GetRecSize - return in BX the record size from the FCB at DS:SI
                                ;
                                ;   Inputs: DS:SI point to a non-extended FCB
                                ;   Outputs:       BX contains the record size
                                ;   Registers modified: None
                                ;--------------------------------------------------------------------------

                                    ; 22/01/2024
                                    ; 07/12/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
                                GetRecSize:
                                    ;mov    bx,[si+0Eh]
00002257 8B5C0E                     MOV     BX,[SI+SYS_FCB.RECSIZ]; get his record size
0000225A 09DB                       OR      BX,BX                   ; is it nul?
                                    ;jz     short getextd_retn
                                    ; 22/01/2024 (BugFix)
0000225C 75F8                       jnz     short getextd_retn
                                    ;MOV    BX,128                  ; use default size
0000225E B380                       mov     bl,128  ; (PCDOS 7.1 IBMDOS.COM)
                                    ;mov    [si+0Eh],bx
00002260 895C0E                     MOV     [SI+SYS_FCB.RECSIZ],BX; stuff it back
00002263 C3                         retn

                                ; 23/01/2024 - Retro DOS v5.0
                                ; PCDOS 7.1 IBMDOS.COM - DOSCODE:61B3h

                                ; 22/07/2018 - Retro DOS v3.0

                                ;BREAK <$FCB_Random_write_Block - write a block of records to a file >
                                ;--------------------------------------------------------------------------
                                ;
                                ;   $FCB_Random_Write_Block - retrieve a location from the FCB, seek to it
                                ;   and write a number of blocks from it.
                                ;
                                ;   Inputs: DS:DX point to an FCB
                                ;   Outputs:       AL = 0 write was successful and the FCB position is updated
                                ;           AL <> 0 Not enough room on disk for the output
                                ;
                                ;--------------------------------------------------------------------------

                                _$FCB_RANDOM_WRITE_BLOCK:
                                    ;mov    AL,0Ah
00002264 B00A                       MOV     AL,RANDOM+BLOCK
00002266 EB12                       JMP     short FCBIO    ; 23/01/2024

                                ;BREAK <$FCB_Random_Read_Block - read a block of records to a file >
                                ;--------------------------------------------------------------------------
                                ;
                                ;   $FCB_Random_Read_Block - retrieve a location from the FCB, seek to it
                                ;   and read a number of blocks from it.
                                ;
                                ;   Inputs: DS:DX point to an FCB
                                ;   Outputs:       AL = error codes defined above
                                ;
                                ;--------------------------------------------------------------------------

                                _$FCB_RANDOM_READ_BLOCK:
                                    ;mov    AL,0Eh
00002268 B00E                       MOV     AL,RANDOM+FCBREAD+BLOCK
0000226A EB0E                       JMP     short FCBIO    ; 23/01/2024

                                ;BREAK <$FCB_Seq_Read - read the next record from a file >
                                ;--------------------------------------------------------------------------
                                ;
                                ;   $FCB_Seq_Read - retrieve the next record from an FCB and read it into
                                ;   memory
```

```
                                ;
                                ;   Inputs: DS:DX point to an FCB
                                ;   Outputs:       AL = error codes defined above
                                ;
                                ;-----------------------------------------------------------------------------

                                _$FCB_SEQ_READ:
                                    ;mov    AL,4
0000226C B004                       MOV     AL,FCBREAD
0000226E EB0A                       JMP     short FCBIO    ; 23/01/2024

                                ;BREAK <$FCB_Seq_Write - write the next record to a file >
                                ;-----------------------------------------------------------------------------
                                ;
                                ;   $FCB_Seq_Write - retrieve the next record from an FCB and write it to the
                                ;   file
                                ;
                                ;   Inputs: DS:DX point to an FCB
                                ;   Outputs:       AL = error codes defined above
                                ;
                                ;-----------------------------------------------------------------------------

                                _$FCB_SEQ_WRITE:
00002270 B000                       MOV     AL,0
00002272 EB06                       JMP     short FCBIO    ; 23/01/2024

                                ;BREAK <$FCB_Random_Read - Read a single record from a file >
                                ;-----------------------------------------------------------------------------
                                ;
                                ;   $FCB_Random_Read - retrieve a location from the FCB, seek to it and read a
                                ;   record from it.
                                ;
                                ;   Inputs: DS:DX point to an FCB
                                ;   Outputs:       AL = error codes defined above
                                ;
                                ;-----------------------------------------------------------------------------

                                _$FCB_RANDOM_READ:
                                    ;mov    AL,6
00002274 B006                       MOV     AL,RANDOM+FCBREAD
                                    ; 23/01/2024
                                    ;jmp    FCBIO          ; single block
00002276 EB02                       jmp     short FCBIO

                                ;BREAK <$FCB_Random_Write - write a single record to a file >
                                ;-----------------------------------------------------------------------------
                                ;
                                ;   $FCB_Random_Write - retrieve a location from the FCB, seek to it and write
                                ;   a record to it.
                                ;
                                ;   Inputs: DS:DX point to an FCB
                                ;   Outputs:       AL = error codes defined above
                                ;
                                ;-----------------------------------------------------------------------------

                                _$FCB_RANDOM_WRITE:
                                    ;mov    AL,2
00002278 B002                       MOV     AL,RANDOM
                                    ; 23/01/2024
                                    ;;jmp   FCBIO
                                    ;jmp    short FCBIO

                                ;BREAK <FCBIO - do internal FCB I/O>
                                ;-----------------------------------------------------------------------------
                                ;
                                ;   FCBIO - look at FCBOP and merge all FCB operations into a single routine.
                                ;
                                ;   Inputs: FCBOP flags which operations need to be performed
                                ;           DS:DX point to FCB
                                ;           CX may have count of number of records to xfer
                                ;   Outputs:       AL has error code
                                ;   Registers modified: all
                                ;-----------------------------------------------------------------------------

                                    ; 09/11/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
                                    ; DOSCODE:5B17h (MSDOS 5.0 MSDOS.SYS)

                                    ; 23/01/2024
                                    ; DOSCODE:5B2Bh (MSDOS 6.22 MSDOS.SYS)

                                    ; 23/01/2024 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMDOS.COM)
                                    ; DOSCODE:61C9h (PCDOS 7.1 IBMDOS.COM)

                                FCBIO:

                                FEOF    EQU     1
                                FTRIM   EQU     2

                                %define     FCBErr  byte [bp-1]  ; byte
                                %define     cRec    word [bp-3]  ; word
                                ;%define RecPos    word [bp-7]  ; dword
                                %define RecPosL     word [bp-7]  ; word
                                %define RecPosH     word [bp-5]  ; word
                                %define     RecSize word [bp-9]  ; word
                                ;%define bPos      word [bp-13] ; dword
                                %define bPosL       word [bp-13] ; word
                                %define bPosH       word [bp-11] ; word
                                %define cByte       word [bp-15] ; word
                                %define cResult word [bp-17] ; word
                                %define     cRecRes word [bp-19] ; word
                                %define     FCBOp   byte [bp-20] ; byte
                                ; 23/01/2024
                                %define bPos bp-13

                                    ;Enter

0000227A 55                         push    bp
0000227B 89E5                       mov     bp,sp
0000227D 83EC14                     sub     sp,20
                                    ;mov    [bp-20],al
00002280 8846EC                     MOV     FCBOp,AL
                                    ;mov    byte [bp-1],0
00002283 C646FF00                   MOV     FCBErr,0               ;   FCBErr = 0;
00002287 E8C0FF                     call    GetExtended            ;   FCB = GetExtended ();
                                    ;test   byte [bp-20],8
0000228A F646EC08                   TEST    FCBOp,BLOCK            ;   if ((OP&BLOCK) == 0)
0000228E 7503                       JNZ     short GetPos
00002290 B90100                     MOV     CX,1                   ;       cRec = 1;
                                GetPos:
                                    ;mov    [bp-3],cx
00002293 894EFD                     MOV     cRec,CX                ;*Tail coalesce
00002296 E885FF                     call    GetExtent              ;   RecPos = GetExtent ();
00002299 E8BBFF                     call    GetRecSize             ;   RecSize = GetRecSize ();
                                    ;mov    [bp-9],bx
```

```
                                    MOV     RecSize,BX
                                    ;test   byte [bp-20],2
0000229F F646EC02                   TEST    FCBOp,RANDOM            ;   if ((OP&RANDOM) <> 0)
000022A3 7403                       JZ      short GetRec
000022A5 E868FF                     call    GetRR                   ;       RecPos = GetRR ();
                                GetRec:
                                    ;mov    [bp-7],ax
000022A8 8946F9                     MOV     RecPosL,AX              ;*Tail coalesce
                                    ;mov    [bp-5],dx
000022AB 8956FB                     MOV     RecPosH,DX
000022AE E87FFF                     call    SetExtent               ;   SetExtent (RecPos);
                                    ;mov    ax,[bp-5]
000022B1 8B46FB                     MOV     AX,RecPosH              ;   bPos = RecPos * RecSize;
000022B4 F7E3                       MUL     BX
000022B6 89C7                       MOV     DI,AX
                                    ;mov    ax,[bp-7]
000022B8 8B46F9                     MOV     AX,RecPosL
000022BB F7E3                       MUL     BX
000022BD 01FA                       ADD     DX,DI
                                    ;mov    [bp-13],ax
000022BF 8946F3                     MOV     bPosL,AX
                                    ;mov    [bp-11],dx
000022C2 8956F5                     MOV     bPosH,DX
                                    ;mov    ax,[bp-3]
000022C5 8B46FD                     MOV     AX,cRec                 ;   cByte = cRec * RecSize;
000022C8 F7E3                       MUL     BX
                                    ;mov    [bp-15],ax
000022CA 8946F1                     MOV     cByte,AX

                                ;hkn;       SS override
000022CD 360306[2C03]               ADD     AX,[SS:DMAADD]          ;   if (cByte+DMA > 64K) {
000022D2 83D200                     ADC     DX,0
000022D5 7419                       JZ      short DoOper
                                    ;mov    byte [bp-1],2
000022D7 C646FF02                   MOV     FCBErr,FTRIM            ;       FCBErr = FTRIM;

                                ;hkn;       SS override
000022DB 36A1[2C03]                 MOV     AX,[SS:DMAADD]          ;       cRec = (64K-DMA)/RecSize;
000022DF F7D8                       NEG     AX
000022E1 7501                       JNZ     short DoDiv
000022E3 48                         DEC     AX
                                DoDiv:
000022E4 31D2                       XOR     DX,DX
000022E6 F7F3                       DIV     BX
                                    ;mov    [bp-3],ax
000022E8 8946FD                     MOV     cRec,AX
000022EB F7E3                       MUL     BX                      ;       cByte = cRec * RecSize;
                                    ;mov    [bp-15],ax
000022ED 8946F1                     MOV     cByte,AX                ;       }
                                DoOper:
000022F0 31DB                       XOR     BX,BX
                                    ;mov    [bp-17],bx
000022F2 895EEF                     MOV     cResult,BX              ;   cResult = 0;
                                    ;cmp    [bp-15],bx
000022F5 395EF1                     CMP     cByte,BX                ;   if (cByte <> 0 ||
000022F8 7506                       JNZ     short DoGetExt
                                    ;test   byte [bp-1],2
000022FA F646FF02                   TEST    FCBErr,FTRIM            ;       (FCBErr&FTRIM) == 0) {
                                    ;JZ     short DoGetExt
                                    ;JMP    short SkipOp
                                    ; 16/12/2022
000022FE 756E                       jnz     short SkipOp
                                    ; 09/11/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
                                    ;JZ     short DoGetExt
                                    ;JMP    short SkipOp
                                DoGetExt:
00002300 E89AFE                     call    SFTFromFCB              ;       if (!SFTFromFCB (SFT,FCB))
00002303 730F                       JNC     short ContinueOp
                                FCBDeath:
00002305 E885E3                     call    FCB_RET_ERR             ; signal error, map for extended
                                    ;mov    word [bp-19],0
00002308 C746ED0000                 MOV     cRecRes,0               ; no bytes transferred
                                    ;mov    byte [bp-1],1
0000230D C646FF01                   MOV     FCBErr,FEOF             ;           return FTRIM;
00002311 E9E700                     JMP     FCBSave                 ; bam!
                                ContinueOp:
                                    ; 23/01/2024
                                    ; (PCDOS 7.1 IBMDOS.COM)
                                    ;
                                    ;;mov   ax,[si+10h]
                                    ;MOV    AX,[SI+SYS_FCB.FILSIZ]
                                    ;;mov   [es:di+11h],ax
                                    ;MOV    [ES:DI+SF_ENTRY.sf_size],AX
                                    ;;mov   ax,[si+12h]
                                    ;MOV    AX,[SI+SYS_FCB.FILSIZ+2]
                                    ;;mov   [es:di+13h],ax
                                    ;MOV    [ES:DI+SF_ENTRY.sf_size+2],AX
                                    ;;;
00002314 1E                         push    ds
00002315 C54410                     lds     ax,[si+SYS_FCB.FILSIZ]
00002318 26894511                   mov     [es:di+SF_ENTRY.sf_size],ax
0000231C 268C5D13                   mov     [es:di+SF_ENTRY.sf_size+2],ds
00002320 C546F3                     lds     ax,[bPos] ; lds ax,[bp-13]
00002323 8CDA                       mov     dx,ds
00002325 1F                         pop     ds
                                    ;;;
                                    ;;mov   ax,[bp-13]
                                    ;MOV    AX,bPosL
                                    ;;mov   dx,[bp-11]
                                    ;MOV    DX,bPosH

                                    ;mov    [es:di+15h],ax
00002326 26894515                   MOV     [ES:DI+SF_ENTRY.sf_position],AX
                                    ;xchg   dx,[es:di+17h]
0000232A 26875517                   XCHG    [ES:DI+SF_ENTRY.sf_position+2],DX
0000232E 52                         PUSH    DX                      ; save away Open age.
                                    ;mov    cx,[bp-15]
0000232F 8B4EF1                     MOV     CX,cByte                ;       cResult =

                                ;hkn; DOS_Read is in DOSCODE
00002332 BF[773B]                   MOV     DI,DOS_READ             ;           *(OP&FCBRead ? DOS_Read
                                    ;test   byte [bp-20],4
00002335 F646EC04                   TEST    FCBOp,FCBREAD           ;                : DOS_Write)(cRec);
00002339 7503                       JNZ     short DoContext

                                ;hkn; DOS_Write is in DOSCODE
0000233B BF[783D]                   MOV     DI,DOS_WRITE
                                DoContext:
0000233E 55                         push    bp
0000233F 1E                         push    ds
00002340 56                         push    si

                                ;hkn; SS is DOSDATA
00002341 16                         push    ss
```

```
16106 00002342 1F                          pop     ds
16107
16108                                   ;; Fix for disk full
16109 00002343 FFD7                        CALL    DI      ; DOS_READ or DOS_WRITE
16110
16111 00002345 5E                          pop     si
16112 00002346 1F                          pop     ds
16113 00002347 5D                          pop     bp
16114 00002348 72BB                        JC      short FCBDeath
16115
16116 0000234A 36803E[0B06]00              CMP     BYTE [SS:DISK_FULL],0 ; treat disk full as error
16117 00002350 7406                        JZ      short NODSKFULL
16118 00002352 36C606[0B06]00              MOV     BYTE [SS:DISK_FULL],0 ; clear the flag
16119
16120                                      ; 23/01/2024
16121                                      ; (PCDOS 7.1 IBMDOS.COM)
16122                                      ;;mov   byte [bp-1],1
16123                                      ;MOV    FCBErr,FEOF           ; set disk full flag
16124
16125                                   NODSKFULL:
16126                                   ;; Fix for disk full
16127                                      ;mov    [bp-17],cx
16128 00002358 894EEF                      MOV     cResult,CX
16129 0000235B E80AFB                      call    SaveFCBInfo           ;       SaveFCBInfo (FCB);
16130                                      ;pop    word [es:di+17h]
16131 0000235E 268F4517                    POP     WORD [ES:DI+SF_ENTRY.sf_position+2] ; restore open age
16132                                                                    ; (sf_OpenAge = SF_ENTRY.sf_position+2)
16133
16134                                      ; 23/01/2024
16135                                      ; (PCDOS 7.1 IBMDOS.COM)
16136                                      ;
16137                                      ;;mov   ax,[es:di+11h]
16138                                      ;MOV    AX,[ES:DI+SF_ENTRY.sf_size]
16139                                      ;;mov   [si+10h],ax
16140                                      ;MOV    [SI+SYS_FCB.FILSIZ],AX
16141                                      ;;mov   ax,[es:di+13h]
16142                                      ;MOV    AX,[ES:DI+SF_ENTRY.sf_size+2]
16143                                      ;;mov   [si+12h],ax
16144                                      ;MOV    [SI+SYS_FCB.FILSIZ+2],AX
16145                                      ;;;
16146 00002362 06                          push    es
16147 00002363 26C44511                    les     ax,[es:di+SF_ENTRY.sf_size]
16148 00002367 894410                      mov     [si+SYS_FCB.FILSIZ],ax
16149 0000236A 8C4412                      mov     [si+SYS_FCB.FILSIZ+2],es
16150 0000236D 07                          pop     es
16151                                      ;;;
16152                                                                    ;       }
16153                                   SkipOp:
16154                                      ;mov    ax,[bp-17]
16155 0000236E 8B46EF                      MOV     AX,cResult            ;   cRecRes = cResult / RecSize;
16156 00002371 31D2                        XOR     DX,DX
16157                                      ;div    word [bp-9]
16158 00002373 F776F7                      DIV     RecSize
16159                                      ;mov    [bp-19],ax
16160 00002376 8946ED                      MOV     cRecRes,AX
16161                                      ;add    [bp-7],ax
16162 00002379 0146F9                      ADD     RecPosL,AX            ;   RecPos += cRecResult;
16163                                      ;adc    word [bp-5],0
16164 0000237C 8356FB00                    ADC     RecPosH,0
16165
16166                                   ; If we have not gotten the expected number of records, we signal an EOF
16167                                   ; condition. On input, this is EOF. On output this is usually disk full.
16168                                   ; BUT... Under 2.0 and before, all device output IGNORED this condition. So
16169                                   ; do we.
16170
16171                                      ;cmp    ax,[bp-3]
16172 00002380 3B46FD                      CMP     AX,cRec               ;   if (cRecRes <> cRec)
16173 00002383 7411                        JZ      short TryBlank
16174                                      ;test   byte [bp-20],4
16175 00002385 F646EC04                    TEST    FCBOp,FCBREAD         ;       if (OP&FCBRead || !DEVICE)
16176 00002389 7507                        JNZ     short SetEOF
16177                                      ; 07/12/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
16178                                      ; MSDOS 3.3
16179                                      ;;test  word [es:di+5],80h
16180                                      ;TEST   word [ES:DI+SF_ENTRY.sf_flags],devid_device
16181                                      ;JNZ    short TryBlank
16182                                      ; MSDOS 5.0 & MSDOS 6.0
16183                                      ;test   byte [es:di+5],80h
16184 0000238B 26F6450580                  test    byte [ES:DI+SF_ENTRY.sf_flags],devid_device
16185 00002390 7504                        jnz     short TryBlank
16186
16187                                   SetEOF:
16188                                      ;mov    byte [bp-1],1
16189 00002392 C646FF01                    MOV     FCBErr,FEOF           ;       FCBErr = FEOF;
16190                                   TryBlank:                        ;
16191 00002396 09D2                        OR      DX,DX                 ;   if (cResult%RecSize <> 0) {
16192 00002398 7426                        JZ      short SetExt
16193                                      ;add    word [bp-7],1
16194 0000239A 8346F901                    ADD     RecPosL,1             ;       RecPos++;
16195                                      ;adc    word [bp-5],0
16196 0000239E 8356FB00                    ADC     RecPosH,0
16197                                      ;test   byte [bp-20],4
16198 000023A2 F646EC04                    TEST    FCBOp,FCBREAD         ;       if(OP&FCBRead) <> 0) {
16199 000023A6 7418                        JZ      short SetExt
16200                                      ;inc    word [bp-19]
16201 000023A8 FF46ED                      INC     cRecRes               ;       cRecRes++;
16202                                      ;mov    byte [bp-1],3
16203 000023AB C646FF03                    MOV     FCBErr,FTRIM+FEOF     ;       FCBErr = FTRIM | FEOF;
16204                                      ;mov    cx,[bp-9]
16205 000023AF 8B4EF7                      MOV     CX,RecSize            ;       Blank (RecSize-cResult%RecSize,
16206 000023B2 29D1                        SUB     CX,DX                 ;              DMA+cResult);
16207 000023B4 30C0                        XOR     AL,AL
16208                                   ;hkn;       SS override
16209 000023B6 36C43E[2C03]                les      di,[ss:DMAADD]
16210                                      ;add    di,[bp-17]
16211 000023BB 037EEF                      ADD     DI,cResult
16212 000023BE F3AA                        REP     STOSB                 ;   }  }
16213                                   SetExt:
16214                                      ;mov    dx,[bp-5]
16215 000023C0 8B56FB                      MOV     DX,RecPosH
16216                                      ;mov    ax,[bp-7]
16217 000023C3 8B46F9                      MOV     AX,RecPosL
16218                                      ;test   byte [bp-20],2
16219 000023C6 F646EC02                    TEST    FCBOp,RANDOM          ;   if ((OP&Random) == 0 ||
16220 000023CA 7406                        JZ      short DoSetExt
16221                                      ;test   byte [bp-20],8
16222 000023CC F646EC08                    TEST    FCBOp,BLOCK           ;       (OP&BLOCK) <> 0)
16223 000023D0 7403                        JZ      short TrySetRR
16224                                   DoSetExt:
16225 000023D2 E85BFE                      call    SetExtent             ;       SetExtent (RecPos, FCB);
16226                                   TrySetRR:
16227                                      ;test   byte [bp-20],8
16228 000023D5 F646EC08                    TEST    FCBOp,BLOCK           ;   if ((op&BLOCK) <> 0)
16229 000023D9 740F                        JZ      short TryReturn
```

```
                                    ;mov    [si+21h],ax
000023DB 894421                     MOV     [SI+SYS_FCB.RR],AX      ;       FCB->RR = RecPos;
                                    ;mov    [si+23h],dl
000023DE 885423                     MOV     [SI+SYS_FCB.RR+2],DL
                                    ;cmp    word [si+0Eh],64
000023E1 837C0E40                   CMP     word [SI+SYS_FCB.RECSIZ],64
000023E5 7303                       JAE     short TryReturn
                                    ;mov    [si+24h],dh
000023E7 887424                     MOV     [SI+SYS_FCB.RR+2+1],DH; Set 4th byte only if record size < 64
                                TryReturn:
                                    ;test   byte [bp-20],4
000023EA F646EC04                   TEST    FCBOp,FCBREAD           ;   if (!(FCBOP & FCBREAD)) {
000023EE 750B                       JNZ     short FCBSave
000023F0 1E                         push    ds                      ;       FCB->FDate = date;
000023F1 E86CE7                     call    DATE16                  ;       FCB->FTime = time;
000023F4 1F                         pop     ds
                                    ;mov    [si+14h],ax
000023F5 894414                     MOV     [SI+SYS_FCB.FDATE],AX
                                    ;mov    [si+16h],dx
000023F8 895416                     MOV     [SI+SYS_FCB.FTIME],DX ;       }
                                FCBSave:
                                    ;test   byte [bp-20],8
000023FB F646EC08                   TEST    FCBOp,BLOCK             ;   if ((op&BLOCK) <> 0)
000023FF 7409                       jz      short DoReturn
                                    ;mov    cx,[bp-19]
00002401 8B4EED                     MOV     CX,cRecRes              ;       user_CX = cRecRes;
00002404 E870E0                     call    Get_User_Stack
                                    ;mov    [si+4],cx
00002407 894C04                     MOV     [SI+user_env.user_CX],CX
                                DoReturn:
                                    ;mov    al,[bp-1]
0000240A 8A46FF                     MOV     AL,FCBErr               ;   return (FCBERR);
                                    ;Leave
0000240D 89EC                       mov     sp,bp
0000240F 5D                         pop     bp
00002410 C3                         retn

                                ; 22/07/2018 - Retro DOS v3.0

                                ;Break <$FCB_Open - open an old-style FCB>
                                ;--------------------------------------------------------------------------
                                ;
                                ;   $FCB_Open - CPM compatability file open. The user has formatted an FCB
                                ;   for us and asked to have the rest filled in.
                                ;
                                ;   Inputs: DS:DX point to an unopenned FCB
                                ;   Outputs:       AL indicates status 0 is ok FF is error
                                ;           FCB has the following fields filled in:
                                ;               Time/Date Extent/NR Size
                                ;--------------------------------------------------------------------------

                                    ; 23/01/2024 - Retro DOS v5.0
                                    ; PCDOS 7.1 IBMDOS.COM - DOSCODE:6362h

                                _$FCB_OPEN:             ; System call 15

                                    ;mov    ax,2
00002411 B80200                     MOV     AX,SHARING_COMPAT+open_for_both

                                ;hkn; DOS_Open is in DOSCODE
00002414 B9[FB31]                   MOV     CX,DOS_OPEN

                                ; The following is common code for Creation and openning of FCBs. AX is
                                ; either attributes (for create) or open mode (for open)... DS:DX points to
                                ; the FCB

                                DoAccess:
00002417 1E                         push    ds
00002418 52                         push    dx
00002419 51                         push    cx
0000241A 50                         push    ax                  ; save FCB pointer away

                                ;hkn;       OpenBuf is in DOSDATA
0000241B BF[BE03]                   MOV     DI,OPENBUF
0000241E E8DB57                     call    TransFCB            ; crunch the fcb
00002421 58                         pop     ax
00002422 59                         pop     cx
00002423 5A                         pop     dx
00002424 1F                         pop     ds                  ; get fcb
00002425 7303                       JNC     short FindFCB       ; everything seems ok
                                FCBOpenErr:
                                    ; AL has error code
00002427 E963E2                     jmp     FCB_RET_ERR
                                FindFCB:
0000242A E81DFE                     call    GetExtended         ; DS:SI will point to FCB

                                    ; 17/05/2019 - Retro DOS v4.0

                                    ; MSDOS 3.3
                                    ;call   LRUFCB
                                    ;jc     short HardMessage

                                    ; MSDOS 6.0
0000242D 50                         push    ax
0000242E B001                       mov     al,1                ;indicate Open/Create operation
00002430 E803FB                     call    LRUFCB              ; get a sft entry (no error)
00002433 58                         pop     ax
00002434 722A                       jc      short HardMessage

                                    ;mov    word [es:di+2],8000h
00002436 26C745020080               mov     word [es:di+SF_ENTRY.sf_mode],sf_isFCB
0000243C 1E                         push    ds
0000243D 56                         push    si
0000243E 53                         push    bx                  ; save fcb pointer
0000243F 89CE                       MOV     SI,CX

                                ;hkn; SS is DOSDATA
00002441 16                         push    ss
00002442 1F                         pop     ds                      ; let DOS_Open see variables
00002443 FFD6                       CALL    SI ; DOS_OPEN or DOS_CREATE ; go open the file
00002445 5B                         pop     bx
00002446 5E                         pop     si
00002447 1F                         pop     ds                  ; get fcb

                                ;hkn; SS override
00002448 36C43E[9E05]               LES     DI,[SS:THISSFT]             ; get sf pointer
0000244D 7318                       JNC     short FCBOK         ; operation succeeded
                                failopen:
0000244F 50                         PUSH    AX
00002450 B052                       MOV     AL,"R" ; 52h        ; clear out field (free sft)
00002452 E8C3FC                     call    BlastSFT
00002455 58                         POP     AX
                                    ;cmp    ax,4
00002456 83F804                     CMP     AX,error_too_many_open_files
```

```
16354 00002459 7405                     JZ      short HardMessage
16355                                   ;cmp    ax,24h
16356 0000245B 83F824                   CMP     AX,error_sharing_buffer_exceeded
16357 0000245E 7505                     jnz     short DeadFCB
16358                                 HardMessage:
16359 00002460 50                       PUSH    AX
16360 00002461 E86FFD                   call    FCBHardErr
16361 00002464 58                       POP     AX
16362                                 DeadFCB:
16363                                   ; 07/12/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
16364                                   ;jmp    FCB_RET_ERR
16365 00002465 EBC0                     jmp     short FCBOpenErr
16366                                 FCBOK:
16367                                   ; MSDOS 6.0
16368 00002467 E8D8F3                   call    IsSFTNet                ;AN007;F.C. >32mb  Non Fat file?
16369 0000246A 7531                     JNZ     short FCBOK2            ;AN007;F.C. >32mb  yes
16370 0000246C E8CB5F                   call    CheckShare              ;AN000;F.C. >32mb  share around?
16371                                   ;JNZ    short FCBOK2            ;AN000;F.C. >32mb  yes
16372                                   ; 23/01/2024 - Retro DOS v5.0
16373                                   ; (PCDOS 7.1 IBMDOS.COM)
16374 0000246F 750A                     jnz     short FCBOK1
16375
16376                                 ;SR;
16377                                 ; If we reach here, we know we have got a local SFT. Let's update the
16378                                 ; LocalSFT variable to reflect this.
16379
16380 00002471 36893E[760F]             mov     [ss:LocalSFT],di
16381 00002476 368C06[780F]             mov     [ss:LocalSFT+2],es; Store the SFT address
16382                                 ;;SR;
16383                                 ;; The check below is not valid anymore since we regenerate for media > 32M.
16384                                 ;;
16385                                 ;;  CMP     WORD [ES:DI+SF_ENTRY.sf_dirsec+2],0
16386                                 ;;                                      ;AN000;F.C. >32mb  if dirsec >32mb
16387                                 ;;  JZ      short FCBOK2                ;AN000;F.C. >32mb    then error
16388                                 ;;  MOV     AX,error_sys_comp_not_loaded  ;AN000;F.C. >32mb
16389                                 ;;  JMP     short failopen              ;AN000;F.C. >32mb
16390
16391                                   ; 23/01/2024 - Retro DOS v5.0
16392                                   ; (PCDOS 7.1 IBMDOS.COM)
16393                                   ;;;
16394                                 FCBOK1:
16395                                   ;test   byte [es:di+5],80h
16396 0000247B 26F6450580               test    byte [es:di+SF_ENTRY.sf_flags],devid_device
16397 00002480 751B                     jnz     short FCBOK2   ; local device
16398                                   ;test   byte [es:di+4],8
16399 00002482 26F6450408               test    byte [es:di+SF_ENTRY.sf_attr],attr_volume_id
16400 00002487 7514                     jnz     short FCBOK2
16401                                           ; local file
16402 00002489 06                       push    es
16403 0000248A 57                       push    di
16404                                   ;les    di,[es:di+07h]
16405 0000248B 26C47D07                 les     di,[es:di+SF_ENTRY.sf_devptr] ; local file's DPB
16406                                   ;cmp    word [es:di+0Fh],0
16407 0000248F 26837D0F00               cmp     word [es:di+DPB.FAT_SIZE],0 ; (16 bit FAT size)
16408 00002494 5F                       pop     di
16409 00002495 07                       pop     es
16410 00002496 7505                     jnz     short FCBOK2   ; not FAT32 (ok)
16411                                   ;mov    ax,0Fh         ; error_invalid_drive (for FAT32)
16412 00002498 B80F00                   mov     ax,error_invalid_drive
16413 0000249B EBB2                     jmp     short failopen
16414                                   ;;;
16415
16416                                 FCBOK2:
16417                                   ; MSDOS 6.0 (& MSDOS 3.3)
16418 0000249D 26FF05                   inc     word [es:di]
16419                                   ;INC    word [ES:DI+SF_ENTRY.sf_ref_count] ; increment reference count
16420
16421                                   ; 23/01/2024 - Retro DOS v5.0
16422                                   ; (PCDOS 7.1 IBMDOS.COM)
16423                                   ;;;
16424 000024A0 E87525                   call    set_sftfcb_entry
16425                                   ;;;
16426
16427 000024A3 E8C2F9                   call    SaveFCBInfo
16428
16429                                   ; MSDOS 3.3
16430                                   ;call   SetOpenAge
16431                                   ; MSDOS 6.0 (& MSDOS 3.3)
16432                                   ;test   word [es:di+5],80h
16433                                   ;TEST   word [ES:DI+SF_ENTRY.sf_flags],devid_device
16434 000024A6 26F6450580               test    byte [ES:DI+SF_ENTRY.sf_flags],devid_device  ; 28/07/2019
16435 000024AB 7508                     JNZ     short FCBNoDrive      ; do not munge drive on devices
16436
16437 000024AD 8A04                     MOV     AL,[SI]               ; get drive byte
16438 000024AF E8DB56                   call    GETTHISDRV            ; convert
16439                                   ;INC    AL
16440                                   ; 17/12/2022
16441 000024B2 40                       inc     ax
16442 000024B3 8804                     MOV     [SI],AL               ; stash in good drive letter
16443
16444                                 FCBNoDrive:
16445                                   ;mov    word [si+0Eh],128
16446 000024B5 C7440E8000               MOV     word [SI+SYS_FCB.RECSIZ],80h ; stuff in default record size
16447
16448                                   ; 23/01/2024 - Retro DOS v5.0
16449                                   ; (PCDOS 7.1 IBMDOS.COM)
16450                                   ;;;
16451                                   ;;mov   ax,[es:di+0Dh]
16452                                   ;MOV    AX,[ES:DI+SF_ENTRY.sf_time] ; set time
16453                                   ;;mov   [si+16h],ax
16454                                   ;MOV    [SI+SYS_FCB.FTIME],AX
16455                                   ;;mov   ax,[es:di+0Fh]
16456                                   ;MOV    AX,[ES:DI+SF_ENTRY.sf_date] ; set date
16457                                   ;;mov   [si+14h],ax
16458                                   ;MOV    [SI+SYS_FCB.FDATE],AX
16459                                   ;;mov   ax,[es:di+11h]
16460                                   ;MOV    AX,[ES:DI+SF_ENTRY.sf_size] ; set sizes
16461                                   ;;mov   [si+10h],ax
16462                                   ;MOV    [SI+SYS_FCB.FILSIZ],AX
16463                                   ;;mov   ax,[es:di+13h]
16464                                   ;MOV    AX,[ES:DI+SF_ENTRY.sf_size+2]
16465                                   ;;mov   [si+12h],ax
16466                                   ;MOV    [SI+SYS_FCB.FILSIZ+2],AX
16467                                   ;
16468 000024BA 06                       push    es
16469                                   ;les    ax,[es:di+0Dh]
16470 000024BB 26C4450D                 les     ax,[es:di+SF_ENTRY.sf_time]
16471                                   ;mov    [si+16h],ax
16472 000024BF 894416                   mov     [si+SYS_FCB.FTIME],ax ; set time
16473                                   ;mov    [si+14h],es
16474 000024C2 8C4414                   mov     [si+SYS_FCB.FDATE],es ; set date
16475 000024C5 07                       pop     es
16476 000024C6 06                       push    es
16477                                   ;les    ax,[es:di+11h]
```

```
16478 000024C7 26C44511                 les     ax,[es:di+SF_ENTRY.sf_size] ; set size
16479                                   ;mov    [si+10h],ax
16480 000024CB 894410                   mov     [si+SYS_FCB.FILSIZ],ax
16481                                   ;mov    [si+12h],ax
16482 000024CE 8C4412                   mov     [si+SYS_FCB.FILSIZ+2],es
16483 000024D1 07                       pop     es
16484                                   ;;;
16485
16486 000024D2 31C0                     XOR     AX,AX                   ; convenient zero
16487                                   ;mov    [si+0Ch],ax
16488 000024D4 89440C                   MOV     [SI+SYS_FCB.EXTENT],AX; point to beginning of file
16489
16490                                 ; We must scan the set of FCB SFTs for one that appears to match the current
16491                                 ; one.      We cheat and use CheckFCB to match the FCBs.
16492
16493                                 ;hkn;       SS override
16494 000024D7 36C43E[4000]                LES     DI,[SS:SFTFCB]          ; get the pointer to head of the list
16495                                   ;mov    ah,[es:di+4]
16496 000024DC 268A6504                 MOV     AH,[ES:DI+SFT.SFCount]; get number of SFTs to scan
16497                                 OpenScan:
16498                                   ;cmp    al,[si+18h]
16499 000024E0 3A4418                   CMP     AL,[SI+fcb_sfn]                 ; don't compare ourselves
16500 000024E3 7407                     JZ      short SkipCheck
16501 000024E5 50                       push    ax                      ; preserve count
16502 000024E6 E848FC                   call    CheckFCB                ; do they match
16503 000024E9 58                       pop     ax                      ; get count back
16504 000024EA 7309                     JNC     short OpenFound                 ; found a match!
16505                                 SkipCheck:
16506 000024EC FEC0                     INC     AL                      ; advance to next FCB
16507 000024EE 38E0                     CMP     AL,AH                   ; table full?
16508 000024F0 75EE                     JNZ     short OpenScan          ; no, go for more
16509                                 OpenDone:
16510 000024F2 30C0                     xor     al,al                   ; return success
16511 000024F4 C3                       retn
16512
16513                                 ; The SFT at ES:DI is the one that is already in use for this FCB. We set the
16514                                 ; FCB to use this one. We increment its ref count. We do NOT close it at all.
16515                                 ; Consider:
16516                                 ;
16517                                 ;   open (foo)     delete (foo) open (bar)
16518                                 ;
16519                                 ; This causes us to recycle (potentially) bar through the same local SFT as
16520                                 ; foo even though foo is no longer needed; this is due to the server closing
16521                                 ; foo for us when we delete it. Unfortunately, we cannot see this closure.
16522                                 ; If we were to CLOSE bar, the server would then close the only reference to
16523                                 ; bar and subsequent I/O would be lost to the redirector.
16524                                 ;
16525                                 ; This gets solved by NOT closing the sft, but zeroing the ref count
16526                                 ; (effectively freeing the SFT) and informing the sharer (if relevant) that
16527                                 ; the SFT is no longer in use. Note that the SHARER MUST keep its ref counts
16528                                 ; around. This will allow us to access the same file through multiple network
16529                                 ; connections and NOT prematurely terminate when the ref count on one
16530                                 ; connection goes to zero.
16531
16532                                 OpenFound:
16533                                   ;mov    [si+18h],al
16534 000024F5 884418                   MOV     [SI+fcb_sfn],AL         ; assign with this
16535 000024F8 26FF05                   inc     word [es:di]
16536                                   ;INC    word [ES:DI+SF_ENTRY.sf_ref_count]
16537                                                                   ; remember this new invocation
16538                                   ; 23/01/2024 - Retro DOS v5.0
16539                                   ; (PCDOS 7.1 IBMDOS.COM)
16540                                   ;;;
16541 000024FB E81A25                   call    set_sftfcb_entry
16542                                   ;;;
16543
16544                                   ; 24/01/2024
16545 000024FE 16                       push    ss
16546 000024FF 1F                       pop     ds
16547
16548                                   ;MOV    AX,[SS:FCBLRU]          ; update LRU counts
16549 00002500 A1[1000]                 mov     ax,[FCBLRU] ; 24/01/2024
16550                                   ;mov    [es:di+15h],ax
16551 00002503 26894515                 MOV     [ES:DI+sf_LRU],AX
16552                                 ;
16553                                 ; We have an FCB sft that is now of no use. We release sharing info and then
16554                                 ; blast it to prevent other reuse.
16555                                 ;
16556                                   ;push   ss
16557                                   ;pop    ds
16558
16559 00002507 C43E[9E05]                 LES     DI,[THISSFT]
16560 0000250B 26FF0D                   dec     word [es:di]
16561                                   ;DEC    word [ES:DI+SF_ENTRY.sf_ref_count]
16562                                                                   ; free the newly allocated SFT
16563 0000250E E8735F                   call    ShareEnd
16564 00002511 B043                     MOV     AL,'C'  ; 43h
16565 00002513 E802FC                   call    BlastSFT
16566 00002516 EBDA                     JMP     short OpenDone
16567
16568                                 ;BREAK      <$FCB_Create - create a new directory entry>
16569                                 ;-----------------------------------------------------------------------------
16570                                 ;
16571                                 ;   $FCB_Create - CPM compatability file create. The user has formatted an
16572                                 ;   FCB for us and asked to have the rest filled in.
16573                                 ;
16574                                 ;   Inputs: DS:DX point to an unopenned FCB
16575                                 ;   Outputs:        AL indicates status 0 is ok FF is error
16576                                 ;           FCB has the following fields filled in:
16577                                 ;               Time/Date Extent/NR Size
16578                                 ;-----------------------------------------------------------------------------
16579
16580                                 _$FCB_CREATE:               ; System call 22
16581
16582                                 ;hkn; DOS_Create is in DOSCODE
16583 00002518 B9[CA30]                 MOV     CX,DOS_CREATE           ; routine to call
16584 0000251B 31C0                     XOR     AX,AX                   ; attributes to create
16585 0000251D E82AFD                   call    GetExtended             ; get extended FCB
16586 00002520 7403                     JZ      short DoAccessJ                 ; not an extended FCB
16587 00002522 8A44FF                   MOV     AL,[SI-1]               ; get attributes
16588                                 DoAccessJ:
16589 00002525 E9EFFE                   JMP     DoAccess                ; do dirty work
16590
16591                                 ;=============================================================================
16592                                 ; SEARCH.ASM, MSDOS 6.0, 1991
16593                                 ;=============================================================================
16594                                 ; 22/07/2018 - Retro DOS v3.0
16595                                 ; 17/05/2019 - Retro DOS v4.0
16596
16597                                 ; DOSCODE:5DDFh (MSDOS 6.21, MSDOS.SYS)
16598
16599                                 ; 09/11/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
16600                                 ; DOSCODE:5DCBh (MSDOS 5.0, MSDOS.SYS)
16601
```

```
                                ;** Search.asm
                                ;--------------------------------------------------------------------------
                                ;   Directory search system calls.
                                ;   These will be passed direct text of the pathname from the user.
                                ;   They will need to be passed through the macro expander prior to
                                ;   being sent through the low-level stuff.
                                ;   I/O specs are defined in DISPATCH. The system calls are:
                                ;
                                ;   $Dir_Search_First       written
                                ;   $Dir_Search_Next        written
                                ;   $Find_First     written
                                ;   $Find_Next              written
                                ;   PackName                written
                                ;
                                ;   Modification history:
                                ;
                                ;     Created: ARR 4 April 1983

                                ;--------------------------------------------------------------------------
                                ; Procedure Name : $DIR_SEARCH_FIRST
                                ;
                                ; Inputs:
                                ;   DS:DX Points to unopenned FCB
                                ; Function:
                                ;   Directory is searched for first matching entry and the directory
                                ;   entry is loaded at the disk transfer address
                                ; Returns:
                                ;   AL = -1 if no entries matched, otherwise 0
                                ;--------------------------------------------------------------------------

                                    ; IBMDOS.COM (MSDOS 3.3) - Offset 2B88h

                                    ; 24/01/2024
                                    ; MSDOS 5.0 MSDOS.SYS - DOSCODE:5DCBh
                                    ; MSDOS 6.22 MSDOS.SYS - DOSCODE:5DDFh
                                    ; PCDOS 7.1 IBMDOS.COM - DOSCODE:647Bh
                                    ; Windows ME IO.SYS - BIOSCODE:61E5h

                                _$DIR_SEARCH_FIRST:
0000:2528 368916[A605]              MOV     [SS:THISFCB],DX
0000:252D 368C1E[A805]              MOV     [SS:THISFCB+2],DS
0000:2532 89D6                      MOV     SI,DX
0000:2534 803CFF                    CMP     BYTE [SI],0FFH
0000:2537 7503                      JNZ     short NORMFCB4
0000:2539 83C607                    ADD     SI,7                    ; Point to drive select byte
                                NORMFCB4:
0000:253C FF34                      push    word [SI]               ; Save original drive byte for later

0000:253E 16                        push    ss
0000:253F 07                        pop     es                      ; get es to address DOSGroup

0000:2540 BF[BE03]                  MOV     DI,OPENBUF              ; appropriate buffer
0000:2543 E8B656                    call    TransFCB                ; convert the FCB, set SATTRIB EXTFCB
0000:2546 7304                      JNC     short SearchIt          ; no error, go and look
0000:2548 5B                        pop     bx                      ; Clean stack

                                ; Error code is in AX

                                    ; 09/11/2022
                                dcf_errj:
0000:2549 E941E1                    jmp     FCB_RET_ERR             ; error

                                SearchIt:
0000:254C 16                        push    ss
0000:254D 1F                        pop     ds                      ; get ready for search
                                    ;push   word [DMAADD]
                                    ;push   word [DMAADD+2]
                                    ; 24/01/2024
0000:254E C43E[2C03]                les     di,[DMAADD]
0000:2552 57                        push    di
0000:2553 06                        push    es
0000:2554 C706[2C03][BE04]          MOV     WORD [DMAADD],SEARCHBUF
0000:255A 8C1E[2E03]                MOV     WORD [DMAADD+2],DS
                                    ; MSDOS 3.3
                                    ;call   DOS_SEARCH_FIRST
                                    ; MSDOS 6.0
0000:255E E8BD0F                    call    GET_FAST_SEARCH             ; search
0000:2561 8F06[2E03]                pop     word [DMAADD+2]
0000:2565 8F06[2C03]                pop     word [DMAADD]
0000:2569 735A                      JNC     short SearchSet             ; no error, transfer info
0000:256B 5B                        pop     bx                      ; Clean stack

                                ; Error code is in AX

                                    ; 09/11/2022
                                    ;jmp    FCB_RET_ERR
0000:256C EBDB                      jmp     short dcf_errj

                                ;--------------------------------------------------------------------------
                                ;
                                ; Procedure Name : $DIR_SEARCH_NEXT
                                ;
                                ; Inputs:
                                ;   DS:DX points to unopenned FCB returned by $DIR_SEARCH_FIRST
                                ; Function:
                                ;   Directory is searched for the next matching entry and the directory
                                ;   entry is loaded at the disk transfer address
                                ; Returns:
                                ;   AL = -1 if no entries matched, otherwise 0
                                ;--------------------------------------------------------------------------

                                    ; 24/01/2024
                                    ; MSDOS 5.0 MSDOS.SYS - DOSCODE:5E5Fh
                                    ; MSDOS 6.22 MSDOS.SYS - DOSCODE:5E73h
                                    ; PCDOS 7.1 IBMDOS.COM - DOSCODE:6517h
                                    ; Windows ME IO.SYS - BIOSCODE:6273h

                                _$DIR_SEARCH_NEXT:
0000:256E 368916[A605]              MOV     [SS:THISFCB],DX
0000:2573 368C1E[A805]              MOV     [SS:THISFCB+2],DS
                                    ; 24/01/2024 (PCDOS 7.1)
                                    ;MOV    byte [SS:SATTRIB],0
                                    ;MOV    byte [SS:EXTFCB],0
0000:2578 B000                      mov     al,0
0000:257A 36A2[6D05]                mov     [SS:SATTRIB],al ; 0
0000:257E 36A2[6D05]                mov     [SS:SATTRIB],al ; 0

0000:2582 16                        push    ss
0000:2583 07                        pop     es

0000:2584 BF[BE04]                  MOV     DI,SEARCHBUF

0000:2587 89D6                      MOV     SI,DX
0000:2589 803CFF                    CMP     BYTE [SI],0FFh
```

```
16726 0000258C 750D                     JNZ     short NORMFCB6
16727 0000258E 83C606                   ADD     SI,6
16728 00002591 AC                       LODSB

16730 00002592 36A2[6D05]               MOV     [SS:SATTRIB],AL
16731 00002596 36FE0E[6C05]             DEC     byte [SS:EXTFCB]
16732                                 NORMFCB6:
16733 0000259B AC                       LODSB                   ; Get original user drive byte
16734 0000259C 50                       push    ax              ; Put it on stack
16735 0000259D 8A4414                   MOV     AL,[SI+20]      ; Get correct search contin drive byte
16736 000025A0 AA                       STOSB                   ; Put in correct place
16737 000025A1 B90A00                   MOV     CX,20/2
16738 000025A4 F3A5                     REP     MOVSW           ; Transfer in rest of search contin info
16739
16740 000025A6 16                       push    ss
16741 000025A7 1F                       pop     ds
16742
16743                                   ;push   word [DMAADD]
16744                                   ;push   word [DMAADD+2]
16745                                   ; 24/01/2024
16746 000025A8 C43E[2C03]               les     di,[DMAADD]
16747 000025AC 57                       push    di
16748 000025AD 06                       push    es
16749 000025AE C706[2C03][BE04]         MOV     WORD [DMAADD],SEARCHBUF
16750 000025B4 8C1E[2E03]               MOV     WORD [DMAADD+2],DS
16751 000025B8 E86210                   call    DOS_SEARCH_NEXT ; Find it
16752 000025BB 8F06[2E03]               pop     word [DMAADD+2]
16753 000025BF 8F06[2C03]               pop     word [DMAADD]
16754 000025C3 7249                     JC      short SearchNoMore
16755                                   ; 24/01/2024
16756                                   ;JMP    SearchSet       ; Ok set return
16757
16758                                 ;;; ; 24/01/2024 - Retro DOS v5.0
16759
16760                                 ; The search was successful (or the search-next). We store the information
16761                                 ; into the user's FCB for continuation.
16762
16763                                 SearchSet:
16764 000025C5 BE[BE04]                 MOV     SI,SEARCHBUF
16765 000025C8 C43E[A605]               LES     DI,[THISFCB]    ; point to the FCB
16766 000025CC F606[6C05]FF             TEST    byte [EXTFCB],0FFh
16767 000025D1 7403                     JZ      short NORMFCB1
16768 000025D3 83C707                   ADD     DI,7            ; Point past the extension
16769                                 NORMFCB1:
16770 000025D6 5B                       pop     bx              ; Get original drive byte
16771 000025D7 08DB                     OR      BL,BL
16772 000025D9 7506                     JNZ     short SearchDrv
16773 000025DB 8A1E[3603]               MOV     BL,[CURDRV]
16774 000025DF FEC3                     INC     BL
16775                                 SearchDrv:
16776 000025E1 AC                       LODSB                   ; Get correct search contin drive byte
16777 000025E2 86C3                     XCHG    AL,BL           ; Search byte to BL, user byte to AL
16778 000025E4 47                       INC     DI
16779                                   ;STOSB                  ; Store the correct "user" drive byte
16780                                                           ;  at the start of the search info
16781 000025E5 B90A00                   MOV     CX,20/2
16782 000025E8 F3A5                     REP     MOVSW           ; Rest of search cont info, SI -> entry
16783 000025EA 86C3                     XCHG    AL,BL           ; User drive byte back to BL, search
16784                                                           ;   byte to AL
16785 000025EC AA                       STOSB                   ; Search contin drive byte at end of
16786                                                           ;   contin info
16787 000025ED C43E[2C03]               LES     DI,[DMAADD]
16788 000025F1 F606[6C05]FF             TEST    byte [EXTFCB],0FFh
16789 000025F6 740C                     JZ      short NORMFCB2
16790 000025F8 B0FF                     MOV     AL,0FFh
16791 000025FA AA                       STOSB
16792 000025FB FEC0                     INC     AL
16793                                   ;MOV    CX,5
16794                                   ; 17/12/2022
16795                                   ;mov    cl,5
16796                                   ;REP    STOSB
16797                                   ; 03/07/2024
16798 000025FD AB                       stosw
16799 000025FE AB                       stosw
16800 000025FF AA                       stosb
16801 00002600 A0[6D05]                 MOV     AL,[SATTRIB]
16802 00002603 AA                       STOSB
16803                                 NORMFCB2:
16804 00002604 88D8                     MOV     AL,BL           ; User Drive byte
16805 00002606 AA                       STOSB
16806                                   ;MOV    CX,16           ; 32 / 2 words of dir entry
16807                                   ; 17/12/2022
16808 00002607 B110                     mov     cl,16
16809 00002609 F3A5                     REP     MOVSW
16810 0000260B E97CE0                   jmp     FCB_RET_OK
16811                                 ;;;
16812
16813                                 SearchNoMore:
16814 0000260E C43E[A605]               LES     DI,[THISFCB]
16815 00002612 F606[6C05]FF             TEST    byte [EXTFCB],0FFh
16816 00002617 7403                     JZ      short NORMFCB8
16817 00002619 83C707                   ADD     DI,7            ; Point past the extension
16818                                 NORMFCB8:
16819 0000261C 5B                       pop     bx              ; Get original drive byte
16820 0000261D 26881D                   MOV     [ES:DI],BL      ; Store the correct "user" drive byte
16821                                                           ;  at the right spot
16822                                 ; error code is in AX
16823
16824 00002620 E96AE0                   jmp     FCB_RET_ERR
16825
16826                                 ; 17/05/2019 - Retro DOS v4.0
16827
16828                                 ; DOSCODE:5EE6h (MSDOS 6.21, MSDOS.SYS)
16829
16830                                 ;--------------------------------------------------------------------------
16831                                 ;
16832                                 ;   Procedure Name : $FIND_FIRST
16833                                 ;
16834                                 ;   Assembler usage:
16835                                 ;       MOV AH, FindFirst
16836                                 ;       LDS DX, name
16837                                 ;       MOV CX, attr
16838                                 ;       INT 21h
16839                                 ;   ; DMA address has datablock
16840                                 ;
16841                                 ;   Error Returns:
16842                                 ;       AX = error_path_not_found
16843                                 ;          = error_no_more_files
16844                                 ;--------------------------------------------------------------------------
16845
16846                                     ; 09/11/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
16847                                     ; DOSCODE:5ED2h (MSDOS 5.0, MSDOS.SYS)
16848
16849                                     ; 24/01/2024 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMDOS.COM)
```

```
                                    ; DOSCODE:6594h (PCDOS 7.1, IBMDOS.COM)

                                _$FIND_FIRST:
00002623 89D6                       MOV     SI,DX                   ; get name in appropriate place
00002625 36880E[6D05]               MOV     [SS:SATTRIB],CL                 ; Search attribute to correct loc

0000262A BF[BE03]                   MOV     DI,OPENBUF              ; appropriate buffer

0000262D E83256                     call    TransPathSet            ; convert the path
00002630 7305                       JNC     short Find_it           ; no error, go and look
                                FindError:
                                    ;mov    al,3
00002632 B003                       mov     al,error_path_not_found         ; error and map into one.
                                    ; 09/11/2022
                                FF_errj:
00002634 E941E0                     jmp     SYS_RET_ERR
                                Find_it:
00002637 16                         push    ss
00002638 1F                         pop     ds

                                    ;push   word [DMAADD]
                                    ;push   word [DMAADD+2]
                                    ; 24/01/2024 (PCDOS 7.1 IBMDOS.COM)
00002639 C43E[2C03]                 les     di,[DMAADD]
0000263D 57                         push    di
0000263E 06                         push    es

0000263F C706[2C03][BE04]           MOV     WORD [DMAADD],SEARCHBUF
00002645 8C1E[2E03]                 MOV     WORD [DMAADD+2],DS
                                    ; MSDOS 3.3
                                    ;call   DOS_SEARCH_FIRST
                                    ; MSDOS 6.0
00002649 E8D20E                     call    GET_FAST_SEARCH         ; search
0000264C 8F06[2E03]                 pop     word [DMAADD+2]
00002650 8F06[2C03]                 pop     word [DMAADD]

                                    ; 16/12/2022
                                    ;JNC    short FindSet           ; no error, transfer info
00002654 72DE                       jc      short FF_errj  ; jmp SYS_RET_ERR
                                    ;
                                    ;jmp    SYS_RET_ERR
                                    ; 09/11/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
                                ;FFF_errj:
                                    ;jmp    short FF_errj  ; jmp SYS_RET_ERR

                                FindSet:
00002656 BE[BE04]                   MOV     SI,SEARCHBUF
00002659 C43E[2C03]                 LES     DI,[DMAADD]
0000265D B91500                     MOV     CX,21
00002660 F3A4                       REP     MOVSB
00002662 56                         PUSH    SI                      ; Save pointer to start of entry
                                    ;mov    al,[si+0Bh]
00002663 8A440B                     MOV     AL,[SI+dir_entry.dir_attr]
00002666 AA                         STOSB
                                    ;add    si,16h ; 22
00002667 83C616                     ADD     SI,dir_entry.dir_time
0000266A A5                         MOVSW                           ; dir_time
0000266B A5                         MOVSW                           ; dir_date
0000266C 46                         INC     SI
0000266D 46                         INC     SI                      ; Skip dir_first
0000266E A5                         MOVSW                           ; dir_size (2 words)
0000266F A5                         MOVSW
00002670 5E                         POP     SI                      ; Point back to dir_name
00002671 E83300                     CALL    PackName
00002674 E9F7DF                     jmp     SYS_RET_OK              ; bye with no errors

                                ;--------------------------------------------------------------------------
                                ;
                                ;   Procedure Name : $FIND_NEXT
                                ;
                                ;   Assembler usage:
                                ;   ; dma points at area returned by find_first
                                ;       MOV AH, findnext
                                ;       INT 21h
                                ;   ; next entry is at dma
                                ;
                                ;   Error Returns:
                                ;       AX = error_no_more_files
                                ;--------------------------------------------------------------------------

                                    ; 09/11/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)

                                    ; 24/01/2024 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMDOS.COM)
                                    ; DOSCODE:65ECh (PCDOS 7.1, IBMDOS.COM)

                                _$FIND_NEXT:
00002677 16                         push    ss
00002678 07                         pop     es

00002679 BF[BE04]                   MOV     DI,SEARCHBUF

0000267C 36C536[2C03]               LDS     SI,[SS:DMAADD]

00002681 B91500                     MOV     CX,21
00002684 F3A4                       REP     MOVSB                   ; Put the search continuation info
                                                                    ;  in the right place
00002686 16                         push    ss
00002687 1F                         pop     ds                      ; get ready for search

                                    ; 24/01/2024 (Retro DOS v5-v4)
                                    ;push   word [DMAADD]
                                    ;push   word [DMAADD+2]
00002688 C43E[2C03]                 les     di,[DMAADD]
0000268C 57                         push    di
0000268D 06                         push    es
0000268E C706[2C03][BE04]           MOV     WORD [DMAADD],SEARCHBUF
00002694 8C1E[2E03]                 MOV     WORD [DMAADD+2],DS
00002698 E8820F                     call    DOS_SEARCH_NEXT         ; Find it
0000269B 8F06[2E03]                 pop     word [DMAADD+2]
0000269F 8F06[2C03]                 pop     word [DMAADD]
000026A3 73B1                       JNC     short FindSet           ; No error, set info
                                    ;jmp    SYS_RET_ERR
                                    ; 16/12/2022
000026A5 EB8D                       jmp     short FF_errj  ; jmp SYS_RET_ERR
                                    ; 09/11/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
                                    ;jmp    short FFF_errj ; jmp SYS_RET_ERR

                                ;--------------------------------------------------------------------------
                                ;** PackName - Convert file names from FCB to ASCIZ format.
                                ;
                                ;   PackName transfers a file name from DS:SI to ES:DI and converts it to
                                ;   the ASCIZ format.
                                ;
                                ;   ENTRY   (DS:SI) = 11 character FCB or dir entry name
```

```
16974                                   ;           (ES:DI) = destination area (13 bytes)
16975                                   ;   EXIT    (ds:SI) and (es:DI) advanced
16976                                   ;   USES    al, CX, SI, DI, Flags  (BUGBUG - not verified - jgl)
16977                                   ;--------------------------------------------------------------------------
16978
16979                                       ; 25/01/2024 - Retro DOS v5.0
16980                                       ; PCDOS 7.1 IBMDOS.COM - DOSCODE:6627h
16981
16982                                   PackName:
16983                                   ;   Move over 8 characters to cover the name component, then trim it's
16984                                   ;   trailing blanks.
16985
16986                                       ;MOV    CX,8                    ; Pack the name
16987                                       ;REP    MOVSB                   ; Move all of it
16988                                       ; 25/01/2024
16989 000026A7 B90400                      mov     cx,4
16990 000026AA F3A5                        rep     movsw
16991                                   main_kill_tail:
16992 000026AC 26807DFF20                  CMP     BYTE [ES:DI-1]," "
16993 000026B1 7507                        JNZ     short find_check_dot
16994 000026B3 4F                          DEC     DI                      ; Back up over trailing space
16995 000026B4 41                          INC     CX
16996 000026B5 83F908                      CMP     CX,8
16997 000026B8 72F2                        JB      short main_kill_tail
16998                                   find_check_dot:
16999                                       ;CMP    WORD [SI],(" " << 8) | " "
17000 000026BA 813C2020                    cmp     word [si],2020h
17001 000026BE 7506                        JNZ     short got_ext           ; Some chars in extension
17002 000026C0 807C0220                    CMP     BYTE [SI+2]," "
17003 000026C4 740F                        JZ      short find_done                 ; No extension
17004                                   got_ext:
17005 000026C6 B02E                        MOV     AL,"."  ; 2Eh
17006 000026C8 AA                          STOSB
17007                                       ;MOV    CX,3
17008                                       ;; 18/12/2022
17009                                       ;;mov   cl,3
17010                                       ;;REP   MOVSB
17011                                       ;movsb
17012                                       ;movsb
17013                                       ;movsb
17014                                       ; 25/01/2024
17015 000026C9 A5                          movsw
17016 000026CA A4                          movsb
17017                                   ext_kill_tail:
17018 000026CB 26807DFF20                  CMP     BYTE [ES:DI-1]," "
17019 000026D0 7503                        JNZ     short find_done
17020 000026D2 4F                          DEC     DI                      ; Back up over trailing space
17021 000026D3 EBF6                        JMP     short ext_kill_tail
17022                                   find_done:
17023 000026D5 31C0                        XOR     AX,AX
17024 000026D7 AA                          STOSB                           ; NUL terminate
17025 000026D8 C3                          retn
17026
17027                                   ;--------------------------------------------------------------------------
17028
17029                                   ; 24/01/2024
17030                                   %if 0
17031                                       ; 17/05/2019 - Retro DOS v4.0
17032                                   GET_FAST_SEARCH:
17033                                       ; 22/07/2018
17034                                       ; MSDOS 6.0
17035                                       ; 17/12/2022
17036                                       OR      byte [ss:DOS34_FLAG+1],(SEARCH_FASTOPEN>>8)  ; 04h
17037                                       ;OR     word [ss:DOS34_FLAG],SEARCH_FASTOPEN  ; 400h
17038                                                                       ;FO.trigger fastopen ;AN000;
17039                                       ;call   DOS_SEARCH_FIRST
17040                                       ;retn
17041                                       ; 17/12/2022
17042                                       jmp     DOS_SEARCH_FIRST
17043                                   %endif
17044
17045                                   ;==========================================================================
17046                                   ; PATH.ASM, MSDOS 6.0, 1991
17047                                   ;==========================================================================
17048                                   ; 06/08/2018 - Retro DOS v3.0
17049                                   ; 17/05/2019 - Retro DOS v4.0
17050
17051                                   ; DOSCODE:5FB0h (MSDOS 6.21, MSDOS.SYS)
17052
17053                                   ;** Directory related system calls. These will be passed direct text of the
17054                                   ;   pathname from the user. They will need to be passed through the macro
17055                                   ;   expander prior to being sent through the low-level stuff. I/O specs are
17056                                   ;   defined in DISPATCH. The system calls are:
17057                                   ;
17058                                   ;   $CURRENT_DIR  Written
17059                                   ;   $RMDIR    Written
17060                                   ;   $CHDIR    Written
17061                                   ;   $MKDIR    Written
17062                                   ;
17063                                   ;
17064                                   ;   Modification history:
17065                                   ;
17066                                   ;       Created: ARR 4 April 1983
17067                                   ;            MZ 10 May 1983     CurrentDir implemented
17068                                   ;            MZ 11 May 1983     RmDir, ChDir, MkDir implemented
17069                                   ;            EE 19 Oct 1983     RmDir no longer allows you to delete a
17070                                   ;                               current directory.
17071                                   ;            MZ 19 Jan 1983     Brain damaged applications rely on success
17072
17073                                   ;   I_Need  ThisCDS,DWORD          ; pointer to Current CDS
17074                                   ;   I_Need  WFP_Start,WORD         ; pointer to beginning of directory text
17075                                   ;   I_Need  Curr_Dir_End,WORD      ; offset to end of directory part
17076                                   ;   I_Need  OpenBuf,128            ; temp spot for translated name
17077                                   ;   I_need  fSplice,BYTE           ; TRUE => do splice
17078                                   ;   I_Need  NoSetDir,BYTE          ; TRUE => no exact match on splice
17079                                   ;   I_Need  cMeta,BYTE
17080                                   ;   I_Need  DrvErr,BYTE                                   ;AN000;
17081
17082                                   ;BREAK <$CURRENT_DIR - dump the current directory into user space>
17083                                   ;--------------------------------------------------------------------------
17084                                   ;
17085                                   ;   Procedure Name : $CURRENT_DIR
17086                                   ;
17087                                   ;   Assembler usage:
17088                                   ;           LDS     SI,area
17089                                   ;           MOV     DL,drive
17090                                   ;           INT     21h
17091                                   ;       ; DS:SI is a pointer to 64 byte area that contains drive
17092                                   ;       ; current directory.
17093                                   ;   Error returns:
17094                                   ;       AX = error_invalid_drive
17095                                   ;
17096                                   ;--------------------------------------------------------------------------
17097
```

```
                                    ; 06/08/2018 - Retro DOS v3.0
                                    ; IBMDOS.COM (MSDOS 3.3, 1987) - Offset 2D4Eh

                                    ; 25/01/2024 - Retro DOS v5.0
                                    ; MSDOS 5.0 MSDOS.SYS - DOSCODE:5F9Ch  ; Retro DOS v4.1 (& v4.0)
                                    ; MSDOS 6.22 MSDOS.SYS - DOSCODE:5FB0h ; Retro DOS v4.2
                                    ; Windows ME IO.SYS - BIOSCODE:6393h
                                    ; PCDOS 7.1 IBMDOS.COM - DOSCODE:6664h ; Retro DOS v5.0

                                _$CURRENT_DIR:
000026D9 E806F2                     call    ECritDisk
000026DC 88D0                       MOV     AL,DL                   ; get drive number (0=def, 1=A)
000026DE E89054                     call    GetVisDrv               ; grab it
000026E1 7310                       JNC     short CurrentValidate ; no error -> go and validate dir
                                CurdirErr:
000026E3 E829F2                     call    LCritDisk

                                    ; MSDOS 3.3
                                    ;mov    al,0Fh

                                    ; MSDOS 6.0
000026E6 1E                         push    ds
000026E7 2E8E1E[0700]               mov     ds,[cs:DosDSeg]
000026EC A0[1006]                   mov     al,[DrvErr]             ;IFS.                   ;AN000;
000026EF 1F                         pop     ds

                                curdir_errj:
000026F0 E985DF                     jmp     SYS_RET_ERR             ;IFS. make noise        ;AN000;

                                CurrentValidate:
000026F3 1E                         push    ds                      ; save destination
000026F4 56                         push    si

                                    ;LDS    SI,[CS:THISCDS] ; MSDOS 3.3

                                    ; MSDOS 6.0
000026F5 2E8E1E[0700]               mov     ds,[cs:DosDSeg]
                                    ; 25/01/2024 (PCDOS 7.1 IBMDOS.COM)
000026FA C606[4C03]00               mov     byte [NoSetDir],0 ; *

                                    ; 25/01/2024
                                    ;lds     si,[THISCDS]

                                ; 16/12/2022
                                %if 0
                                    ; 09/11/2022 (following test instruction is nonsense!)
                                    ; (I am leaving it here for MSDOS 5.0 MSDOS.SYS compatibility)

                                    ;test   word [si+43h],8000h
                                    TEST    word [SI+curdir.flags],curdir_isnet
                                    ;jnz    short $+2  ; 09/11/2022
                                    jnz     short DoCheck
                                %endif

                                ; Random optimization nuked due to some utilities using GetCurrentDir to do
                                ; media check.
                                ;   CMP     word [SI+curdir.ID],0
                                ;   JZ      short GetDst
                                DoCheck:
                                    ;MOV    byte [cs:NoSetDir],0  ; interested only in contents

                                    ; 25/01/2024
                                    ; MSDOS 6.0
                                    ;push   ds
                                    ;mov    ds,[cs:DosDSeg]
                                    ;mov    byte [NoSetDir],0 ; *
                                    ;pop    ds

000026FF BF[BE03]                   MOV     DI,OPENBUF
00002702 E85B27                     call    ValidateCDS             ; output is ES:DI -> CDS

00002705 06                         push    es                      ; swap source and destination
00002706 57                         push    di
00002707 5E                         pop     si
00002708 1F                         pop     ds
                                GetDst:
00002709 5F                         pop     di
0000270A 07                         pop     es                      ; get real destination
0000270B 72D6                       JC      short CurdirErr
                                    ;ADD    SI,curdir.text ; add si,0 ; 09/08/2018
                                    ;
                                    ; 09/11/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
                                    ; DOSCODE:5FE2h (MSDOS 5.0, MSDOS.SYS)
                                    ; 16/12/2022
                                    ;add    si,0  ; add si,curdir.text
                                    ;
                                    ;add    si,[si+4Fh] ; 17/05/2019
0000270D 03744F                     ADD     SI,[SI+curdir.end]
00002710 803C5C                     CMP     BYTE [SI],'\'  ; 5Ch  ; root or subdirs present?
00002713 7501                       JNZ     short CurrentCopy
00002715 46                         INC     SI
                                CurrentCopy:
                                ;   call    FStrCpy
                                ;; 10/29/86 E5 char
00002716 50                         PUSH    AX
00002717 AC                         LODSB                           ; get char
00002718 08C0                       OR      AL,AL
0000271A 7413                       JZ      short FOK
0000271C 3C05                       CMP     AL,05H
0000271E 740D                       JZ      short FCHANGE
00002720 EB01                       JMP     short FFF
                                FCPYNEXT:
00002722 AC                         LODSB                           ; get char
                                FFF:
00002723 3C5C                       CMP     AL,'\'                  ; beginning of directory
00002725 7508                       JNZ     short FOK               ; no
00002727 AA                         STOSB                           ; put into user's buffer
00002728 AC                         LODSB                           ; 1st char of dir is 05?
00002729 3C05                       CMP     AL,05H
0000272B 7502                       JNZ     short FOK               ; no
                                FCHANGE:
0000272D B0E5                       MOV     AL,0E5H                 ; make it E5
                                FOK:
0000272F AA                         STOSB                           ; put into user's buffer
00002730 08C0                       OR      AL,AL                   ; final char
00002732 75EE                       JNZ     short FCPYNEXT          ; no
00002734 58                         POP     AX

                                ;; 10/29/86 E5 char
00002735 30C0                       xor     AL,AL                   ; MZ 19 Jan 84
00002737 E8D5F1                     call    LCritDisk
0000273A E931DF                     jmp     SYS_RET_OK              ; no more, bye!

                                ; 17/05/2019 - Retro DOS v4.0
```

```
17222
17223                                   ; DOSCODE:6029h (MSDOS 6.21, MSDOS.SYS)
17224
17225                                   ;BREAK <$RmDir -- Remove a directory>
17226                                   ;--------------------------------------------------------------------------
17227                                   ;
17228                                   ; Procedure Name : $RmDir
17229                                   ;
17230                                   ; Inputs:
17231                                   ;   DS:DX Points to asciz name
17232                                   ; Function:
17233                                   ;   Delete directory if empty
17234                                   ; Returns:
17235                                   ;   STD XENIX Return
17236                                   ;   AX = error_path_not_found If path bad
17237                                   ;   AX = error_access_denied If
17238                                   ;           Directory not empty
17239                                   ;           Path not directory
17240                                   ;           Root directory specified
17241                                   ;           Directory malformed (. and .. not first two entries)
17242                                   ;           User tries to delete a current directory
17243                                   ;   AX = error_current_directory
17244                                   ;
17245                                   ;--------------------------------------------------------------------------
17246
17247                                       ; 10/11/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
17248                                       ; DOSCODE:6015h (MSDOS 5.0, MSDOS.SYS)
17249
17250                                       ; 25/01/2025 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMDOS.COM)
17251                                       ; DOSCODE:66C8h (PCDOS 7.1, IBMDOS.COM)
17252
17253                                   _$RMDIR:
17254 0000273D 52                         push  dx                      ; Save ptr to name
17255 0000273E 1E                         push  ds
17256 0000273F 89D6                       mov   si,dx                   ; Load ptr into si
17257 00002741 BF[BE03]                   mov   di,OPENBUF              ; di = ptr to buf for trans name
17258 00002744 57                         push  di
17259 00002745 E82255                     call  TransPathNoSet          ; Translate the name
17260 00002748 5F                         pop   di                      ; di = ptr to buf for trans name
17261 00002749 7306                       jnc   short rmlset            ; If transpath succeeded, continue
17262 0000274B 1F                         pop   ds
17263 0000274C 5A                         pop   dx                      ; Restore the name
17264                                     ;mov   al,3
17265 0000274D B003                       mov   al,error_path_not_found ; Otherwise, return an error
17266                                     ; 16/12/2022
17267                                   rmdir_errj: ; 10/08/2018
17268                                   chdir_errj:
17269 0000274F EB9F                       jmp   short curdir_errj
17270                                     ;jmp   SYS_RET_ERR
17271                                   rmlset:
17272 00002751 36803E[7A05]FF             CMP   byte [ss:CMETA],-1      ;   if (cMeta >= 0)
17273 00002757 7518                       Jnz   short rmerr             ;       return (-1);
17274 00002759 16                         push  ss
17275 0000275A 07                         pop   es
17276 0000275B 30C0                       xor   al,al                   ; al = 0 , ie drive a:
17277                                   rmloop:
17278 0000275D E87C54                     call  GetCDSFromDrv           ; Get curdir for drive in al
17279 00002760 7215                       jc    short rmcont            ; If error, exit loop & cont normally
17280
17281                                     ; 25/01/2024 - Retro DOS v5.0
17282                                     ; (PCDOS 7.1 IBMDOS.COM)
17283                                     ;;;
17284                                     ;;test  word [si+43h],4000h
17285                                     ;test   word [si+curdir.flags],curdir_inuse
17286 00002762 F6444440                   test  byte [si+curdir.flags+1],(curdir_inuse>>8)
17287 00002766 7405                       jz    short rmdir_nxt
17288                                     ;;;
17289
17290 00002768 E824F0                     call  StrCmp                  ; Are the 2 paths the same?
17291 0000276B 7404                       jz    short rmerr             ; Yes, report error.
17292                                   rmdir_nxt:  ; 25/01/2024
17293 0000276D FEC0                       inc   al                      ; No, inc al to next drive number
17294 0000276F EBEC                       jmp   short rmloop            ; Go check next drive.
17295                                   rmerr:
17296 00002771 1F                         pop   ds
17297 00002772 5A                         pop   dx                      ; Restore ptr the name
17298                                     ;mov   al,10h
17299 00002773 B010                       mov   al,error_current_directory ; error
17300                                     ; 16/12/2022
17301                                     ; 10/11/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
17302                                   ;chdir_errj:
17303 00002775 EBD8                       jmp   short rmdir_errj
17304                                   rmcont:
17305 00002777 1F                         pop   ds
17306 00002778 5A                         pop   dx                      ; Restore ptr the name
17307 00002779 BE[7A3A]                   MOV   SI,DOS_RMDIR
17308
17309                                     ; 25/01/2024 - Retro DOS v5.0
17310                                     ; (PCDOS 7.1 IBMDOS.COM)
17311                                     ;;;
17312 0000277C E8AAF0                     call  TestNet
17313                                     ;mov   di,67          ; DIRSTRLEN
17314                                     ; 26/06/2024
17315 0000277F B043                       mov   al,67
17316 00002781 7302                       jnc   short rmcont2   ; local directory
17317                                     ;mov   di,128
17318 00002783 B080                       mov   al,128
17319                                   rmcont2:
17320                                     ; 26/06/2024
17321                                     ;mov   [ss:PATHNAMELEN],di
17322 00002785 36A2[5411]                 mov   [ss:PATHNAMELEN],al
17323                                     ;;;
17324 00002789 E99900                     JMP   DoDirCall
17325
17326                                   ; 17/05/2019 - Retro DOS v4.0
17327
17328                                   ; DOSCODE:6065h (MSDOS 6.21, MSDOS.SYS)
17329
17330                                   ;BREAK <$ChDir -- Change current directory on a drive>
17331                                   ;--------------------------------------------------------------------------
17332                                   ;
17333                                   ; $ChDir - Top-level change directory system call.  This call is responsible
17334                                   ; for setting up the CDS for the specified drive appropriately.  There are
17335                                   ; several cases to consider:
17336                                   ;
17337                                   ;   o       Local, simple CDS.  In this case, we take the input path and convert
17338                                   ;   it into a WFP. We verify the existance of this directory and then
17339                                   ;   copy the WFP into the CDS and set up the ID field to point to the
17340                                   ;   directory cluster.
17341                                   ;   o       Net CDS.  We form the path from the root (including network prefix)
17342                                   ;   and verify its existance (via DOS_Chdir).  If successful, we copy the
17343                                   ;   WFP back into the CDS.
17344                                   ;   o       SUBST'ed CDS.  This is no different than the local, simple CDS.
17345                                   ;   o       JOIN'ed CDS.  This is trouble as there are two CDS's at work.  If we
```

```
                                ;   call TransPath, we will get the PHYSICAL CDS that the path refers to
                                ;   and the PHYSICAL WFP that the input path refers to.  This is perfectly
                                ;   good for the validation but not for currency.  We call TransPathNoSet
                                ;   to process the path but to return the logical CDS and the logical
                                ;   path.  We then copy the logical path into the logical CDS.
                                ;
                                ; Inputs:
                                ;   DS:DX Points to asciz name
                                ; Returns:
                                ;   STD XENIX Return
                                ;   AX = chdir_path_not_found if error
                                ;
                                ;--------------------------------------------------------------------------

                                    ; 25/01/2024 - Retro DOS v5.0 (PCDOS 7.1 IBMDOS.COM)

                                _$CHDIR:
                                    ; 25/01/2024
                                    ;;;
0000278C 36C606[5411]43             mov     byte [ss:PATHNAMELEN],67 ; Retro DOS v5.0
                                    ;mov    word [ss:PATHNAMELEN],67 ; DIRSTRLEN = 67
                                    ;;;
00002792 BF[BE03]                   MOV     DI,OPENBUF              ; spot for translated name
00002795 89D6                       MOV     SI,DX                   ; get source
00002797 E8C454                     call    TransPath               ; go munge the path and get real CDS
0000279A 7304                       JNC     short ChDirCrack        ; no errors, try path
                                ChDirErrP:
                                    ;mov    al,3
0000279C B003                       MOV     AL,error_path_not_found
                                ChDirErr:
                                    ;jmp    SYS_RET_ERR     ; oops!
                                    ; 10/11/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
0000279E EBAF                       jmp     short chdir_errj

                                ChDirCrack:
000027A0 803E[7A05]FF               CMP     byte [CMETA],-1             ; No meta chars allowed.
000027A5 75F5                       JNZ     short ChDirErrP
                                ; We cannot do a ChDir (yet) on a raw CDS. This is treated as a path not
                                ; found.

000027A7 C43E[A205]                     LES     DI,[THISCDS]
000027AB 83FFFF                         CMP     DI,-1                   ; if (ThisCDS == NULL)
000027AE 74EC                           JZ      short ChDirErrP                 ;    error ();

                                ; Find out if the directory exists.

000027B0 E87F12                         call    DOS_CHDIR
                                        ;Jc     short ChDirErr
                                        ; 16/12/2022
000027B3 729A                           jc      short chdir_errj
                                ;
                                ; Get back CDS to see if a join as seen. Set the currency pointer (only if
                                ; not network). If one was seen, all we need to do is copy in the text
                                ;
                                        ; 25/01/2024 - Retro DOS V5.0
000027B5 FF36[DC0A]                     push    word [DIRSTART_HW] ; *

000027B9 C43E[A205]                     LES     DI,[THISCDS]
                                        ;test   word [es:di+43h],2000h
                                        ; 17/12/2022
000027BD 26F6454420                     test    byte [ES:DI+curdir.flags+1],curdir_splice>>8
                                        ;TEST   word [ES:DI+curdir.flags],curdir_splice

                                        ; 25/01/2024 (PCDOS 7.1)
                                        ;mov    dx,[DIRSTART_HW]

000027C2 742B                           JZ      short GotCDS

                                ; The CDS was joined. Let's go back and grab the logical CDS.

000027C4 06                             push    es
000027C5 57                             push    di
                                        ;push   dx ; 25/01/2024 (PCDOS 7.1)
000027C6 51                             push    cx ; word [DIRSTART]  ; save CDS and cluster...
000027C7 E8ADDC                         call    Get_User_Stack        ; get original text

                                        ;mov    di,[si+6]
000027CA 8B7C06                         MOV     DI,[SI+user_env.user_DX]
                                        ;mov    ds,[si+0Eh]
000027CD 8E5C0E                         MOV     DS,[SI+user_env.user_DS]

000027D0 BE[BE03]                       MOV     SI,OPENBUF              ; spot for translated name
000027D3 87F7                           XCHG    SI,DI
000027D5 30C0                           XOR     AL,AL                   ; do no splicing
000027D7 57                             push    di
000027D8 E88F54                         call    TransPathNoSet          ; Munge path
000027DB 5E                             pop     si

                                ; There should NEVER be an error here.

                                ;IF FALSE
                                ;   JNC SKipErr
                                ;   fmt <>,<>,<"$p: Internal CHDIR error\n">
                                ;SkipErr:
                                ;ENDIF
000027DC C43E[A205]                     LES     DI,[THISCDS]            ; get new CDS
                                        ;mov    word [es:di+49h],-1
000027E0 26C74549FFFF                   MOV     word [ES:DI+curdir.ID],-1
                                                                        ; no valid cluster here...
                                        ;;; 25/01/2024 (PCDOS 7.1)
                                        ;mov    word [es:di+4Bh],-1
000027E6 26C7454BFFFF                   mov     word [es:di+curdir.ID+2],-1
                                        ;;;
000027EC 59                             pop     cx ; word [DIRSTART]
                                        ;pop    dx ; 25/01/2024 (PCDOS 7.1)
000027ED 5F                             pop     di
000027EE 07                             pop     es

                                ; ES:DI point to the physical CDS, CX is the ID (local only)

                                GotCDS:
                                    ; 25/01/2024 - Retro DOS v5.0
000027EF 5A                         pop     dx ; word [DIRSTART_HW] ; *

                                ; wfp_start points to the text. See if it is long enough

                                    ; MSDOS 3.3
                                    ;push   ss
                                    ;pop    ds
                                    ;mov    si,[WFP_START]
                                    ;push   cx
                                    ;call   DStrLen
                                    ;cmp    cx,67 ; cmp cx,DIRSTRLEN
```

```
                                    ;pop    cx
                                    ;ja     short ChDirErrP

                                    ; MSDOS 6.0
000027F0 E85C00                     CALL    Check_PathLen           ;PTM.           ;AN000;
000027F3 77A7                       JA      short ChDirErrP
                                    ; MSDOS 3.3 & MSDOS 6.0
                                    ;TEST   word [ES:DI+curdir.flags],curdir_isnet ; 8000h
                                    ; 17/12/2022
000027F5 26F6454480                 test    byte [ES:DI+curdir.flags+1],curdir_isnet>>8
000027FA 7518                       JNZ     short SkipRecency
                                    ; MSDOS 6.0
                                    ;test   word [es:di+43h],2000h
                                    ; 17/12/2022
000027FC 26F6454420                 test    byte [ES:DI+curdir.flags+1],curdir_splice>>8
                                    ;TEST   word [ES:DI+curdir.flags],curdir_splice
                                                                        ;PTM. for Join and Subst ;AN000;
00002801 7405                       JZ      short setdirclus        ;PTM.           ;AN000;
00002803 B9FFFF                     MOV     CX,-1                   ;PTM.           ;AN000;
                                    ;;; 25/01/2024 (PCDOS 7.1 IBMDOS.COM)
00002806 89CA                       mov     dx,cx
                                setdirclus:
                                    ;mov    [es:di+49h],cx
00002808 26894D49                   MOV     [ES:DI+curdir.ID],CX
                                    ;;; 25/01/2024 (PCDOS 7.1 IBMDOS.COM)
0000280C 2689554B                   mov     [es:di+curdir.ID+2],dx
                                    ;;;
00002810 C43E[A205]                 LES     DI,[THISCDS]            ; get logical CDS
                                SkipRecency:
00002814 E8A9EF                     call    FStrCpy
00002817 30C0                       XOR     AL,AL
                                mkdir_ok:
00002819 E952DE                     jmp     SYS_RET_OK

                                ; 17/05/2019 - Retro DOS v4.0

                                ; DOSCODE:60E1h (MSDOS 6.21, MSDOS.SYS)

                                ;BREAK <$MkDir - Make a directory entry>
                                ;--------------------------------------------------------------------------
                                ;
                                ; Procedure Name : $MkDir
                                ; Inputs:
                                ;   DS:DX Points to asciz name
                                ; Function:
                                ;   Make a new directory
                                ; Returns:
                                ;   STD XENIX Return
                                ;   AX = mkdir_path_not_found if path bad
                                ;   AX = mkdir_access_denied  If
                                ;           Directory cannot be created
                                ;           Node already exists
                                ;           Device name given
                                ;           Disk or directory(root) full
                                ;--------------------------------------------------------------------------

                                    ; 10/11/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)

                                    ; 25/01/2024 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMDOS.COM)
                                    ; PCDOS 7.1 IBMDOS.COM - DOSCODE:67B0h

                                _$MKDIR:
                                    ;MOV    SI,DOS_MKDIR
                                    ;;;
                                    ; 25/01/2024 (PCDOS 7.1 IBMDOS.COM)
0000281C 36C606[5411]43             mov     byte [ss:PATHNAMELEN],67 ; Retro DOS v5.0
                                    ;mov    word [ss:PATHNAMELEN],67 ; DIRSTRLEN (standard length = 67)
                                mkdir_x:    ; (windows) extended length (128)
00002822 BE[1D39]                   mov     si,DOS_MKDIR
                                    ;;;
                                DoDirCall:
00002825 BF[BE03]                   MOV     DI,OPENBUF              ; spot for translated name

00002828 56                         push    si
00002829 89D6                       MOV     SI,DX                   ; get source
0000282B E83054                     call    TransPath               ; go munge the path
0000282E 5E                         pop     si
0000282F 7305                       JNC     short MkDirCrack        ; no errors, try path
                                MkErrP:
00002831 B003                       MOV     AL,error_path_not_found       ; oops!
                                MkErr:
00002833 E942DE                     jmp     SYS_RET_ERR
                                MkDirCrack:
00002836 36803E[7A05]FF             CMP     byte [SS:CMETA],-1
0000283C 75F3                       JNZ     short MkErrP

                                    ; MSDOS 3.3
                                    ;push   ss
                                    ;pop    ds
                                    ;call   si
                                    ;jb     short MkErr
                                    ;;jmp   short mkdir_ok
                                    ;jmp    SYS_RET_OK

                                    ; MSDOS 6.0
0000283E 56                         PUSH    SI                      ;PTM.                   ;AN000;
                                                ; 25/01/2024 (PCDOS 7.1 IBMDOS.COM)
                                                ; check path len > [PATNAMELEN] ; 67 or 128
0000283F E80D00                     CALL    Check_PathLen           ;PTM. check path len > 67 ? ;AN000;
00002842 5E                         POP     SI                      ;PTM.                   ;AN000;
00002843 7604                       JBE     short pathok            ;PTM.                   ;AN000;
                                    ;mov    al,5
00002845 B005                       MOV     AL,error_access_denied;PTM. ops!
                                    ;jmp    SYS_RET_ERR             ;PTM.
00002847 EBEA                       jmp     short MkErr
                                pathok:
00002849 FFD6                       CALL    SI                      ; go get file
0000284B 72E6                       JC      short MkErr             ; error
                                    ; 16/12/2022
                                    ; 10/11/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
0000284D EBCA                       jmp     short mkdir_ok          ; ok
                                    ;jmp    SYS_RET_OK

                                ;--------------------------------------------------------------------------
                                ;
                                ; Procedure Name : Check_PathLen
                                ;
                                ; Inputs:
                                ;   nothing
                                ; Function:
                                ;   check if final path length greater than 67
                                ; Returns:
                                ;   Above flag set if > 67
                                ;
```

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

                                    ; 25/01/2024 - Retro DOS v5.0
                                    ; (PCDOS 7.1 IBMDOS.COM)

                                Check_PathLen:
                                    ; 09/09/2018
                                    ;mov    SI,[WFP_START]
0000284F 368B36[B205]               MOV     SI,[SS:WFP_START] ; MSDOS 6.0
                                Check_PathLen2:
                                    ; 25/01/2024 - Retro DOS v5.0
                                    ; Note: dx is not changed in 'Check_PathLen';
                                    ;       no need to push/pop (*)
                                    ;
00002854 16                         push    ss
00002855 1F                         pop     ds
                                    ;mov    SI,[WFP_START]   ; MSDOS 3.3
00002856 51                         push    CX
                                    ; (PCDOS 7.1 IBMDOS.COM)
                                    ;push   dx ; 25/01/2024       (*)
00002857 E87DEF                     CALL    DStrLen
                                    ; (PCDOS 7.1 IBMDOS.COM)
0000285A 3B0E[5411]                 cmp     cx,[PATHNAMELEN] ; 67 or 128
                                    ;CMP    CX,DIRSTRLEN ; 67
                                    ;pop    dx
0000285E 59                         POP     CX
0000285F C3                         retn

                                ;==========================================================================
                                ; IOCTL.ASM, MSDOS 6.0, 1991
                                ;==========================================================================
                                ; 07/08/2018 - Retro DOS v3.0
                                ; 17/05/2019 - Retro DOS v4.0

                                ;** IOCTL system call.
                                ;--------------------------------------------------------------------------
                                ;   $IOCTL
                                ;
                                ;     Revision history:
                                ;
                                ;          Created: ARR 4 April 1983
                                ;
                                ;          GenericIOCTL added:          KGS     22 April 1985
                                ;
                                ;          A000    version 4.00   Jan. 1988
                                ;
                                ;          Used jump table to dispatch IOCTL functions. HKN 3/12/90
                                ;

                                ;BREAK <IOCTL - munge on a handle to do device specific stuff>
                                ;--------------------------------------------------------------------------
                                ;
                                ;   Assembler usage:
                                ;       MOV     BX, Handle
                                ;       MOV     DX, Data
                                ;
                                ;   (or LDS     DX,BUF
                                ;       MOV     CX,COUNT)
                                ;
                                ;       MOV     AH, Ioctl
                                ;       MOV     AL, Request
                                ;       INT     21h
                                ;
                                ;   AH = 0  Return a combination of low byte of sf_flags and device driver
                                ;       attribute word in DX, handle in BX:
                                ;       DH = high word of device driver attributes
                                ;       DL = low byte of sf_flags
                                ;    1  Set the bits contained in DX to sf_flags.  DH MUST be 0.  Handle
                                ;       in BX.
                                ;    2  Read CX bytes from the device control channel for handle in BX
                                ;       into DS:DX.  Return number read in AX.
                                ;    3  Write CX bytes to the device control channel for handle in BX from
                                ;       DS:DX.  Return bytes written in AX.
                                ;    4  Read CX bytes from the device control channel for drive in BX
                                ;       into DS:DX.  Return number read in AX.
                                ;    5  Write CX bytes to the device control channel for drive in BX from
                                ;       DS:DX.  Return bytes written in AX.
                                ;    6  Return input status of handle in BX. If a read will go to the
                                ;       device, AL = 0FFh, otherwise 0.
                                ;    7  Return output status of handle in BX. If a write will go to the
                                ;       device, AL = 0FFh, otherwise 0.
                                ;    8  Given a drive in BX, return 1 if the device contains non-
                                ;       removable media, 0 otherwise.
                                ;    9  Return the contents of the device attribute word in DX for the
                                ;       drive in BX.  0200h is the bit for shared.      1000h is the bit for
                                ;       network. 8000h is the bit for local use.
                                ;    A  Return 8000h if the handle in BX is for the network or not.
                                ;    B  Change the retry delay and the retry count for the system. BX is
                                ;       the count and CX is the delay.
                                ;
                                ;   Error returns:
                                ;       AX = error_invalid_handle
                                ;          = error_invalid_function
                                ;          = error_invalid_data
                                ;
                                ;--------------------------------------------------------------------------
                                ;
                                ;   This is the documentation copied from DOS 4.0 it is much better
                                ;   than the above
                                ;
                                ;   There are several basic forms of IOCTL calls:
                                ;
                                ;
                                ;   ** Get/Set device information:       **
                                ;
                                ;   ENTRY   (AL) = function code
                                ;             0 - Get device information
                                ;             1 - Set device information
                                ;           (BX) = file handle
                                ;           (DX) = info for "Set Device Information"
                                ;   EXIT    'C' set if error
                                ;             (AX) = error code
                                ;           'C' clear if OK
                                ;             (DX) = info for "Get Device Information"
                                ;   USES    ALL
                                ;
                                ;
                                ;   **  Read/Write Control Data From/To Handle  **
                                ;
                                ;   ENTRY   (AL) = function code
                                ;             2 - Read device control info
                                ;             3 - Write device control info
                                ;           (BX) = file handle
                                ;           (CX) = transfer count
```

```
17718                           ;           (DS:DX) = address for data
17719                           ;   EXIT    'C' set if error
17720                           ;             (AX) = error code
17721                           ;           'C' clear if OK
17722                           ;             (AX) = count of bytes transfered
17723                           ;   USES    ALL
17724                           ;
17725                           ;
17726                           ;   **  Read/Write Control Data From/To Block Device  **
17727                           ;
17728                           ;   ENTRY   (AL) = function code
17729                           ;             4 - Read device control info
17730                           ;             5 - Write device control info
17731                           ;           (BL) = Drive number (0=default, 1='A', 2='B', etc)
17732                           ;           (CX) = transfer count
17733                           ;           (DS:DX) = address for data
17734                           ;   EXIT    'C' set if error
17735                           ;             (AX) = error code
17736                           ;           'C' clear if OK
17737                           ;             (AX) = count of bytes transfered
17738                           ;   USES    ALL
17739                           ;
17740                           ;
17741                           ;   **  Get Input/Output Status  **
17742                           ;
17743                           ;   ENTRY   (AL) = function code
17744                           ;             6 - Get Input status
17745                           ;             7 - Get Output Status
17746                           ;           (BX) = file handle
17747                           ;   EXIT    'C' set if error
17748                           ;             (AX) = error code
17749                           ;           'C' clear if OK
17750                           ;             (AL) = 00 if not ready
17751                           ;             (AL) = FF if ready
17752                           ;   USES    ALL
17753                           ;
17754                           ;
17755                           ;   **  Get Drive Information  **
17756                           ;
17757                           ;   ENTRY   (AL) = function code
17758                           ;             8 - Check for removable media
17759                           ;             9 - Get device attributes
17760                           ;           (BL) = Drive number (0=default, 1='A', 2='B', etc)
17761                           ;   EXIT    'C' set if error
17762                           ;             (AX) = error code
17763                           ;           'C' clear if OK
17764                           ;             (AX) = 0/1 media is removable/fixed (func. 8)
17765                           ;             (DX) = device attribute word (func. 9)
17766                           ;   USES    ALL
17767                           ;
17768                           ;
17769                           ;   **  Get Redirected bit**
17770                           ;
17771                           ;   ENTRY   (AL) = function code
17772                           ;             0Ah - Network stuff
17773                           ;           (BX) = file handle
17774                           ;   EXIT    'C' set if error
17775                           ;             (AX) = error code
17776                           ;           'C' clear if OK
17777                           ;             (DX) = SFT flags word, 8000h set if network file
17778                           ;   USES    ALL
17779                           ;
17780                           ;
17781                           ;   **  Change sharer retry parameters  **
17782                           ;
17783                           ;   ENTRY   (AL) = function code
17784                           ;             0Bh - Set retry parameters
17785                           ;           (CX) = retry loop count
17786                           ;           (DX) = number of retries
17787                           ;   EXIT    'C' set if error
17788                           ;             (AX) = error code
17789                           ;           'C' clear if OK
17790                           ;   USES    ALL
17791                           ;
17792                           ;
17793                           ;   ================================================================
17794                           ;
17795                           ;   **  New Standard Control  **
17796                           ;
17797                           ;   ALL NEW IOCTL FACILITIES SHOULD USE THIS FORM.      THE OTHER
17798                           ;   FORMS ARE OBSOLETE.
17799                           ;
17800                           ;   ================================================================
17801                           ;
17802                           ;   ENTRY   (AL) = function code
17803                           ;             0Ch - Control Function subcode
17804                           ;           (BX) = File Handle
17805                           ;           (CH) = Category Indicator
17806                           ;           (CL) = Function within category
17807                           ;           (DS:DX) = address for data, if any
17808                           ;           (SI) = Passed to device as argument, use depends upon function
17809                           ;           (DI) = Passed to device as argument, use depends upon function
17810                           ;   EXIT    'C' set if error
17811                           ;             (AX) = error code
17812                           ;           'C' clear if OK
17813                           ;             (SI) = Return value, meaning is function dependent
17814                           ;             (DI) = Return value, meaning is function dependent
17815                           ;             (DS:DX) = Return address, use is function dependent
17816                           ;   USES    ALL
17817                           ;
17818                           ;    ============== Generic IOCTL Definitions for DOS 3.2 ============
17819                           ;     (See inc\ioctl.inc for more info)
17820                           ;
17821                           ;   ENTRY   (AL) = function code
17822                           ;             0Dh - Control Function subcode
17823                           ;           (BL) = Drive Number (0 = Default, 1= 'A')
17824                           ;           (CH) = Category Indicator
17825                           ;           (CL) = Function within category
17826                           ;           (DS:DX) = address for data, if any
17827                           ;           (SI) = Passed to device as argument, use depends upon function
17828                           ;           (DI) = Passed to device as argument, use depends upon function
17829                           ;
17830                           ;   EXIT    'C' set if error
17831                           ;             (AX) = error code
17832                           ;           'C' clear if OK
17833                           ;             (DS:DX) = Return address, use is function dependent
17834                           ;   USES    ALL
17835                           ;
17836                           ;--------------------------------------------------------------------------
17837
17838                               ; 17/05/2019 - Retro DOS v4.0
17839                               ; DOSCODE:611Eh (MSDOS 6.21, MSDOS.SYS)
17840
17841                               ; 11/11/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
```

```
                                    ; DOSCODE:610Ah (MSDOS 5.0, MSDOS.SYS)

                                    ; 14/01/2024 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMDOS.COM)
                                    ; DOSCODE:67F7h (PCDOS 7.1, IBMDOS.COM)

                                IOCTLJMPTABLE:     ;label  word
                                    ; MSDOS 3.3 (& MSDOS 6.0)
00002860 [9C28]                     dw      ioctl_getset_data      ; 0
00002862 [9C28]                     dw      ioctl_getset_data      ; 1
00002864 [E628]                     dw      ioctl_control_string   ; 2
00002866 [E628]                     dw      ioctl_control_string   ; 3
00002868 [7E2A]                     dw      ioctl_get_dev          ; 4
0000286A [7E2A]                     dw      ioctl_get_dev          ; 5
0000286C [FE28]                     dw      ioctl_status           ; 6
0000286E [FE28]                     dw      ioctl_status           ; 7
00002870 [D829]                     dw      ioctl_rem_media                ; 8
00002872 [212A]                     dw      ioctl_drive_attr       ; 9
00002874 [702A]                     dw      ioctl_handle_redir     ; A
00002876 [132A]                     dw      Set_Retry_Parameters   ; B
00002878 [1A29]                     dw      GENERICIOCTLHANDLE     ; C
0000287A [3429]                     dw      GENERICIOCTL           ; D
                                    ; MSDOS 6.0 (& MSDOS 3.3)
0000287C [1F2B]                     dw      ioctl_drive_owner      ; E
0000287E [1F2B]                     dw      ioctl_drive_owner      ; F
                                    ; MSDOS 6.0
00002880 [1A29]                     dw      query_handle_support   ; 10h
00002882 [3429]                     dw      query_device_support   ; 11h

                                    ; 11/11/2022
                                _$IOCTL:
00002884 8CDE                       MOV     SI,DS                  ; Stash DS for calls 2,3,4 and 5
00002886 16                         push    ss
00002887 1F                         pop     ds                     ;hkn; SS is DOSDATA

                                    ; MSDOS 3.3
                                    ;cmp    al,0Fh
                                    ; MSDOS 6.0
00002888 3C11                       cmp     al,11h                 ; al must be between 0 & 11h
0000288A 770D                       ja      short ioctl_bad_funj2  ; if not bad function #

                                    ; 14/01/2024
                                    ; 28/05/2019
                                    ;push   AX      ; 14/01/2024   ; Need to save AL for generic IOCTL
0000288C 89C7                       mov     di,ax                  ; di NOT a PARM
0000288E 81E7FF00                   and     di,0FFh                ; di = al
00002892 D1E7                       shl     di,1                   ; di = index into jmp table
                                    ;pop    AX                     ; Restore AL for generic IOCTL

00002894 2EFFA5[6028]               jmp     word [CS:DI+IOCTLJMPTABLE]

                                ioctl_bad_funj2:
00002899 E90401                     JMP     ioctl_bad_fun  ; 10/08/2018

                                ;--------------------------------------------------------------------------
                                ;
                                ; IOCTL: AL = 0,1
                                ;
                                ; ENTRY: DS = DOSDATA
                                ;
                                ;--------------------------------------------------------------------------

                                    ; 29/01/2024 - Retro DOS v5.0
                                ioctl_getset_data:
                                    ; MSDOS 6.0
0000289C E8344E                     call    SFFromHandle           ; ES:DI -> SFT
0000289F 7305                       JNC     short ioctl_check_permissions ; have valid handle
                                ioctl_bad_handle:
                                    ;mov    al,6
000028A1 B006                       mov     al,error_invalid_handle
                                ioctl_error:
000028A3 E9D2DD                     jmp     SYS_RET_ERR

                                ioctl_check_permissions:
000028A6 3C00                       CMP     AL,0
                                    ;mov    al,[es:di+5]
000028A8 268A4505                   MOV     AL,[ES:DI+SF_ENTRY.sf_flags]; Get low byte of flags
000028AC 7419                       JZ      short ioctl_read       ; read the byte

000028AE 08F6                       or      dh,dh
000028B0 7404                       JZ      short ioctl_check_device ; can I set with this data?
                                    ;mov    al,0Dh
000028B2 B00D                       mov     al,error_invalid_data ; no DH <> 0
                                    ;jmp    SYS_RET_ERR
000028B4 EBED                       jmp     short ioctl_error

                                ioctl_check_device:
000028B6 A880                       test    AL,devid_device  ; 80h; can I set this handle?
000028B8 74DF                       jz      short ioctl_bad_funj2
000028BA 80CA80                     OR      DL,devid_device        ; Make sure user doesn't turn off the
                                                                   ;   device bit!! He can muck with the
                                                                   ;   others at will.
                                    ;MOV    byte [EXTERR_LOCUS],errLOC_SerDev ; 4
                                    ; 29/01/2024
                                    ;;;
000028BD E82BEA                     call    set_exerr_locus_ser ; (PCDOS 7.1 IBMDOS.COM)
                                    ;;;
000028C0 26885505                   MOV     BYTE [ES:DI+SF_ENTRY.sf_flags],DL  ;AC000;MS.; Set flags
                                ioctl_ok:
000028C4 E9A7DD                     jmp     SYS_RET_OK

                                ioctl_read:
                                    ;MOV    byte [EXTERR_LOCUS],errLOC_Disk  ; 2
                                    ; 29/01/2024
                                    ;;;
000028C7 E81CEA                     call    set_exerr_locus_disk ; (PCDOS 7.1 IBMDOS.COM)
                                    ;;;
000028CA 30E4                       XOR     AH,AH
000028CC A880                       test    AL,devid_device        ; Should I set high byte
000028CE 740B                       JZ      short ioctl_no_high    ; no
                                    ;MOV    byte [EXTERR_LOCUS],errLOC_SerDev ; 4
                                    ; 29/01/2024
                                    ;;;
000028D0 E818EA                     call    set_exerr_locus_ser ; (PCDOS 7.1 IBMDOS.COM)
                                    ;;;
                                    ;les    di,[es:di+7]
000028D3 26C47D07                   LES     DI,[ES:DI+SF_ENTRY.sf_devptr] ; Get device pointer
                                    ;mov    ah,[es:di+5]
000028D7 268A6505                   MOV     AH,[ES:DI+SYSDEV.ATT+1] ; Get high byte
                                ioctl_no_high:
000028DB 89C2                       MOV     DX,AX
                                ioctl_set_dx:      ; 16/12/2022
000028DD E897DB                     call    Get_User_Stack
                                    ;mov    [si+6],dx
000028E0 895406                     MOV     [SI+user_env.user_DX],DX
```

```
                                    ;;jmp   SYS_RET_OK
                                    ; 11/11/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
                                ioctl_ok_j:
                                    ; 16/12/2022
000028E3 E98BDD                     jmp     SYS_RET_OK_clc  ; (after 'Get_User_Stack')
                                    ;jmp    short ioctl_ok
                                    ; 26/07/2019
                                    ;jmp    SYS_RET_OK_clc

                                ;--------------------------------------------------------------------------
                                ;
                                ; IOCTL: AL = 2,3
                                ;
                                ; ENTRY: DS = DOSDATA
                                ;    SI = user's DS
                                ;
                                ;--------------------------------------------------------------------------

                                    ; 07/12/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
                                ioctl_control_string:
000028E6 E8EA4D                     call    SFFromHandle          ; ES:DI -> SFT
000028E9 72B6                       JC      short ioctl_bad_handle; invalid handle
                                    ; 07/12/2022
                                    ;TEST   word [ES:DI+SF_ENTRY.sf_flags],devid_device ; can I?
                                    ;jz     short ioctl_bad_funj2               ; No it is a file
                                    ; MSDOS 5.0 & MSDOS 6.0
000028EB 26F6450580                 test    byte [ES:DI+SF_ENTRY.sf_flags],devid_device ; can I?
000028F0 74A7                       jz      short ioctl_bad_funj2               ; No it is a file
                                    ;MOV    byte [EXTERR_LOCUS],errLOC_SerDev ; 4
                                    ; 29/01/2024
                                    ;;;
000028F2 E8F6E9                     call    set_exerr_locus_ser ; (PCDOS 7.1 IBMDOS.COM)
                                    ;;;
000028F5 26C47D07                   LES     DI,[ES:DI+SF_ENTRY.sf_devptr] ; Get device pointer
000028F9 30DB                       XOR     BL,BL                 ; Unit number of char dev = 0
000028FB E98801                     JMP     ioctl_do_string

                                ;--------------------------------------------------------------------------
                                ;
                                ; IOCTL: AL = 6,7
                                ;
                                ; ENTRY: DS = DOSDATA
                                ;
                                ;--------------------------------------------------------------------------

                                ioctl_status:
000028FE B401                       MOV     AH,1
00002900 2C06                       SUB     AL,6                  ; 6=0,7=1
00002902 7402                       JZ      short ioctl_get_status
00002904 B403                       MOV     AH,3
                                ioctl_get_status:
00002906 50                         PUSH    AX
00002907 E8BF15                     call    GET_IO_SFT
0000290A 58                         POP     AX
                                    ;JNC    short DO_IOFUNC
                                    ;JMP    short ioctl_bad_handle; invalid SFT
                                    ; 16/12/2022
0000290B 7294                       jc      short ioctl_bad_handle
                                DO_IOFUNC:
0000290D E8B827                     call    IOFUNC
00002910 88C4                       MOV     AH,AL
00002912 B0FF                       MOV     AL,0FFH
                                    ;JNZ    short ioctl_status_ret
                                    ; 29/01/2024
00002914 75AE                       jnz     short ioctl_ok
00002916 FEC0                       INC     AL
                                ioctl_status_ret:
                                    ;jmp    SYS_RET_OK
                                    ; 11/11/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
                                    ;jmp    short ioctl_ok_j
                                    ; 16/12/2022
00002918 EBAA                       jmp     short ioctl_ok

                                ;--------------------------------------------------------------------------
                                ;
                                ; Generic IOCTL entry point. AL = C, D, 10h, 11h
                                ;
                                ;   here we invoke the Generic IOCTL using the IOCTL_Req structure.
                                ;   SI:DX -> Users Device Parameter Table
                                ;   IOCALL -> IOCTL_Req structure
                                ;
                                ;   If on entry AL >= IOCTL_QUERY_HANDLE the function is a
                                ;   QueryIOCtlSupport call ELSE it's a standard generic IOCtl
                                ;   call.
                                ;
                                ; BUGBUG: Don't push anything on the stack between GENERIOCTL: and
                                ;         the call to Check_If_Net because Check_If_Net gets our
                                ;         return address off the stack if the drive is invalid.
                                ;
                                ;--------------------------------------------------------------------------

                                    ; 29/01/2024 - Retro DOS v5.0

                                query_handle_support:     ; Entry point for handles
                                GENERICIOCTLHANDLE:
0000291A E8B64D                     call    SFFromHandle          ; Get SFT for device.
0000291D 727E                       jc      short ioctl_bad_handlej

                                    ;test   word [es:di+5],8000h
                                    ;TEST   word [ES:DI+SF_ENTRY.sf_flags],sf_isnet     ; M031;
                                    ;test   byte [es:di+6],80h
0000291F 26F6450680                 test    byte [ES:DI+SF_ENTRY.sf_flags+1],(sf_isnet>>8)
00002924 757A                       jnz     short ioctl_bad_fun   ; Cannot do this over net.

                                    ;mov    byte [EXTERR_LOCUS],errLOC_SerDev ; 4
                                    ; 29/01/2024
                                    ;;;
00002926 E8C2E9                     call    set_exerr_locus_ser ; (PCDOS 7.1 IBMDOS.COM)
                                    ;;;

                                    ;les    di,[es:di+7]
00002929 26C47D07                   les     di,[es:di+SF_ENTRY.sf_devptr] ; Get pointer to device.
                                    ;;;
                                    ; 29/01/2024 - PCDOS 7.1 IBMDOS.COM
0000292D C606[B103]FF               mov     byte [IOCTL_drvnum],0FFh      ; invalidate drive number
                                                                          ; (for extended -lock/unlock- functions)
                                    ;;;
00002932 EB16                       jmp     short Do_GenIOCTL

                                    ; 29/01/2024 - Retro DOS v5.0
                                    ; PCDOS 7.1 IBMDOS.COM - DOSCODE:68DAh
                                    ; (WINME IO.SYS - BIOSCODE:6612h)

                                query_device_support:     ; Entry point for devices:
```

```
                                GENERICIOCTL:
                                    ;mov    byte [EXTERR_LOCUS],errLOC_Disk ; 2
                                    ; 29/01/2024
                                    ;;;
00002934 E8AFE9                     call    set_exerr_locus_disk
00002937 80FD48                     cmp     ch,48h                  ; category (extended, disk lock/unlock)
0000293A 7405                       je      short GenIOCTL_chk_net       ; extended (MSDOS/PCDOS 7)
                                    ;;;

0000293C 80FD08                     cmp     ch,IOC_DC ; 8           ; Only disk devices are allowed to use
0000293F 755F                       jne     short ioctl_bad_fun     ; no handles with Generic IOCTL.

                                    ; 29/01/2024
                                    ;;;
                                GenIOCTL_chk_net:
00002941 881E[B103]                 mov     [IOCTL_drvnum],bl       ; drive number
                                    ;;;

00002945 E8C801                     CALL    Check_If_Net            ; ES:DI := Get_hdr_block of device in BL
00002948 7556                       JNZ     short ioctl_bad_fun     ; There are no "net devices", and they

                                Do_GenIOCTL:
                                    ; 29/01/2024 - Retro DOS v5.0
                                    ;;;
0000294A 80FD48                     cmp     ch,48h                  ; category code 48h for FAT32
0000294D 7456                       je      short GenIOCTL_extended ; MSDOS/PCDOS 7 functions (lock/unlock)
                                    ;cmp    ch,8
0000294F 80FD08                     cmp     ch,IOC_DC ; 8                   ; disk control
00002952 7451                       je      short GenIOCTL_extended
                                GenIOCTL_normal:                    ; MSDOS 5-6.22 functions
                                    ;;;

                                    ;TEST   word [ES:DI+SYSDEV.ATT],DEV320
                                                                    ; Can device handle Generic IOCTL funcs
                                    ; 09/09/2018
                                    ;test   byte [es:di+4],40h
00002954 26F6450440                 TEST    byte [ES:DI+SYSDEV.ATT],DEV320 ; 0040h
00002959 7445                       jz      short ioctl_bad_fun

                                    ; 17/05/2019 - Retro DOS v4.0

                                    ; MSDOS 6.0
                                    ;mov    byte [IOCALL_REQFUNC],19 ; 13h
0000295B C606[7E03]13               mov     byte [IOCALL_REQFUNC],GENIOCTL ; Assume real Request
                                    ;cmp    al,10h
00002960 3C10                       cmp     AL,IOCTL_QUERY_HANDLE ; See if this is just a query
00002962 7C0C                       jl      short SetIOCtlBlock

                                    ;TEST   word [ES:DI+SYSDEV.ATT],IOQUERY ; See if device supports a query
                                    ;test   byte [es:di+4],80h
00002964 26F6450480                 TEST    byte [ES:DI+SYSDEV.ATT],IOQUERY ; See if device supports a query
00002969 7435                       jz      short ioctl_bad_fun     ; No support for query
                                    ;
                                    ;mov    byte [IOCALL_REQFUNC],19h
0000296B C606[7E03]19               mov     byte [IOCALL_REQFUNC],IOCTL_QUERY ; Just a query (5.00)

                                SetIOCtlBlock:
00002970 06                         PUSH    ES                      ; DEVIOCALL2 expects Device header block
00002971 57                         PUSH    DI                      ; in DS:SI
                                                                    ; Setup Generic IOCTL Request Block
                                    ;mov    byte [IOCALL_REQLEN],23
00002972 C606[7C03]17               mov     byte [IOCALL_REQLEN],IOCTL_REQ.size
                                    ; 07/09/2018 (MSDOS 3.3)
                                    ;;mov   byte [IOCALL_REQFUNC],19
                                    ;mov    byte [IOCALL_REQFUNC],GENIOCTL ; 07/09/2018
                                    ;
00002977 881E[7D03]                 MOV     [IOCALL_REQUNIT],BL
0000297B 882E[8903]                 MOV     [IOCALL+IOCTL_REQ.MAJORFUNCTION],CH
0000297F 880E[8A03]                 MOV     [IOCALL+IOCTL_REQ.MINORFUNCTION],CL
00002983 8936[8B03]                 MOV     [IOCALL+IOCTL_REQ.REG_SI],SI
00002987 893E[8D03]                 MOV     [IOCALL+IOCTL_REQ.REG_DI],DI
0000298B 8916[8F03]                 MOV     [IOCALL+IOCTL_REQ.GENERICIOCTL_PACKET],DX
0000298F 8936[9103]                 MOV     [IOCALL+IOCTL_REQ.GENERICIOCTL_PACKET+2],SI

                                ;hkn; IOCALL is in DOSDATA
00002993 BB[7C03]                   MOV     BX,IOCALL

00002996 16                         PUSH    SS
00002997 07                         POP     ES
                                                                    ; DS:SI -> Device header.
00002998 5E                         POP     SI
00002999 1F                         POP     DS
                                    ; 10/08/2018
0000299A E92201                     jmp     ioctl_do_IO             ; Perform Call to device driver

                                ioctl_bad_handlej:
0000299D E901FF                     jmp     ioctl_bad_handle

                                    ; 29/01/2024
                                ioctl_bad_fun:
000029A0 B001                       mov     al, error_invalid_function  ; 1
000029A2 E9D3DC                     jmp     SYS_RET_ERR

                                    ; 29/01/2024 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMDOS.COM)
                                GenIOCTL_extended:
000029A5 80F96A                     cmp     cl,6Ah                  ; UNLOCK LOGICAL VOLUME
                                    ;je     short GenIOCTL_chk_lock
000029A8 740E                       je      short GenIOCTL_lock_unlock
000029AA 80F94A                     cmp     cl,4Ah                  ; LOCK LOGICAL VOLUME
000029AD 75A5                       jne     short GenIOCTL_normal
                                ;GenIOCTL_chk_lock:
                                    ;cmp    cl,4Ah                  ; LOCK LOGICAL VOLUME
                                    ;jne    short GenIOCTL_lock_unlock
000029AF 80FF04                     cmp     bh,4                    ; lock level (0-4)
000029B2 7404                       je      short GenIOCTL_lock_unlock
000029B4 08FF                       or      bh,bh
000029B6 75E8                       jnz     short ioctl_bad_fun
                                GenIOCTL_lock_unlock:
000029B8 8A1E[B103]                 mov     bl,[IOCTL_drvnum]       ; drive number (1=A:, 2=B: ..)
000029BC 30FF                       xor     bh,bh
000029BE 4B                         dec     bx
000029BF 80FB1A                     cmp     bl,26                   ; logical disk number limit
000029C2 73DC                       jnb     short ioctl_bad_fun
000029C4 80F96A                     cmp     cl,6Ah                  ; UNLOCK LOGICAL VOLUME
000029C7 7507                       jne     short GenIOCTL_lock
000029C9 80A7[0813]7F               and     byte [bx+drive_flags],7Fh ; UNLOCK
000029CE EB05                       jmp     short GenIOCTL_OK
                                GenIOCTL_lock:
000029D0 808F[0813]80               or      byte [bx+drive_flags],80h ; LOCK
                                GenIOCTL_OK:
000029D5 E996DC                     jmp     SYS_RET_OK

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

```
                                ; IOCTL: AL = 8
                                ;
                                ; ENTRY: DS = DOSDATA
                                ;
                                ; BUGBUG: Don't push anything on the stack between ioctl_rem_media: and
                                ;         the call to Check_If_Net because Check_If_Net gets our
                                ;         return address off the stack if the drive is invalid.
                                ;
                                ;--------------------------------------------------------------------------

                                    ; 30/01/2024
                                ioctl_rem_media:
                                    ; MSDOS 3.3 (& MSDOS 6.0)
000029D8 E83501                     CALL    Check_If_Net
000029DB 75C3                       JNZ     short ioctl_bad_fun   ; There are no "net devices", and they
                                                                  ;   certainly don't know how to do this
                                                                  ;   call.
                                    ;test   word [es:di+4],800h
                                    ;TEST   word [ES:DI+SYSDEV.ATT],DEVOPCL ; See if device can
                                    ;test   byte [es:di+5],8
000029DD 26F6450508                 TEST    byte [es:di+SYSDEV.ATT+1],(DEVOPCL>>8)
000029E2 74BC                       JZ      short ioctl_bad_fun           ; NO

                                ;hkn; SS override for IOCALL
                                    ; 30/01/2024
                                    ; ds = ss = DOSDATA segment ('Get_Driver_BL' in 'Check_If_Net')
                                    ;MOV    byte [SS:IOCALL_REQFUNC],DEVRMD ; 15
000029E4 C606[7E03]0F               mov     byte [IOCALL_REQFUNC],DEVRMD ; 15
000029E9 B00D                       MOV     AL,REMHL  ; 13
000029EB 88DC                       MOV     AH,BL                 ; Unit number
                                    ;MOV    [SS:IOCALL_REQLEN],AX
000029ED A3[7C03]                   mov     [IOCALL_REQLEN],ax
000029F0 31C0                       XOR     AX,AX
                                    ;MOV    [SS:IOCALL_REQSTAT],AX
000029F2 A3[7F03]                   mov     [IOCALL_REQSTAT],ax ; 0

000029F5 06                         PUSH    ES
000029F6 1F                         POP     DS
000029F7 89FE                       MOV     SI,DI                 ; DS:SI -> driver
000029F9 16                         PUSH    SS
000029FA 07                         POP     ES

                                ;hkn; IOCALL is in DOSDATA (msconst.asm)
                                    ; 30/01/2024
                                    ; (ds <> ss, ss = DOSDATA segment)
000029FB BB[7C03]                   MOV     BX,IOCALL             ; ES:BX -> Call header
000029FE 1E                         push    ds
000029FF 56                         push    si
00002A00 E89228                     call    DEVIOCALL2
00002A03 5E                         pop     si
00002A04 1F                         pop     ds

                                ;hkn; SS override
00002A05 36A1[7F03]                 MOV     AX,[SS:IOCALL_REQSTAT]; Get Status word
                                    ;AND    AX,STBUI ; 200h               ; Mask to busy bit
                                    ; 29/01/2024
00002A09 80E402                     and     ah,STBUI>>8 ; 2
00002A0C B109                       MOV     CL,9
00002A0E D3E8                       SHR     AX,CL                 ; Busy bit to bit 0
                                ioctl_da_ok_j:     ; 11/11/2022
00002A10 E95BDC                     jmp     SYS_RET_OK

                                    ; 29/01/2024
                                ;--------------------------------------------------------------------------
                                ;
                                ; IOCTL: AL = B
                                ;
                                ; ENTRY: DS = DOSDATA
                                ;
                                ;--------------------------------------------------------------------------

                                Set_Retry_Parameters:
                                    ; 09/09/2018
00002A13 890E[1C00]                 MOV     [RetryLoop],CX        ; 0 retry loop count allowed
00002A17 09D2                       OR      DX,DX                 ; zero retries not allowed
00002A19 7485                       JZ      short ioctl_bad_fun
                                    ; 29/01/2024
                                    ;jnz    short set_new_retry_cnt
                                    ;jmp    ioctl_bad_fun
                                ;set_new_retry_cnt:
00002A1B 8916[1A00]                 MOV     [RetryCount],DX               ; Set new retry count
                                doneok:
                                    ;jmp    SYS_RET_OK            ; Done
                                    ; 11/11/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
                                    ;jmp    short ioctl_status_ret
                                    ; 16/12/2022
                                    ;jmp    short ioctl_ok  ; jmp SYS_RET_OK
                                    ; 29/01/2024
00002A1F EBEF                       jmp     short ioctl_da_ok_j

                                ;--------------------------------------------------------------------------
                                ;
                                ; IOCTL: AL = 9
                                ;
                                ; ENTRY: DS = DOSDATA
                                ;
                                ;--------------------------------------------------------------------------

                                    ; 30/01/2024 - Retro DOS v5.0

                                ioctl_drive_attr:
                                    ; MSDOS 3.3 (& MSDOS 6.0)
00002A21 88D8                       mov     al,bl
00002A23 E86751                     call    GETTHISDRV
00002A26 7243                       jc      short ioctl_drv_err
00002A28 E8BA00                     call    Get_Driver_BL
                                    ; MSDOS 6.0
00002A2B 723E                       JC      short ioctl_drv_err   ; drive not valid

                                    ; 03/07/2024
                                    ; jnz = net drive (current dir is net)
                                    ; jz = local drive

                                    ; 03/07/2024
                                    ; 30/01/2024 - Retro DOS v5.0
                                    ; 30/01/2024 - PCDOS 7.1 IBMDOS.COM
                                    ;;;
00002A2D BA4209                     mov     dx,942h               ; 0942h -> Attribute word
                                                                  ; bit 11 - open/close/remmedia calls supported
                                                                  ; bit 8 - (new type driver)
                                                                  ; bit 6 - Generic IOCTL call supported
                                                                  ; bit 1 - driver supports 32-bit sector addressing
00002A30 7504                       jnz     short ioctl_drive_attr2 ; NET device
                                                  ; 30/01/2024
```

```
                                                ; NOTE: 'jnz' condition is correct for Windows ME
                                                ; 'Get_Driver_BL ' because it tests bit 0 of [drive_flags]
                                                ; but in PCDOS 7.1 'Get_Driver_BL', this flag
                                                ;       (remote or removable? disk flag) is not tested
                                                ; the last test is net device test) ; (*!)
                                        ;;;;

                                        ;mov    dx,[es:di+4]
00002A32 268B5504                       mov     dx,[es:di+SYSDEV.ATT]
                                                ; get device attribute word
                                        ; 26/06/2024
                                    ioctl_drive_attr2: ; 30/01/2024
00002A36 88C3                           MOV     BL,AL           ; Phys letter to BL (A=0)

                                    ;hkn; SS override
                                        ; 30/01/2024
                                        ; (MSDOS 6.22 IO.SYS - DOSCODE:62B8h)
                                        ; (PCDOS 7.1 IBMDOS.COM - DOSCODE:69DBh)
                                        ;LES    DI,[SS:THISCDS]         ; NOTE: PCDOS 7.1 has bug here,
                                                ; ds must be same with ss here...
                                                ; because there is 'les di, [ds:THISCDS]' in
                                                ; Get_Driver_BL
                                                ; and a second 'test byte ptr es:[di+44h],80h'
                                                ; is not necessary; also its result (jnz)
                                                ; overwrites DS. /// Erdogan Tan - 30/01/2024
                                        ; 30/01/2024
                                        ; Retro DOS v5.0
                                        ; (Windows ME IO.SYS - BIOSCODE:67ABh)
00002A38 C43E[A205]                     les     di,[THISCDS]

                                        ;test   word [es:di+43h],8000h
                                        ;TEST   word [ES:DI+curdir.flags],curdir_isnet
                                        ;test   byte [es:di+44h],80h
00002A3C 26F6454480                     TEST    byte [ES:DI+curdir.flags+1],(curdir_isnet>>8) ; (*!)
00002A41 7403                           JZ      short IOCTLShare

                                        ;or     dx,1000h ; (MSDOS 3.3)

                                    ;   Net devices don't return a device attribute word.
                                    ;   Bit 12 = 1, meaning net device, all others = 0.

00002A43 BA0010                         MOV     DX,1000h ; MSDOS 6.0

                                    IOCTLShare:
                                        ; 30/01/2024
                                        ; ds = ss = DOSDATA segment
                                        ;push   ss
                                        ;pop    ds

00002A46 BE[BE03]                       MOV     SI,OPENBUF
00002A49 80C341                         ADD     BL,"A" ; 41h
00002A4C 881C                           MOV     [SI],BL
00002A4E C744013A00                     MOV     WORD [SI+1],003AH ; ":",0
00002A53 B80003                         MOV     AX,0300h
00002A56 F8                             CLC
                                        ;INT    int_IBM
00002A57 CD2A                           int      2Ah    ; Microsoft Networks - CHECK DIRECT I/O
                                                        ; DS:SI -> ASCIZ disk device name
                                                        ; (may be full path or only drive
                                                        ; specifier--must include the colon)
                                                        ; Return: CF clear if absolute disk access allowed
00002A59 7303                           JNC     short IOCTLLocal      ; Not shared
                                        ;OR     DX,0200H              ; Shared, bit 9
                                        ; 17/12/2022
00002A5B 80CE02                         or      dh,02h
                                    IOCTLLocal:
                                        ;test   word [es:di+43h],1000h
                                        ;TEST   word [ES:DI+curdir.flags],curdir_local
                                        ;test   byte [es:di+44h],10h
00002A5E 26F6454410                     TEST    byte [ES:DI+curdir.flags+1],(curdir_local>>8)
                                        ;JZ     short ioctl_set_DX
                                        ; 16/12/2022
00002A63 7403                           jz      short _ioctl_set_DX
                                        ;OR     DX,8000h
                                        ; 17/12/2022
00002A65 80CE80                         or      dh,80h
                                    ;ioctl_set_DX:
                                    _ioctl_set_DX:
                                        ; 16/12/2022
00002A68 E972FE                         jmp     ioctl_set_dx
                                    ; 16/12/2022
                                    %if 0
                                        call    Get_User_Stack
                                        MOV     [SI+user_env.user_DX],DX
                                        ;;jmp   SYS_RET_OK
                                        ;; 25/06/2019
                                        ;jmp    SYS_RET_OK_clc
                                        ; 11/11/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
                                    ioctl_gd_ok_j:
                                        jmp     short ioctl_da_ok_j
                                    %endif

                                    ioctl_drv_err:
00002A6B B00F                           mov     al,error_invalid_drive ; 0Fh
                                    ioctl_gd_err_j:     ; 11/11/2022
00002A6D E908DC                         jmp     SYS_RET_ERR

                                    ;--------------------------------------------------------------------------
                                    ;
                                    ; IOCTL: AL = A
                                    ;
                                    ; ENTRY: DS = DOSDATA
                                    ;
                                    ;--------------------------------------------------------------------------

                                    ioctl_handle_redir:
00002A70 E8604C                         call    SFFromHandle          ; ES:DI -> SFT
00002A73 7303                           JNC     short ioctl_got_sft   ; have valid handle
00002A75 E929FE                         jmp     ioctl_bad_handle ; 10/08/2018

                                    ioctl_got_sft:
                                        ;mov    dx,[es:di+5]
00002A78 268B5505                       MOV     DX,[ES:DI+SF_ENTRY.sf_flags] ; Get flags
                                        ;JMP    short ioctl_set_DX    ; pass dx to user and return
                                        ; 16/12/2022
00002A7C EBEA                           jmp     short _ioctl_set_DX

                                        ; 16/12/2022
                                    ;ioctl_bad_funj:
                                        ;JMP    ioctl_bad_fun

                                    ;--------------------------------------------------------------------------
                                    ;
                                    ; IOCTL: AL= 4,5
```

```
                                ;
                                ; ENTRY: DS = DOSDATA
                                ;    SI = user's DS
                                ;
                                ;
                                ; BUGBUG: Don't push anything on the stack between ioctl_get_dev: and
                                ;         the call to Check_If_Net because Check_If_Net gets our
                                ;         return address off the stack if the drive is invalid.
                                ;
                                ;--------------------------------------------------------------------------

                                    ; 30/01/2024 - Retro DOS v5.0

                                ioctl_get_dev:
00002A7E E88F00                     CALL    Check_If_Net
                                    ;JNZ    short ioctl_bad_funj  ; There are no "net devices", and they
                                                                  ; certainly don't know how to do this
                                                                  ; call.
                                    ; 16/12/2022
00002A81 7403                       jz      short ioctl_do_string
                                ioctl_bad_funj:
00002A83 E91AFF                     JMP     ioctl_bad_fun

                                ioctl_do_string:
                                    ;test   word [es:di+4],4000h
                                    ;TEST   word [ES:DI+SYSDEV.ATT],DEVIOCTL; See if device accepts control
                                    ;test   byte [es:di+5],40h
00002A86 26F6450540                 TEST    byte [ES:DI+SYSDEV.ATT+1],(DEVIOCTL>>8)
00002A8B 74F6                       JZ      short ioctl_bad_funj          ; NO
                                                                  ; assume IOCTL read
00002A8D C606[7E03]03               MOV     byte [IOCALL_REQFUNC],DEVRDIOCTL ; 3

00002A92 A801                       TEST    AL,1                  ; is it func. 4/5 or 2/3
00002A94 7405                       JZ      short ioctl_control_call ; it is read. goto ioctl_control_call

                                                                  ; it is an IOCTL write
00002A96 C606[7E03]0C               MOV     byte [IOCALL_REQFUNC],DEVWRIOCTL ; 12

                                ioctl_control_call:
                                    ; 30/01/2024
                                    ;MOV    AL,DRDWRHL ; 22       ; MSDOS 6.22 MSDOS.SYS, Windows ME IO.SYS
                                    ; 26/06/2024
00002A9B B014                       mov     al,20 ; PCDOS 7.1 IBMDOS.COM
                                ioctl_setup_pkt:
00002A9D 88DC                       MOV     AH,BL                 ; Unit number
00002A9F A3[7C03]                   MOV     [IOCALL_REQLEN],AX
00002AA2 31C0                       XOR     AX,AX
00002AA4 A3[7F03]                   MOV     [IOCALL_REQSTAT],AX ; 0
00002AA7 A2[8903]                   MOV     [IOMED],AL
00002AAA 890E[8E03]                 MOV     [IOSCNT],CX
00002AAE 8916[8A03]                 MOV     [IOXAD],DX
00002AB2 8936[8C03]                 MOV     [IOXAD+2],SI
00002AB6 06                         PUSH    ES
00002AB7 1F                         POP     DS
00002AB8 89FE                       MOV     SI,DI                 ; DS:SI -> driver
00002ABA 16                         PUSH    SS
00002ABB 07                         POP     ES

00002ABC BB[7C03]                   MOV     BX,IOCALL             ; ES:BX -> Call header
                                ioctl_do_IO:
00002ABF E8D327                     call    DEVIOCALL2

                                ;hkn; SS override for IOCALL
                                    ;test   word [SS:IOCALL_REQSTAT],8000h
                                    ;TEST   word [SS:IOCALL_REQSTAT],STERR ;Error?
                                    ;test   byte [SS:IOCALL_REQSTAT+1],80h
00002AC2 36F606[8003]80             TEST    byte [SS:IOCALL_REQSTAT+1],(STERR>>8)
00002AC8 7507                       JNZ     short ioctl_string_err

                                ;hkn; SS override
00002ACA 36A1[8E03]                 MOV     AX,[SS:IOSCNT]        ; Get actual bytes transferred
                                    ; 16/12/2022
00002ACE E99DDB                     jmp     SYS_RET_OK
                                    ; 11/11/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
                                    ;jmp    short ioctl_gd_ok_j

                                ioctl_string_err:
00002AD1 368B3E[7F03]               MOV     DI,[SS:IOCALL_REQSTAT];Get Error
                                device_err:
00002AD6 81E7FF00                   AND     DI,STECODE ; 00FFh    ; mask out irrelevant bits
00002ADA 89F8                       MOV     AX,DI
00002ADC E8B237                     call    SET_I24_EXTENDED_ERROR

                                ;hkn; use SS override
                                ;hkn;       mov     ax,[CS:EXTERR]
00002ADF 36A1[2403]                 mov     ax,[SS:EXTERR]
                                    ;jmp    SYS_RET_ERR
                                    ; 11/11/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
00002AE3 EB88                       jmp     short ioctl_gd_err_j

                                ; 17/05/2019 - Retro DOS v4.0

                                ;--------------------------------------------------------------------------
                                ; Proc name : Get_Driver_BL
                                ;
                                ;   DS is DOSDATA
                                ;   BL is drive number (0=default)
                                ;   Returns pointer to device in ES:DI, unit number in BL if carry clear
                                ;   No regs modified
                                ;
                                ;--------------------------------------------------------------------------

                                    ; 30/01/2024 - Retro DOS v5.0

                                Get_Driver_BL:
00002AE5 50                         PUSH    AX
00002AE6 88D8                       MOV     AL,BL                 ; Drive
00002AE8 E8A250                     call    GETTHISDRV
00002AEB 7221                       jc      short ioctl_bad_drv
00002AED 30DB                       XOR     BL,BL                 ; Unit zero on Net device
00002AEF C606[2303]03               MOV     byte [EXTERR_LOCUS],errLOC_Net ; 3
00002AF4 C43E[A205]                 LES     DI,[THISCDS]
                                    ;test   word [es:di+43h],8000h
                                    ;TEST   word [ES:DI+curdir.flags],curdir_isnet
                                    ;test   byte [es:di+44h],80h
00002AF8 26F6454480                 TEST    byte [ES:DI+curdir.flags+1],(curdir_isnet>>8)
                                    ;les    di,[es:di+45h]
00002AFD 26C47D45                   LES     DI,[ES:DI+curdir.devptr] ; ES:DI -> Dpb or net dev
00002B01 750B                       JNZ     short got_dev_ptr       ; Is net
                                    ;;;
                                    ; 30/01/2024 - PCDOS 7.1 IBMDOS.COM
                                    ;MOV    byte [EXTERR_LOCUS],errLOC_Disk ; 2
00002B03 E8E0E7                     call    set_exerr_locus_disk
                                    ;;;
```

```
18586                                   ;mov    bl,[es:di+1]
18587 00002B06 268A5D01                 MOV     BL,[ES:DI+DPB.UNIT]   ; Unit number
18588                                   ;les    di,[es:di+13h]
18589 00002B0A 26C47D13                 LES     DI,[ES:DI+DPB.DRIVER_ADDR] ; Driver addr
18590                                 got_dev_ptr:
18591                                   ; 30/01/2024
18592                                   ; cf=0
18593                                   ;CLC
18594                                 ioctl_bad_drv:
18595 00002B0E 58                       POP     AX
18596 00002B0F C3                       retn
18597
18598                                 ;--------------------------------------------------------------------------
18599                                 ; Proc Name : Check_If_Net:
18600                                 ;
18601                                 ;
18602                                 ; Checks if the device is over the net or not. Returns result in ZERO flag.
18603                                 ; If no device is found, the return address is popped off the stack, and a
18604                                 ; jump is made to ioctl_drv_err.
18605                                 ;
18606                                 ; On Entry:
18607                                 ; Registers same as those for Get_Driver_BL
18608                                 ;
18609                                 ; On Exit:
18610                                 ; ZERO flag - set if not a net device
18611                                 ;           - reset if net device
18612                                 ; ES:DI -> the device
18613                                 ;
18614                                 ;
18615                                 ; BUGBUG: This function assumes the following stack setup on entry
18616                                 ;
18617                                 ;     SP+2 -> Error return address
18618                                 ;     SP   -> Normal return address
18619                                 ;
18620                                 ;--------------------------------------------------------------------------
18621
18622                                   ; 30/01/2024 - Retro DOS v5.0
18623                                   ; MSDOS 6.22 MSDOS.SYS - DOSCODE:639Ch
18624                                   ; PCDOS 7.1 IBMDOS.COM - DOSCODE:6A91h
18625                                   ; Windows ME IO.SYS - BIOSCODE:68E1h
18626
18627                                 Check_If_Net:
18628                                   ; MSDOS 3.3 (& MSDOS 6.0)
18629 00002B10 E8D2FF                   CALL    Get_Driver_BL
18630 00002B13 7201                     JC      short ioctl_drv_err_pop       ; invalid drive letter
18631
18632                                 ; 30/01/2024 ('Get_Driver_BL' returns with
18633                                 ;         'curdir_isnet' condition/ZF, no need to a second test)
18634                                 %if 0
18635                                   ;;;
18636                                   ; (PCDOS 7.1 IBMDOS.COM, Windows ME IO.SYS)
18637                                   PUSH    ES
18638                                   PUSH    DI
18639                                   LES     DI,[THISCDS]
18640                                   ;test   word [es:di+43h],8000h
18641                                   ;TEST   word [ES:DI+curdir.flags],curdir_isnet
18642                                   ;test   byte [es:di+44h],80h
18643                                   TEST    byte [ES:DI+curdir.flags+1],(curdir_isnet>>8)
18644                                   POP     DI
18645                                   POP     ES
18646                                   ;;;
18647                                 %endif
18648 00002B15 C3                       retn
18649
18650                                 ioctl_drv_err_pop:
18651 00002B16 58                       pop     ax                    ; pop off return address
18652 00002B17 E951FF                   jmp     ioctl_drv_err
18653
18654                                 ioctl_bad_funj3:
18655 00002B1A E983FE                   jmp     ioctl_bad_fun
18656
18657                                 ioctl_string_errj:
18658 00002B1D EBB2                     jmp     short ioctl_string_err  ; 25/05/2019
18659
18660                                 ;--------------------------------------------------------------------------
18661                                 ;
18662                                 ; IOCTL: AL = E, F
18663                                 ;
18664                                 ; ENTRY: DS = DOSDATA
18665                                 ;
18666                                 ;
18667                                 ; BUGBUG: Don't push anything on the stack between ioctl_drive_owner: and
18668                                 ;         the call to Check_If_Net because Check_If_Net gets our
18669                                 ;         return address off the stack if the drive is invalid.
18670                                 ;
18671                                 ;--------------------------------------------------------------------------
18672
18673                                 ioctl_drive_owner:
18674                                   ; MSDOS 3.3 (& MSDOS 6.0)
18675 00002B1F E8EEFF                   Call    Check_If_Net
18676 00002B22 75F6                     JNZ     short ioctl_bad_funj3 ; There are no "net devices", and they
18677                                                                 ;   certainly don't know how to do this
18678                                                                 ;   call.
18679                                   ;TEST   word [ES:DI+SYSDEV.ATT],DEV320       ; See if device can handle this
18680                                   ; 09/09/2018
18681                                   ;test   byte [es:di+4],40h
18682 00002B24 26F6450440               TEST    byte [ES:DI+SYSDEV.ATT],DEV320 ; 0040h
18683 00002B29 74EF                     JZ      short ioctl_bad_funj3 ; NO
18684                                   ;mov    byte [IOCALL_REQFUNC],23
18685 00002B2B C606[7E03]17             mov     byte [IOCALL_REQFUNC],DEVGETOWN      ; default to get owner
18686 00002B30 3C0E                     cmp     al,0Eh                ; Get Owner ?
18687 00002B32 7405                     jz      short GetOwner
18688                                 SetOwner:
18689 00002B34 C606[7E03]18             MOV     byte [IOCALL_REQFUNC],DEVSETOWN ; 24
18690                                 GetOwner:
18691 00002B39 B00D                     MOV     AL,OWNHL ; 13
18692 00002B3B 88DC                     MOV     AH,BL                 ; Unit number
18693 00002B3D A3[7C03]                 MOV     [IOCALL_REQLEN],AX
18694 00002B40 31C0                     XOR     AX,AX
18695 00002B42 A3[7F03]                 MOV     [IOCALL_REQSTAT],AX
18696 00002B45 06                       PUSH    ES
18697 00002B46 1F                       POP     DS
18698 00002B47 89FE                     MOV     SI,DI                 ; DS:SI -> driver
18699 00002B49 16                       PUSH    SS
18700 00002B4A 07                       POP     ES
18701 00002B4B BB[7C03]                 MOV     BX,IOCALL             ; ES:BX -> Call header
18702 00002B4E 1E                       push    ds
18703 00002B4F 56                       push    si
18704 00002B50 E84227                   call    DEVIOCALL2
18705 00002B53 5E                       pop     si
18706 00002B54 1F                       pop     ds
18707                                 ;hkn; SS override
18708                                   ;TEST   word [SS:IOCALL_REQSTAT],STERR ;Error?
18709                                   ;test   byte [SS:IOCALL_REQSTAT+1],80h
```

```
18710 00002B55 36F606[8003]80              TEST    byte [SS:IOCALL_REQSTAT+1],(STERR>>8)
18711 00002B5B 75C0                        jnz     short ioctl_string_errj
18712 00002B5D 36A0[7D03]                  MOV     AL,[SS:IOCALL_REQUNIT]; Get owner returned by device
18713                                                                     ; owner returned is 1-based.
18714 00002B61 E90ADB                      jmp     SYS_RET_OK
18715
18716                                   ;============================================================================
18717                                   ; DELETE.ASM, MSDOS 6.0, 1991
18718                                   ;============================================================================
18719                                   ; 07/08/2018 - Retro DOS v3.0
18720                                   ; 17/05/2019 - Retro DOS v4.0
18721
18722                                   ;   TITLE   DOS_DELETE - Internal DELETE call for MS-DOS
18723                                   ;   NAME    DOS_DELETE
18724
18725                                   ;
18726                                   ;   Microsoft Confidential
18727                                   ;   Copyright (C) Microsoft Corporation 1991
18728                                   ;   All Rights Reserved.
18729                                   ;
18730
18731                                   ;** DELETE.ASM - Low level routine for deleting files
18732                                   ;----------------------------------------------------------------------------
18733                                   ;           DOS_DELETE
18734                                   ;           REN_DEL_Check
18735                                   ;           FastOpen_Delete             ; DOS 3.3
18736                                   ;           FastOpen_Update             ; DOS 3.3
18737
18738                                   ;   Revision history:
18739                                   ;
18740                                   ;   A000  version 4.00    Jan. 1988
18741                                   ;   A001  Fastopen Rename fix     April 1989
18742
18743                                   ;Installed = TRUE
18744
18745                                   ;   i_need NoSetDir,BYTE
18746                                   ;   i_need Creating,BYTE
18747                                   ;   i_need DELALL,BYTE
18748                                   ;   i_need THISDPB,DWORD
18749                                   ;   i_need THISSFT,DWORD
18750                                   ;   i_need THISCDS,DWORD
18751                                   ;   i_need CURBUF,DWORD
18752                                   ;   i_need ATTRIB,BYTE
18753                                   ;   i_need SATTRIB,BYTE
18754                                   ;   i_need WFP_START,WORD
18755                                   ;   i_need REN_WFP,WORD                   ;BN001
18756                                   ;   i_need NAME1,BYTE                     ;BN001
18757                                   ;   i_need FoundDel,BYTE
18758                                   ;   i_need AUXSTACK,BYTE
18759                                   ;   i_need VOLCHNG_FLAG,BYTE
18760                                   ;   i_need JShare,DWORD
18761                                   ;   i_need FastOpenTable,BYTE             ; DOS 3.3
18762                                   ;   i_need FastTable,BYTE                 ; DOS 4.00
18763                                   ;
18764                                   ;   i_need Del_ExtCluster,WORD            ; DOS 4.00
18765                                   ;
18766                                   ;   i_need SAVE_BX,WORD                   ; DOS 4.00
18767                                   ;   i_need DMAADD,DWORD
18768                                   ;   i_need RENAMEDMA,BYTE
18769
18770                                   ;----------------------------------------------------------------------------
18771                                   ;
18772                                   ; Procedure Name : DOS_DELETE
18773                                   ;
18774                                   ; Inputs:
18775                                   ;   [WFP_START] Points to WFP string ("d:/" must be first 3 chars, NUL
18776                                   ;           terminated)
18777                                   ;   [CURR_DIR_END] Points to end of Current dir part of string
18778                                   ;           ( = -1 if current dir not involved, else
18779                                   ;            Points to first char after last "/" of current dir part)
18780                                   ;   [THISCDS] Points to CDS being used
18781                                   ;           (Low word = -1 if NUL CDS (Net direct request))
18782                                   ;   [SATTRIB] Is attribute of search, determines what files can be found
18783                                   ; Function:
18784                                   ;   Delete the specified file(s)
18785                                   ; Outputs:
18786                                   ;   CARRY CLEAR
18787                                   ;           OK
18788                                   ;   CARRY SET
18789                                   ;       AX is error code
18790                                   ;           error_file_not_found
18791                                   ;                   Last element of path not found
18792                                   ;           error_path_not_found
18793                                   ;                   Bad path (not in curr dir part if present)
18794                                   ;           error_bad_curr_dir
18795                                   ;                   Bad path in current directory part of path
18796                                   ;           error_access_denied
18797                                   ;                   Attempt to delete device or directory
18798                                   ;           ***error_sharing_violation***
18799                                   ;                   Deny both access required, generates an INT 24.
18800                                   ;                   This error is NOT returned. The INT 24H is generated,
18801                                   ;                     and the file is ignored (not deleted). Delete will
18802                                   ;                     simply continue on looking for more files.
18803                                   ;                     Carry will NOT be set in this case.
18804                                   ; DS preserved, others destroyed
18805                                   ;
18806                                   ;----------------------------------------------------------------------------
18807
18808                                   FILEFOUND   equ 01h
18809                                   FILEDELETED equ 10h
18810
18811                                       ; 12/11/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
18812                                       ; DOSCODE:63E9h (MSDOS 5.0, MSDOS.SYS)
18813
18814                                       ; 30/01/2024 - Retro DOS v5.0 (Modified MSDOS 5.0 IBMDOS.COM)
18815                                       ; DOSCODE:6B11h (PCDOS 7.1, IBMDOS.COM)
18816
18817                                   DOS_DELETE:
18818
18819                                   ;hkn; DOS_Delete is called from file.asm and fcbio.asm. DS has been set up
18820                                   ;hkn; appropriately at this point.
18821
18822 00002B64 E8C2EC                      call    TestNet
18823 00002B67 7306                        JNC     short LOCAL_DELETE
18824
18825                                   ;IF NOT Installed
18826                                   ;   transfer NET_DELETE
18827                                   ;ELSE
18828                                       ;MOV    AX,(MultNET SHL 8) | 19
18829                                       ;INT    2FH
18830                                       ;return
18831
18832 00002B69 B81311                      mov     ax,1113h
18833 00002B6C CD2F                        int      2Fh    ; Multiplex - NETWORK REDIRECTOR - DELETE REMOTE FILE
```

```
                                                        ; SS = DS = DOS CS, SDA first filename pointer ->
                                                        ;              fully-qualified filename in DOS CS
                                                        ; SDA CDS pointer -> current directory structure for drive with file
                                                        ; Return: CF set on error
00002B6E C3                             retn
                                        ;ENDIF

                                        LOCAL_DELETE:
00002B6F C606[6F05]00                       MOV     byte [FOUNDDEL],0     ; No files found and no files deleted
00002B74 E86BED                             call    ECritDisk
                                            ;mov    word [CREATING],0E500h
00002B77 C706[7E05]00E5                     MOV     WORD [CREATING],DIRFREE*256+0 ; Assume not del *.*
00002B7D 8B36[B205]                         MOV     SI,[WFP_START]
                                        SKPNUL:
00002B81 AC                                 LODSB
00002B82 08C0                               OR      AL,AL
00002B84 75FB                               JNZ     short SKPNUL          ; go to end
00002B86 83EE04                             SUB     SI,4                  ; Back over possible "*.*"
00002B89 813C2A2E                           CMP     WORD [SI],2E2Ah ; "*."
00002B8D 7506                               JNZ     short TEST_QUEST
00002B8F 807C022A                           CMP     BYTE [SI+2],"*"
00002B93 741F                               JZ      short CHECK_ATTS
                                        TEST_QUEST:
00002B95 83EE09                             SUB     SI,9          ; Back over possible "????????.???"
00002B98 87FE                               XCHG    DI,SI

00002B9A 16                                 push    ss
                                            ;pop    ds ; ! Retro DOS v3.0 BUG !
00002B9B 07                                 pop     es ; 17/05/2019

00002B9C B83F3F                             MOV     AX,"??" ; 3F3Fh
00002B9F B90400                             MOV     CX,4          ; four sets of "??"
00002BA2 F3AF                               REPE    SCASW
00002BA4 751C                               JNZ     short NOT_ALL
00002BA6 87FE                               XCHG    DI,SI
00002BA8 AD                                 LODSW
00002BA9 3D2E3F                             CMP     AX,3F2Eh ; ".?"
00002BAC 7514                               JNZ     short NOT_ALL
00002BAE AD                                 LODSW
00002BAF 3D3F3F                             CMP     AX,"??"
00002BB2 750E                               JNZ     short NOT_ALL
                                        CHECK_ATTS:
00002BB4 A0[6D05]                           MOV     AL,[SATTRIB]
                                            ;and    al,1Fh
00002BB7 241F                               AND     AL,attr_hidden+attr_system+attr_directory+attr_volume_id+attr_read_only
                                                                          ; Look only at hidden bits
                                            ;cmp    al,1Fh
00002BB9 3C1F                               CMP     AL,attr_hidden+attr_system+attr_directory+attr_volume_id+attr_read_only
                                                                          ; All must be set
00002BBB 7505                               JNZ     short NOT_ALL

                                        ; NOTE WARNING DANGER-----
                                        ;    This DELALL stuff is not safe. It allows directories to be deleted.
                                        ;   It should ONLY be used by FORMAT in the ROOT directory.

00002BBD C606[7F05]00                       MOV     byte [DELALL],0             ; DEL *.* - flag deleting all
                                        NOT_ALL:
00002BC2 C606[4C03]01                       MOV     byte [NoSetDir],1
00002BC7 E84F1F                             call    GetPathNoSet
00002BCA 7312                               JNC     short Del_found
00002BCC 750B                               JNZ     short _bad_path
00002BCE 08C9                               OR      CL,CL
00002BD0 7407                               JZ      short _bad_path
                                        No_file:
00002BD2 B80200                             MOV     AX,error_file_not_found
                                        ErrorReturn:
00002BD5 F9                                 STC
                                            ;call   LCritDisk
                                            ;retn
                                            ; 18/12/2022
00002BD6 E936ED                             jmp     LCritDisk

                                        _bad_path:
00002BD9 B80300                             MOV     AX,error_path_not_found
00002BDC EBF7                               JMP     short ErrorReturn

                                        Del_found:
00002BDE 750C                               JNZ     short NOT_DIR         ; Check for dir specified
00002BE0 803E[7F05]00                       CMP     byte [DELALL],0             ; DelAll = 0 allows delete of dir.
00002BE5 7405                               JZ      short NOT_DIR
                                        Del_access_err:
00002BE7 B80500                             MOV     AX,error_access_denied
00002BEA EBE9                               JMP     short ErrorReturn

                                        NOT_DIR:
00002BEC 08E4                               OR      AH,AH                 ; Check if device name
00002BEE 78F7                               JS      short Del_access_err  ; Can't delete I/O devices

                                        ; Main delete loop. CURBUF+2:BX points to a matching directory entry.

                                        DELFILE:
00002BF0 800E[6F05]01                       OR      byte [FOUNDDEL],FILEFOUND ; file found, not deleted yet

                                        ; If we are deleting the Volume ID, then we set VOLUME_CHNG flag to make
                                        ; DOS issue a build BPB call the next time this drive is accessed.

00002BF5 1E                                 PUSH    DS
00002BF6 8A26[7F05]                         MOV     AH,[DELALL]
00002BFA C53E[E205]                         LDS     DI,[CURBUF]

                                        ;hkn; SS override
00002BFE 36F606[6B05]01                     TEST    byte [SS:ATTRIB],attr_read_only ; are we deleting RO files too?
00002C04 7509                               JNZ     short DoDelete        ; yes

00002C06 F6470B01                           TEST    byte [BX+dir_entry.dir_attr],attr_read_only
00002C0A 7403                               JZ      short DoDelete        ; not read only

                                            ; 30/01/2024 (PCDOS 7.1 IBMDOS.COM)
                                        Skip_it:
00002C0C 1F                                 POP     DS
00002C0D EB57                               JMP     SHORT DELNXT          ; Skip it (Note ES:BP not set)

                                        DoDelete:
00002C0F E8AF00                             call    REN_DEL_Check         ; Sets ES:BP = [THISDPB]
                                            ;JNC    short DEL_SHARE_OK
                                            ;POP    DS
                                            ;JMP    SHORT DELNXT          ; Skip it
                                            ; 30/01/2024
00002C12 72F8                               jc      short Skip_it

                                        DEL_SHARE_OK:
                                            ; 17/05/2019 - Retro DOS v4.0
                                            ; MSDOS 6.0
                                            ;test   byte [di+5],40h
```

```
18958 00002C14 F6450540                 TEST    byte [DI+BUFFINFO.buf_flags],buf_dirty
18959                                                   ;LB. if already dirty           ;AN000;
18960 00002C18 7507                     JNZ     short yesdirty          ;LB.  don't increment dirty count ;AN000;
18961 00002C1A E8AD3F                   call    INC_DIRTY_COUNT                 ;LB.                                    ;AN000;
18962                                   ;or     byte [di+5],40h
18963 00002C1D 804D0540                 OR      byte [DI+BUFFINFO.buf_flags],buf_dirty
18964                                 yesdirty:
18965 00002C21 8827                     mov     [bx],ah
18966                                   ;MOV    [BX+dir_entry.dir_name],AH ; Put in E5H or 0
18967                                   ;;;
18968                                   ; 30/01/2024 - Retro DOS v5.0
18969                                   ; (PCDOS 7.1 IBMDOS.COM)
18970 00002C23 31DB                     xor     bx,bx ; 0
18971                                   ;cmp    [es:bp+0Fh],bx
18972 00002C25 26395E0F                 cmp     [es:bp+DPB.FAT_SIZE],bx ; 0
18973 00002C29 7503                     jnz      short yesdirty_fc_1   ;  not FAT32
18974 00002C2B 8B5CFA                   mov      bx,[si-6]             ;  high word of the first cluster (FAT32)
18975                                 yesdirty_fc_1:
18976                                   ;mov    [ss:CLUSTNUM_HW],bx
18977 00002C2E 89DA                     mov     dx,bx ; * ; 30/01/2024 - Retro DOS v5.0
18978                                   ;;;
18979
18980 00002C30 8B1C                     MOV     BX,[SI]                 ; Get firclus pointer
18981 00002C32 1F                       POP     DS
18982                                   ;;;
18983 00002C33 8916[E80A]               mov     [CLUSTNUM_HW],dx ; * ; 30/01/2024 - Retro DOS v5.0
18984                                   ;;;
18985 00002C37 800E[6F05]10             OR      byte [FOUNDDEL],FILEDELETED ; 10h ; Deleted file
18986
18987                                   ; 30/01/2024
18988                                   ;CMP    BX,2
18989                                   ;JB     short DELNXT            ; File has invalid FIRCLUS (too small)
18990                                   ;;;
18991                                   ; 30/01/2024 - Retro DOS v5.0
18992                                   ; (PCDOS 7.1 IBMDOS.COM)
18993                                   ;cmp    word [CLUSTNUM_HW],0
18994 00002C3C 21D2                     and     dx,dx ; 30/01/2024 - Retro DOS v5.0
18995 00002C3E 7505                     jnz     short yesdirty_fc_2
18996 00002C40 83FB02                   cmp     bx,2
18997 00002C43 7221                     jb      short DELNXT            ; File has invalid FIRCLUS (too small)
18998                                 yesdirty_fc_2:
18999                                   ;cmp    word [es:bp+0Fh],0
19000 00002C45 26837E0F00               cmp     word [es:bp+DPB.FAT_SIZE],0
19001 00002C4A 750C                     jnz      short yesdirty_fc_3   ; not FAT32
19002                                   ;push   bx
19003                                   ;mov    bx,[CLUSTNUM_HW]
19004                                   ;;cmp   bx,[es:bp+2Fh]
19005                                   ;cmp    bx,[es:bp+DPB.LAST_CLUSTER+2]
19006                                   ;30/01/2024 - Retro DOS v5.0
19007                                   ;mov    dx,[CLUSTNUM_HW]
19008                                   ; dx = [CLUSTNUM_HW] ; *
19009 00002C4C 263B562F                 cmp     dx,[es:bp+DPB.LAST_CLUSTER+2]
19010                                   ;pop    bx
19011 00002C50 750A                     jne     short yesdirty_fc_4
19012                                   ;cmp    bx,[es:bp+2Dh]
19013 00002C52 263B5E2D                 cmp     bx,[es:bp+DPB.LAST_CLUSTER]
19014 00002C56 EB04                     jmp      short yesdirty_fc_4
19015                                 yesdirty_fc_3:
19016                                   ;;;
19017                                   ;cmp    bx,[es:bp+0Dh]
19018 00002C58 263B5E0D                 CMP     BX,[ES:BP+DPB.MAX_CLUSTER]
19019                                 yesdirty_fc_4:     ; 30/01/2024
19020 00002C5C 7708                     JA      short DELNXT            ; File has invalid FIRCLUS (too big)
19021
19022 00002C5E E86F2F                   call    RELEASE                 ; Free file data
19023 00002C61 7258                     JC      short No_fileJ
19024
19025                                 ; DOS 3.3  FastOpen
19026
19027 00002C63 E8D900                   CALL    FastOpen_Delete         ; delete the dir info in fastopen
19028
19029                                 ; DOS 3.3  FastOpen
19030
19031                                 DELNXT:
19032 00002C66 C42E[8A05]               LES     BP,[THISDPB]            ; Possible to get here without this set
19033 00002C6A E8321C                   call    GETENTRY                ; Registers need to be reset
19034 00002C6D 724C                     JC      short No_fileJ
19035 00002C6F E8291B                   call    NEXTENT
19036                                   ;JNC    short DELFILE
19037                                   ; 30/01/2024
19038 00002C72 7203                     jc      short DELNXT2
19039 00002C74 E979FF                   jmp     DELFILE
19040                                 DELNXT2:
19041 00002C77 C42E[8A05]               LES     BP,[THISDPB]            ; NEXTENT sets ES=DOSGROUP
19042
19043                                   ;;;
19044                                   ; 30/01/2024 - Retro DOS v5.0
19045                                   ; (PCDOS 7.1 IBMDOS.COM)
19046                                   ;
19047 00002C7B E8EE07                   call    update_fat32_fsinfo
19048                                   ;;;
19049
19050                                   ; 12/11/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
19051                                   ;MOV    AL,[ES:BP+DPB.DRIVE]
19052                                   ;;mov   al,[es:bp+0]
19053                                   ; 15/12/2022
19054 00002C7E 268A4600                 MOV     AL,[ES:BP]
19055 00002C82 E8133E                   call    FLUSHBUF
19056 00002C85 7234                     JC      short No_fileJ
19057                                 ;
19058                                 ; Now we need to test FoundDel for our flags. The cases to consider are:
19059                                 ;
19060                                 ;   not found not deleted        file not found
19061                                 ;   not found        deleted             *** impossible ***
19062                                 ;   found not deleted            access denied (read-only)
19063                                 ;   found    deleted             no error
19064                                 ;
19065 00002C87 F606[6F05]10             TEST    byte [FOUNDDEL],FILEDELETED ; did we delete a file?
19066 00002C8C 7426                     JZ      short DelError          ; no, figure out what's wrong.
19067
19068                                 ; We set VOLCHNG_FLAG to indicate that we have changed the volume label
19069                                 ; and to force the DOS to issue a media check.
19070
19071 00002C8E F606[6B05]08             TEST    byte [ATTRIB],attr_volume_id ; 8
19072 00002C93 741C                     jz      short No_Set_Flag
19073 00002C95 50                       PUSH    AX
19074 00002C96 06                       PUSH    ES
19075 00002C97 57                       PUSH    DI
19076 00002C98 C43E[A205]               LES     DI,[THISCDS]
19077 00002C9C 268A25                   MOV     AH,[ES:DI]              ; Get drive
19078 00002C9F 80EC41                   SUB     AH,'A'                    ; Convert to 0-based
19079 00002CA2 8826[A10A]               mov     [VOLCHNG_FLAG],AH
19080
19081                                   ; MSDOS 6.0
```

```
00002CA6 30FF                       XOR     BH,BH                   ;>32mb delete volume id from boot record ;AN000;
00002CA8 E8EB04                     call    Set_Media_ID            ;>32mb set volume id to boot record   ;AN000;

00002CAB E8AD38                     call    FATREAD_CDS             ; force media check
00002CAE 5F                         POP     DI
00002CAF 07                         POP     ES
00002CB0 58                         POP     AX
                                No_Set_Flag:
                                    ;call   LCritDisk               ; carry is clear
                                    ;retn
                                    ; 18/12/2022
00002CB1 E95BEC                     jmp     LCritDisk
                                DelError:
00002CB4 F606[6F05]01               TEST    byte [FOUNDDEL],FILEFOUND ; not deleted. Did we find file?
00002CB9 7503                       JNZ     short Del_access_errJ ; yes. Access denied
                                No_fileJ:
00002CBB E914FF                     JMP     No_file ; 10/08/2018          ; Nope
                                Del_access_errJ:
00002CBE E926FF                     JMP     Del_access_err ; 10/08/2018

                                ; 08/08/2018 - Retro DOS v3.0

                                ;Break     <REN_DEL_Check - check for access for rename and delete>
                                ;-----------------------------------------------------------------------------
                                ; Procedure Name : REN_DEL_Check
                                ;
                                ; Inputs:
                                ;   [THISDPB] set
                                ;   [CURBUF+2]:BX points to entry
                                ;   [CURBUF+2]:SI points to firclus field of entry
                                ;   [WFP_Start] points to name
                                ; Function:
                                ;   Check for Exclusive access on given file.
                                ;     Used by RENAME, SET_FILE_INFO, and DELETE.
                                ; Outputs:
                                ;   ES:BP = [THISDPB]
                                ;   NOTE: The WFP string pointed to by [WFP_Start] Will be Modified. The
                                ;           last element will be loaded from the directory entry. This is
                                ;           so the name given to the sharer doesn't have any meta chars in
                                ;           it.
                                ;   Carry set if sharing violation, INT 24H generated
                                ;       NOTE THAT AX IS NOT error_sharing_violation.
                                ;           This is because input AX is preserved.
                                ;           Caller must set the error if needed.
                                ;   Carry clear
                                ;           OK
                                ; AX,DS,BX,SI,DI preserved
                                ;-----------------------------------------------------------------------------

                                    ; 31/01/2024 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMDOS.COM)

                                REN_DEL_Check:

00002CC1 1E                         PUSH    DS
00002CC2 57                         PUSH    DI
00002CC3 50                         PUSH    AX
00002CC4 53                         PUSH    BX
00002CC5 56                         PUSH    SI              ; Save CURBUF pointers

00002CC6 16                         push    ss
00002CC7 07                         pop     es

                                ;hkn; context ES will assume ES to DOSDATA
                                ;hkn; ASSUME       ES:DOSGROUP

                                ;hkn; SS override
00002CC8 368B3E[B205]               MOV     DI,[SS:WFP_START] ; ES:DI -> WFP
00002CCD 89DE                       MOV     SI,BX

                                ;hkn; SS override
00002CCF 368E1E[E405]               MOV     DS,[SS:CURBUF+2] ; DS:SI -> entry (FCB style name)
00002CD4 89FB                       MOV     BX,DI           ; Set backup limit for skipback
                                    ;ADD    BX,2            ; Skip over d: to point to leading '\'
                                    ; 31/01/2024
00002CD6 43                         inc     bx
00002CD7 43                         inc     bx
00002CD8 E8EEEA                     call    StrLen          ; CX is length of ES:DI including NUL
00002CDB 49                         DEC     CX              ; Don't include nul in count
00002CDC 01CF                       ADD     DI,CX           ; Point to NUL at end of string
00002CDE E85E51                     call    SkipBack        ; Back up one element
00002CE1 47                         INC     DI              ; Point to start of last element

                                    ; 17/05/2019 - Retro DOS v4.0
                                ;hkn; SS override
                                    ; MSDOS 6.0
00002CE2 36893E[0106]               MOV     [SS:SAVE_BX],DI       ;IFS. save for DOS_RENAME   ;AN000;
                                    ;
00002CE7 E8BDF9                     call    PackName        ; Transfer name from entry to ASCIZ tail.
00002CEA 5E                         POP     SI              ; Get back entry pointers
00002CEB 5B                         POP     BX
00002CEC 53                         PUSH    BX
00002CED 56                         PUSH    SI              ; Back on stack

00002CEE 16                         push    ss
00002CEF 1F                         pop     ds

                                ;hkn; context DS will assume ES to DOSDATA
                                ;hkn; ASSUME       DS:DOSGROUP

                                    ; 07/07/2024
                                    ;push   es               ; (not necessary) ; 31/01/2024
                                    ;push   di

                                ; Close the file if possible by us.
                                ;
                                ;if installed
00002CF0 FF1E[C400]                 Call    far [JShare+(13*4)] ; 13 = ShCloseFile
                                ;else
                                ;   Call    ShCloseFile
                                ;endif
                                    ;;;
                                    ; 31/01/2024 - Retro DOS v5.0
                                    ; PCDOS 7.1 IBMDOS.COM
00002CF4 803E[0303]FF               cmp     byte [fShare],0FFh
00002CF9 750A                       jne     short rdc_1
00002CFB 8C06[A005]                 mov     [THISSFT+2],es
00002CFF 893E[9E05]                 mov     [THISSFT],di
00002D03 EB0A                       jmp     short rdc_2
                                rdc_1:
                                    ;;;
00002D05 8C1E[A005]                 MOV     [THISSFT+2],DS

                                ;hkn; AUXSTACK is in DOSDATA
00002D09 C706[9E05][6507]           MOV     word [THISSFT],AUXSTACK-SF_ENTRY.size  ; RENAMEDMA+(384-59)
```

```
                                                        ; Scratch space
                                rdc_2:              ; 30/01/2024
                                    ; 07/07/2024
                                    ;pop    di
                                    ;pop    es
                                    ;
00002D0F 30E4                       XOR     AH,AH           ; Indicate file to DOOPEN (high bit off)
00002D11 E8EF28                     call    DOOPEN          ; Fill in SFT for share check
00002D14 C43E[9E05]                 LES     DI,[THISSFT]
                                    ;mov    word [es:di+2],10h
00002D18 26C745021000               MOV     word [ES:DI+SF_ENTRY.sf_mode],SHARING_DENY_BOTH ; 10h
                                                        ; requires exclusive access
                                    ;MOV    word [ES:DI+SF_ENTRY.sf_ref_count],1 ; Pretend open
00002D1E 26C7050100                 mov     word [ES:DI],1
00002D23 E86357                     call    ShareEnter
00002D26 720D                       jc      short CheckDone
00002D28 C43E[9E05]                 LES     DI,[THISSFT]
                                    ;MOV    word [ES:DI+SF_ENTRY.sf_ref_count],0
00002D2C 26C7050000                 mov     word [ES:DI],0 ; Pretend closed and free

00002D31 E85057                     call    ShareEnd        ; Tell sharer we're done with THISSFT
00002D34 F8                         CLC
                                CheckDone:
00002D35 C42E[8A05]                 LES     BP,[THISDPB]
00002D39 5E                         POP     SI
00002D3A 5B                         POP     BX
00002D3B 58                         POP     AX
00002D3C 5F                         POP     DI
00002D3D 1F                         POP     DS
00002D3E C3                         retn

                                ;Break      <FastOpen_Delete - delete dir info in fastopen>
                                ;--------------------------------------------------------------------------
                                ; Procedure Name : FastOpen_Delete
                                ; Inputs:
                                ;   None
                                ; Function:
                                ;   Call FastOpen to delete the dir info.
                                ; Outputs:
                                ;   None
                                ;--------------------------------------------------------------------------

                                    ; 31/01/2024 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMDOS.COM)

                                FastOpen_Delete:
00002D3F 9C                         PUSHF                   ; save flag
00002D40 56                         PUSH    SI              ; save registers
00002D41 57                         push    di ; 31/01/2024 (PCDOS 7.1 IBMDOS.COM)
00002D42 53                         PUSH    BX
00002D43 50                         PUSH    AX
                                    ;mov    si,[WFP_START] ; MSDOS 3.3
                                ;hkn; SS override
                                    ; 17/05/2019 - Retro DOS v4.0
                                    ; MSDOS 6.0
00002D44 368B36[B205]               MOV     SI,[ss:WFP_START] ; ds:si points to path name

00002D49 B003                       MOV     AL,FONC_delete ; al = 3

                                ; 31/01/2024 (PCDOS 7.1 IBMDOS.COM)
                                %if 0
                                fastinvoke:
                                ;hkn; FastTable is in DOSDATA
                                    MOV     BX,FastTable+2
                                    CALL    far [BX]        ; call fastopen
                                    POP     AX              ; restore registers
                                    POP     BX
                                    ;pop    di ; 31/01/2024 (PCDOS 7.1 IBMDOS.COM)
                                    POP     SI
                                    POPF                    ; restore flag
                                    retn
                                %else
00002D4B EB0F                       jmp     short fastinvoke ; 31/01/2024
                                %endif

                                    ; 13/11/2022 Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
                                    ; DOSCODE:65A0h (MSDOS 5.0 MSDOS.SYS)

                                    ; 31/01/2024 Retro DOS v5.0 (Modified PCDOS 7.1 IBMDOS.COM)
                                    ; DOSCODE:65B4h (MSDOS 6.22 MSDOS.SYS)
                                    ; DOSCODE:6D07h (PCDOS 7.1 IBMDOS.COM)

                                ;Break      <FastOpen_Rename - Rename directory>    ; PTR 5622
                                ;--------------------------------------------------------------------------
                                ; PROCEDURE Name : FastOpen_Rename
                                ;
                                ; Inputs:
                                ;    REN_WFP   = Path Name
                                ;    NAME1     = New Name
                                ; Function:
                                ;   Call FastOpen to rename the dir entry in the cache
                                ; Outputs:
                                ;   None
                                ;--------------------------------------------------------------------------

                                FastOpen_Rename:
                                    ; 17/05/2019 - Retro DOS v4.0
                                    ; 08/08/2018 - Retro DOS v3.0
                                    ; MSDOS 6.0
00002D4D 9C                         PUSHF                   ;AN001 save flag
00002D4E 56                         PUSH    SI              ;AN001 save registers
00002D4F 57                         PUSH    DI              ;AN001
00002D50 53                         PUSH    BX              ;AN001
00002D51 50                         PUSH    AX              ;AN001
                                    ;
                                ;hkn; SS override
00002D52 368B36[B405]               MOV     SI,[SS:REN_WFP]       ;AN001 ;;AN001  ds:si-->Path name addrs

                                ;hkn; NAME1 is in DOSDATA
00002D57 BF[4B05]                   MOV     DI,NAME1        ;;AN001  ds:di-->New name addrs
                                    ;mov    al,6
00002D5A B006                       MOV     AL,FONC_Rename ;;AN001  al = 6

                                fastinvoke: ; 31/01/2024 (PCDOS 7.1 IBMDOS.COM)

                                ;hkn; FastTable is in DOSDATA
00002D5C BB[3E11]                   MOV     BX,FastTable+2
00002D5F FF1F                       CALL    far [BX]        ;;AN001  call fastopen

00002D61 58                         POP     AX              ; restore registers  ;AN001
00002D62 5B                         POP     BX                                   ;AN001
00002D63 5F                         POP     DI                                   ;AN001
00002D64 5E                         POP     SI                                   ;AN001
00002D65 9D                         POPF                    ; restore flag       ;AN001
00002D66 C3                         retn                                         ;AN001
```

```
                                ;Break      <FastOpen_Update - update dir info in fastopen>
                                ;-----------------------------------------------------------------------------
                                ; Procedure Name : FastOpen_Update
                                ;
                                ; Inputs:
                                ;   DL      drive number (A=0,B=1,,,)
                                ;   CX      first cluster #
                                ;   AH      0 updates dir entry
                                ;           1 updates CLUSNUM , BP = new CLUSNUM
                                ;   ES:DI   directory entry
                                ; Function:
                                ;   Call FastOpen to update the dir info.
                                ; Outputs:
                                ;   None
                                ;-----------------------------------------------------------------------------

                                FastOpen_Update:
00002D67 9C                         PUSHF                   ; save flag
00002D68 56                         PUSH    SI
00002D69 57                         push    di ; 31/01/2024 (PCDOS 7.1 IBMDOS.COM)
00002D6A 53                         PUSH    BX              ; save regs
00002D6B 50                         PUSH    AX
00002D6C B004                       MOV     AL,FONC_update ; al = 4
00002D6E EBEC                       JMP     short fastinvoke

                                    ; 17/05/2019

                                    ; MSDOS 6.0
                                ;entry Fast_Dispatch              ; future fastxxxx entry       ;AN000;
                                ;-----------------------------------------------------------------------------
                                Fast_Dispatch:
                                ;hkn; FastTable is in DOSDATA
00002D70 BE[3E11]                   MOV     SI,FastTable+2 ; index to the      ;AN000;
                                ;hkn; use SS override
00002D73 36FF1C                     CALL    far [SS:SI]     ; RMFD call fastopen
00002D76 C3                         retn

                                ;============================================================================
                                ; RENAME.ASM, MSDOS 6.0, 1991
                                ;============================================================================
                                ; 08/08/2018 - Retro DOS v3.0
                                ; 17/05/2019 - Retro DOS v4.0

                                ;   TITLE   DOS_RENAME - Internal RENAME call for MS-DOS
                                ;   NAME    DOS_RENAME

                                ;** Low level routine for renaming files
                                ;-----------------------------------------------------------------------------
                                ;   DOS_RENAME
                                ;
                                ;   Modification history:
                                ;
                                ;       Created: ARR 30 March 1983

                                ;-----------------------------------------------------------------------------
                                ;
                                ; Procedure Name : DOS_RENAME
                                ;
                                ; Inputs:
                                ;   [WFP_START] Points to SOURCE WFP string ("d:/" must be first 3
                                ;           chars, NUL terminated)
                                ;   [CURR_DIR_END] Points to end of Current dir part of string [SOURCE]
                                ;           ( = -1 if current dir not involved, else
                                ;            Points to first char after last "/" of current dir part)
                                ;   [REN_WFP] Points to DEST WFP string ("d:/" must be first 3
                                ;           chars, NUL terminated)
                                ;   [THISCDS] Points to CDS being used
                                ;           (Low word = -1 if NUL CDS (Net direct request))
                                ;   [SATTRIB] Is attribute of search, determines what files can be found
                                ; Function:
                                ;   Rename the specified file(s)
                                ;   NOTE: This routine uses most of AUXSTACK as a temp buffer.
                                ; Outputs:
                                ;   CARRY CLEAR
                                ;       OK
                                ;   CARRY SET
                                ;       AX is error code
                                ;           error_file_not_found
                                ;                   No match for source, or dest path invalid
                                ;           error_not_same_device
                                ;                   Source and dest are on different devices
                                ;           error_access_denied
                                ;                   Directory specified (not simple rename),
                                ;                   Device name given, Destination exists.
                                ;                   NOTE: In third case some renames may have
                                ;                    been done if metas.
                                ;           error_path_not_found
                                ;                   Bad path (not in curr dir part if present)
                                ;                   SOURCE ONLY
                                ;           error_bad_curr_dir
                                ;                   Bad path in current directory part of path
                                ;                   SOURCE ONLY
                                ;           error_sharing_violation
                                ;                   Deny both access required, generates an INT 24.
                                ; DS preserved, others destroyed
                                ;
                                ;-----------------------------------------------------------------------------

                                    ; 14/11/2022 Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
                                    ; 31/01/2024 Retro DOS v5.0 (Modified PCDOS 7.1 IBMDOS.COM)

                                DOS_RENAME:

                                ;hkn; DOS_RENAME is called from file.asm and fcbio.asm. DS has been set up
                                ;hkn; at this point to DOSDATA.

00002D77 E8AFEA                     call    TestNet
00002D7A 7306                       JNC     short LOCAL_RENAME

                                ;IF NOT Installed
                                ;   transfer NET_RENAME
                                ;ELSE
                                    ;MOV    AX,(MultNET SHL 8) OR 17
                                    ;INT    2FH
                                    ;return

00002D7C B81111                     mov     ax, 1111h
00002D7F CD2F                       int     2Fh     ; Multiplex - NETWORK REDIRECTOR - RENAME REMOTE FILE
                                                    ; SS = DS = DOS CS,
                                                    ; SDA first filename pointer = offset of fully-qualified old name
                                                    ; SDA CDS pointer -> current directory
                                                    ; Return: CF set on error
00002D81 C3                         retn
```

```
                                ;ENDIF
                                
                                LOCAL_RENAME:
00002D82 C606[2303]02               MOV     byte [EXTERR_LOCUS],errLOC_Disk ; 2
00002D87 8B36[B205]                 MOV     SI,[WFP_START]
00002D8B 8B3E[B405]                 MOV     DI,[REN_WFP]
00002D8F 8A04                       MOV     AL,[SI]
00002D91 8A25                       MOV     AH,[DI]
00002D93 0D2020                     OR      AX,2020H                ; Lower case
00002D96 38E0                       CMP     AL,AH
00002D98 7405                       JZ      short SAMEDRV
00002D9A B81100                     MOV     AX,error_not_same_device ; 11h
00002D9D F9                         STC
00002D9E C3                         retn
                                
                                SAMEDRV:
00002D9F FF36[2E03]                 PUSH    WORD [DMAADD+2]
00002DA3 FF36[2C03]                 PUSH    WORD [DMAADD]
00002DA7 8C1E[2E03]                 MOV     [DMAADD+2],DS
                                
                                ;hkn; RENAMEDMA is in DOSDATA
00002DAB C706[2C03][2006]           MOV     WORD [DMAADD],RENAMEDMA
00002DB1 C606[7005]00               MOV     byte [FOUND_DEV],0      ; Rename fails on DEVS, assume not a dev
00002DB6 E829EB                     call    ECritDisk
00002DB9 E86807                     call    DOS_SEARCH_FIRST        ; Sets [NoSetDir] to 1, [CURBUF+2]:BX
                                                                    ;    points to entry
00002DBC 7314                       JNC     short Check_Dev
00002DBE 83F812                     CMP     AX,error_no_more_files ; 12h
00002DC1 7503                       JNZ     short GOTERR
00002DC3 B80200                     MOV     AX,error_file_not_found ; 2
                                GOTERR:
00002DC6 F9                         STC
                                RENAME_POP:
00002DC7 8F06[2C03]                 POP     WORD [DMAADD]
00002DCB 8F06[2E03]                 POP     WORD [DMAADD+2]
                                    ;call   LCritDisk
                                    ;retn
                                    ; 16/12/2022
00002DCF E93DEB                     jmp     LCritDisk
                                
                                Check_Dev:
                                    ; 17/05/2019 - Retro DOS v4.0
                                    ;mov    ax,5
00002DD2 B80500                     MOV     AX,error_access_denied; Assume error
                                
                                    ; MSDOS 6.0
00002DD5 1E                         PUSH    DS                      ;PTM.                           ;AN000;
00002DD6 C536[2C03]                 LDS     SI,[DMAADD]             ;PTM.  check if source a dir  ;AN000;
                                    ;add    si,21
00002DDA 83C615                     ADD     SI,find_buf.attr        ;PTM.                           ;AN000;
                                    ;test   byte [si+11],10h
00002DDD F6440B10                   TEST    byte [SI+dir_entry.dir_attr],attr_directory ;PTM.       ;AN000;
00002DE1 7407                       JZ      short notdir            ;PTM.                           ;AN000;
00002DE3 8B36[B405]                 MOV     SI,[REN_WFP]            ;PTM.  if yes, make sure path ;AN000;
00002DE7 E86AFA                     call    Check_PathLen2          ;PTM.   length < 67           ;AN000;
                                notdir:
00002DEA 1F                         POP     DS                      ;PTM.                           ;AN000;
00002DEB 77D9                       JA      short GOTERR            ;PTM.                           ;AN000;
                                
                                    ; MSDOS 3.3 & MSDOS 6.0
00002DED 803E[7005]00               CMP     byte [FOUND_DEV],0
00002DF2 75D2                       JNZ     short GOTERR
                                
                                ; At this point a source has been found. There is search continuation info (a
                                ; la DOS_SEARCH_NEXT) for the source at RENAMEDMA, together with the first
                                ; directory entry found.
                                ; [THISCDS], [THISDPB], and [THISDRV] are set and will remain correct
                                ; throughout the RENAME since it is known at this point that the source and
                                ; destination are both on the same device.
                                ; [SATTRIB] is also set.
                                
00002DF4 89DE                       MOV     SI,BX
                                    ;add    si,26
00002DF6 83C61A                     ADD     SI,dir_entry.dir_first
00002DF9 E8C5FE                     call    REN_DEL_Check
00002DFC 7305                       JNC     short REN_OK1
00002DFE B82000                     MOV     AX,error_sharing_violation  ; 20h
00002E01 EBC4                       JMP     short RENAME_POP
                                
                                ;--------------------------------------------------------------------------
                                ; Check if the source is a file or directory. If file, delete the entry
                                ; from the Fastopen cache. If directory, rename it later
                                ;--------------------------------------------------------------------------
                                
                                REN_OK1:                            ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
                                    ; MSDOS 6.0
                                    ; 31/01/2024 (PCDOS 7.1 IBMDOS.COM)
                                    ;PUSH   SI
00002E03 C536[2C03]                 LDS     SI,[DMAADD]             ;BN00X; PTM. check if source a dir ;AN000;
                                    ;add    si,21
00002E07 83C615                     ADD     SI,find_buf.attr        ;;BN00XPTM.P5520                ;AN000;
                                    ;test   byte [si+11],10h
00002E0A F6440B10                   TEST    byte [SI+dir_entry.dir_attr],attr_directory ;;BN00XPTM. ;AN000;
                                    ;JZ     short NOT_DIR1          ;;BN00XPTM.                     ;AN000;
00002E0E 7503                       jnz     short SWAP_SOURCE ; 31/01/2024
                                    ;POP    SI                      ;BN00X
                                    ;JMP    SHORT SWAP_SOURCE       ;BN00X
                                ;NOT_DIR1:                          ;;BN00X it is a file, delete the entry
                                    ;POP    SI
                                
                                    ; MSDOS 3.3 (& MSDOS 6.0)
00002E10 E82CFF                     call    FastOpen_Delete         ; delete dir info in fastopen DOS 3.3
                                SWAP_SOURCE:
                                    ; MSDOS 3.3
                                    ;MOV    SI,[REN_WFP]
                                    ;MOV    [WFP_START],SI
                                    ; MSDOS 6.0
00002E13 A1[B205]                   MOV     AX,[WFP_START]          ; Swap source and destination
00002E16 8B36[B405]                 MOV     SI,[REN_WFP]            ; Swap source and destination
00002E1A 8936[B205]                 MOV     [WFP_START],SI          ; WFP_START = Destination path
00002E1E A3[B405]                   MOV     [REN_WFP],AX            ; REN_WFP   = Source path
                                    ; MSDOS 3.3 (& MSDOS 6.0)
00002E21 C706[B605]FFFF             MOV     word [CURR_DIR_END],-1; No current dir on dest
                                    ;mov    word [CREATING],0E5FFh
00002E27 C706[7E05]FFE5             MOV     WORD [CREATING],DIRFREE*256+0FFh  ; Creating, not DEL *.*
                                                                    ; A rename is like a CREATE_NEW as far
                                                                    ; as the destination is concerned.
00002E2D E8E91C                     call    GetPathNoSet
                                
                                ;   If this GETPATH fails due to file not found, we know all renames will work
                                ;   since no files match the destination name. If it fails for any other
                                ;   reason, the rename fails on a path not found, or whatever (also fails if
                                ;   we find a device or directory). If the GETPATH succeeds, we aren't sure
                                ;   if the rename should fail because we haven't built an explicit name by
```

```
19578                                   ;   substituting for the meta chars in it. In this case the destination file
19579                                   ;   spec with metas is in [NAME1] and the explicit source name is at RENAMEDMA
19580                                   ;   in the directory entry part.
19581
19582 00002E30 722A                        JC      short NODEST
19583
19584                                       ; MSDOS 6.0
19585                                       ;JZ     short BAD_ACC         ; Dest string is a directory  ;AC000;
19586                                       ; !! MSDOS 3.3 !!
19587                                       ;JZ     short BAD_ACC ; !!    ; Dest string is a directory
19588
19589 00002E32 08E4                        OR      AH,AH                 ; Device?
19590 00002E34 7933                        JNS     short SAVEDEST        ; No, continue
19591                                   BAD_ACC:
19592 00002E36 B80500                      MOV     AX,error_access_denied
19593 00002E39 F9                          STC
19594                                   RENAME_CLEAN:
19595 00002E3A 9C                          PUSHF                         ; Save carry state
19596 00002E3B 50                          PUSH    AX                    ; and error code (if carry set)
19597                                       ;;;
19598                                       ; 31/01/2024 - Retro DOS v5.0
19599                                       ; (PCDOS 7.1 IBMDOS.COM)
19600 00002E3C C42E[8A05]                  les     bp,[THISDPB]
19601 00002E40 E82906                      call    update_fat32_fsinfo
19602                                       ;;;
19603 00002E43 A0[7605]                    MOV     AL,[THISDRV]
19604 00002E46 E84F3C                      call    FLUSHBUF
19605 00002E49 58                          POP     AX
19606 00002E4A 803E[4A03]00                CMP     byte [FAILERR],0
19607 00002E4F 7504                        JNZ     short BAD_ERR         ; User FAILed to I 24
19608 00002E51 9D                          POPF
19609 00002E52 E972FF                      JMP     RENAME_POP
19610
19611                                   BAD_ERR:
19612 00002E55 58                          POP     AX                    ; Saved flags
19613                                       ; 16/12/202
19614                                       ; 14/11/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
19615                                   BAD_PATH: ; *
19616 00002E56 B80300                      MOV     AX,error_path_not_found
19617 00002E59 E96AFF                      JMP     GOTERR
19618
19619                                   NODEST:
19620 00002E5C 75F8                        JNZ     short BAD_PATH
19621 00002E5E 803E[4A03]00                CMP     byte [FAILERR],0
19622 00002E63 75F1                        JNZ     short BAD_PATH        ; Search for dest failed
19623                                                                     ; because user FAILed on I 24
19624                                       ; 14/11/2022
19625 00002E65 08C9                        OR      CL,CL
19626                                       ;JNZ    short SAVEDEST
19627                                       ; 17/05/2019
19628 00002E67 74ED                        jz      short BAD_PATH ; *
19629                                   ;BAD_PATH: ; *
19630                                   ;   MOV     AX,error_path_not_found
19631                                   ;   ;STC
19632                                   ;   ;JMP    RENAME_POP
19633                                   ;   ; 17/05/2019
19634                                   ;   jmp     GOTERR
19635
19636                                   ; 16/12/2022
19637                                   %if 0
19638                                       ; 14/11/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
19639                                       or      cl,cl
19640                                       jnz     short SAVEDEST
19641                                       ;jz     short BAD_PATH ; *
19642                                   BAD_PATH: ; *
19643                                       ;mov    ax,3
19644                                       mov     ax,error_path_not_found
19645                                       stc
19646                                       jmp     RENAME_POP
19647                                   %endif
19648
19649                                   SAVEDEST:
19650 00002E69 16                          push    ss
19651 00002E6A 07                          pop     es
19652
19653                                   ;hkn; NAME1 & NAME2 is in DOSDATA
19654 00002E6B BF[5705]                    MOV     DI,NAME2
19655 00002E6E BE[4B05]                    MOV     SI,NAME1
19656
19657 00002E71 B90B00                      MOV     CX,11
19658 00002E74 F3A4                        REP     MOVSB                 ; Save dest with metas at NAME2
19659                                       ;;;
19660                                       ; 31/01/2024
19661                                       ; (PCDOS 7.1 IBMDOS.COM)
19662 00002E76 A1[DC0A]                    mov     ax,[DIRSTART_HW]
19663 00002E79 A3[E60A]                    mov     [DESTSTART_HW],ax
19664                                       ;;;
19665 00002E7C A1[C205]                    MOV     AX,[DIRSTART]
19666 00002E7F A3[6405]                    MOV     [DESTSTART],AX
19667                                   BUILDDEST:
19668                                       ; 31/01/2024
19669                                       ;push   ss
19670                                       ;pop    es                    ; needed due to JMP BUILDDEST below
19671
19672                                   ;hkn; RENAMEDMA, NAME1, NAME2 in DOSDATA
19673 00002E82 BB[3506]                    MOV     BX,RENAMEDMA+21               ; Source of replace chars
19674 00002E85 BF[4B05]                    MOV     DI,NAME1              ; Real dest name goes here
19675 00002E88 BE[5705]                    MOV     SI,NAME2              ; Raw dest
19676
19677 00002E8B B90B00                      MOV     CX,11
19678
19679                                       ; 17/05/2019 - Retro DOS v4.0
19680
19681                                       ; MSDOS 6.0
19682 00002E8E E82601                      CALL    NEW_RENAME            ;IFS. replace ? chars ;AN000;
19683
19684                                       ; MSDOS 3.3
19685
19686                                   ; 08/08/2018 - Retro DOS v3.0
19687                                   ; MSDOS 6.0
19688                                   ;--------------------------------------------------------------------------
19689                                   ;Procedure: NEW_RENAME
19690                                   ;
19691                                   ;Input: DS:SI -> raw string with ?
19692                                   ;   ES:DI -> destination string
19693                                   ;   DS:BX -> source string
19694                                   ;Function: replace ? chars of raw string with chars in source string and
19695                                   ;      put in destination string
19696                                   ;Output: ES:DI-> new string
19697                                   ;--------------------------------------------------------------------------
19698                                   ;
19699                                   ;NEW_RENAME:
19700                                   ;NEWNAM:
19701                                   ;   ; IBMDOS.COM (MSDOS 3.3, 1987) - Offset 341Ah
```

```
19702                                   ;   LODSB
19703                                   ;   CMP     AL,"?"
19704                                   ;   JNZ     short NOCHG
19705                                   ;   MOV     AL,[BX]                 ; Get replace char
19706                                   ;NOCHG:
19707                                   ;   STOSB
19708                                   ;   INC     BX                      ; Next replace char
19709                                   ;   LOOP    NEWNAM
19710                                   ;   ; MSDOS 6.0
19711                                   ;   ;retn
19712
19713                                       ; MSDOS 3.3 & MSDOS 6.0
19714                                       ;mov    byte [ATTRIB],16h
19715 00002E91 C606[6B05]16               MOV     byte [ATTRIB],attr_all; Stop duplicates with any attributes
19716 00002E96 C606[7E05]FF               MOV     byte [CREATING],0FFH
19717 00002E9B E8361F                     call    DEVNAME                 ; Check if we built a device name
19718 00002E9E 7396                       JNC     short BAD_ACC
19719                                     ;;;
19720                                     ; 31/01/2024
19721                                     ; (PCDOS 7.1 IBMDOS.COM)
19722 00002EA0 8B1E[E60A]                 mov     bx,[DESTSTART_HW]
19723 00002EA4 891E[EE0A]                 mov     [ROOTCLUST_HW],bx
19724                                     ;;;
19725 00002EA8 8B1E[6405]                 MOV     BX,[DESTSTART]
19726 00002EAC C42E[8A05]                 LES     BP,[THISDPB]
19727 00002EB0 E8AC1A                     call    SETDIRSRCH              ; Reset search to start of dir
19728 00002EB3 7281                       JC      short BAD_ACC           ; Screw up
19729 00002EB5 E88918                     call    FINDENTRY               ; See if new name already exists
19730                                     ;JNC    short BAD_ACC           ; Error if found
19731                                     ; 31/01/2024
19732 00002EB8 7346                       jnc     short BAD_ACCJ
19733 00002EBA 803E[4A03]00               CMP     byte [FAILERR],0
19734 00002EBF 753F                       JNZ     short BAD_ACCJ          ; Find failed because user FAILed to I 24
19735 00002EC1 A1[6405]                   MOV     AX,[DESTSTART]          ; DIRSTART of dest
19736                                     ;;;
19737                                     ; 31/01/2024
19738 00002EC4 8B16[E60A]                 mov     dx,[DESTSTART_HW]
19739 00002EC8 C42E[8A05]                 les     bp,[THISDPB]
19740 00002ECC 26837E0F00                 cmp     word [es:bp+DPB.FAT_SIZE],0
19741                                     ;cmp    word [es:bp+0Fh],0
19742 00002ED1 7506                       jnz     short builddst_1        ; not FAT32
19743 00002ED3 3B16[3106]                 cmp     dx,[RENAMEDMA+17]       ; DIRSTART_HW of source
19744 00002ED7 7506                       jne     short builddst_2
19745                                   builddst_1:
19746                                     ;;;
19747 00002ED9 3B06[2F06]                 CMP     AX,[RENAMEDMA+15]       ; DIRSTART of source
19748 00002EDD 7451                       JZ      short SIMPLE_RENAME     ; If =, just give new name
19749                                   builddst_2: ; 31/01/2024
19750                                     ;mov    al,[RENAMEDMA+32]
19751 00002EDF A0[4006]                   MOV     AL,[RENAMEDMA+21+dir_entry.dir_attr]
19752 00002EE2 A810                       TEST    AL,attr_directory ; 10h
19753 00002EE4 751A                       JNZ     short BAD_ACCJ          ; Can only do a simple rename on dirs,
19754                                                                       ; otherwise the . and .. entries get
19755                                                                       ; wiped.
19756 00002EE6 A2[6B05]                   MOV     [ATTRIB],AL
19757 00002EE9 8C1E[A005]                 MOV     [THISSFT+2],DS
19758
19759                                   ;hkn; AUXSTACK is in DOSDATA
19760                                     ;mov    si,RENAMEDMA+145h
19761 00002EED BE[6507]                   MOV     SI,AUXSTACK-SF_ENTRY.size ; RENAMEDMA+325
19762 00002EF0 8936[9E05]                 MOV     [THISSFT],SI
19763                                     ;mov    word [SI+2],2
19764 00002EF4 C744020200                 MOV     word [SI+SF_ENTRY.sf_mode],SHARING_COMPAT+open_for_both
19765 00002EF9 31C9                       XOR     CX,CX                   ; Set "device ID" for call into makenode
19766 00002EFB E88125                     call    RENAME_MAKE             ; This is in mknode
19767 00002EFE 7303                       JNC     short GOT_DEST
19768                                   BAD_ACCJ:
19769 00002F00 E933FF                     JMP     BAD_ACC
19770
19771                                   GOT_DEST:
19772 00002F03 53                         push    bx
19773 00002F04 C43E[9E05]                 LES     DI,[THISSFT]            ; RENAME_MAKE entered this into sharing
19774 00002F08 E87955                     call    ShareEnd                ; we need to remove it.
19775 00002F0B 5B                         pop     bx
19776
19777                                   ; A zero length entry with the correct new name has now been made at
19778                                   ;   [CURBUF+2]:BX.
19779
19780 00002F0C C43E[E205]                  LES     DI,[CURBUF]
19781
19782                                   ; 31/01/2024 - Retro DOS v5.0
19783                                   %if 0
19784                                       ; MSDOS 6.0
19785                                       ;test   byte [es:di+5],40h
19786                                       TEST    byte [ES:DI+BUFFINFO.buf_flags],buf_dirty
19787                                                                       ;LB. if already dirty           ;AN000;
19788                                       JNZ     short yesdirty1             ;LB.  don't increment dirty count ;AN000;
19789                                       call    INC_DIRTY_COUNT         ;LB.                            ;AN000;
19790                                       ;or     byte [es:di+5],40h
19791                                       OR      byte [ES:DI+BUFFINFO.buf_flags],buf_dirty
19792                                   yesdirty1:
19793                                   %else
19794                                       ; 31/01/2024 (PCDOS 7.1 IBMDOS.COM)
19795 00002F10 E8AB3C                     call    SET_BUF_DIRTY
19796                                   %endif
19797 00002F13 89DF                        MOV     DI,BX
19798                                       ;add    di,11
19799 00002F15 83C70B                     ADD     DI,dir_entry.dir_attr ; Skip name
19800
19801                                   ;hkn; RENAMEDMA is in DOSDATA
19802                                       ;mov    si,RENAMEDMA+32 ; 05/07/2024
19803 00002F18 BE[4006]                   MOV     SI,RENAMEDMA+21+dir_entry.dir_attr
19804                                       ;mov    cx,21
19805 00002F1B B91500                     MOV     CX,dir_entry.size-dir_entry.dir_attr
19806 00002F1E F3A4                       REP     MOVSB
19807 00002F20 E86E00                     CALL    GET_SOURCE
19808 00002F23 7269                       JC      short RENAME_OVER
19809 00002F25 89DF                       MOV     DI,BX
19810 00002F27 8E06[E405]                 MOV     ES,[CURBUF+2]
19811 00002F2B B0E5                       MOV     AL,DIRFREE ; 0E5h
19812 00002F2D AA                         STOSB                           ; "free" the source
19813 00002F2E EB13                       JMP     SHORT DIRTY_IT
19814
19815                                   SIMPLE_RENAME:
19816 00002F30 E85E00                     CALL    GET_SOURCE              ; Get the source back
19817 00002F33 7259                       JC      short RENAME_OVER
19818 00002F35 89DF                       MOV     DI,BX
19819 00002F37 8E06[E405]                 MOV     ES,[CURBUF+2]
19820
19821                                   ;hkn; NAME1 is in DOSDATA
19822 00002F3B BE[4B05]                   MOV     SI,NAME1                ; New Name
19823 00002F3E B90B00                     MOV     CX,11
19824 00002F41 F3A4                       REP     MOVSB
19825                                   DIRTY_IT:
```

```
19826 00002F43 8B3E[E205]                   MOV     DI,[CURBUF]
19827
19828                                   ; 31/01/2024 - Retro DOS v5.0
19829                                   %if 0
19830                                       ; MSDOS 6.0
19831                                       TEST    byte [ES:DI+BUFFINFO.buf_flags],buf_dirty
19832                                                               ;LB. if already dirty           ;AN000;
19833                                       JNZ     short yesdirty2             ;LB.  don't increment dirty count ;AN000;
19834                                       call    INC_DIRTY_COUNT     ;LB.                        ;AN000;
19835
19836                                       OR      byte [ES:DI+BUFFINFO.buf_flags],buf_dirty
19837                                   %else
19838                                       ; 31/01/2024 (PCDOS 7.1 IBMDOS.COM)
19839 00002F47 E8743C                     call    SET_BUF_DIRTY
19840                                   %endif
19841
19842                                   ;--------------------------------------------------------------------------------
19843                                   ; Check if the source is a directory of file. If directory rename it to the
19844                                   ; the new name in the Fastopen cache buffer. If file name it has been
19845                                   ; previously deleted.
19846                                   ;--------------------------------------------------------------------------------
19847
19848                                   ;yesdirty2: ; 31/01/2024
19849                                       ; MSDOS 6.0
19850 00002F4A 56                         PUSH    SI
19851 00002F4B C536[2C03]                 LDS     SI,[DMAADD]             ;;BN00XPTM. chek if source a dir ;AN000;
19852                                     ;add    si,21
19853 00002F4F 83C615                     ADD     SI,find_buf.attr        ;;BN00XPTM.P5520            ;AN000;
19854                                     ;test   byte [si+0Bh],10h
19855 00002F52 F6440B10                   TEST    byte [SI+dir_entry.dir_attr],attr_directory ;;BN00XPTM.    ;AN000;
19856 00002F56 7403                       JZ      short NOT_DIR2          ;;BN00XPTM.                 ;AN000;
19857 00002F58 E8F2FD                     call    FastOpen_Rename                 ;;BN00X rename dir entry in fastopen
19858                                     ; 31/01/2024
19859                                     ;POP    SI
19860                                     ;JMP    SHORT NOT_DIRTY1
19861                                   NOT_DIR2:                           ;;BN00X it is a file, delete the entry
19862 00002F5B 5E                         POP     SI
19863                                   NOT_DIRTY1:                         ;;BN00X
19864                                   NEXT_SOURCE:
19865                                   ;hkn; RENAMEDMA is in DOSDATA
19866 00002F5C BE[2106]                   MOV     SI,RENAMEDMA+1          ;Name
19867
19868                                   ; WARNING! Rename_Next leaves the disk critical section *ALWAYS*. We need
19869                                   ; to enter it before going to RENAME_Next.
19870
19871 00002F5F E880E9                     call    ECritDisk
19872 00002F62 C606[7E05]00               MOV     byte [CREATING],0 ; Correct setting for search (we changed it
19873                                                                 ;  to FF when we made the prev new file).
19874 00002F67 E8FF06                     call    RENAME_NEXT
19875
19876                                   ; Note, now, that we have exited the previous ENTER and so are back to where
19877                                   ; we were before.
19878
19879 00002F6A 7222                       JC      short RENAME_OVER
19880
19881                                     ;lea    si,[bx+26]
19882 00002F6C 8D771A                     LEA     SI,[BX+dir_entry.dir_first]
19883 00002F6F E84FFD                     call    REN_DEL_Check
19884 00002F72 7306                       JNC     short REN_OK2
19885 00002F74 B82000                     MOV     AX,error_sharing_violation ; 20h
19886                                   jmp_to_rename_clean: ; 28/12/2022
19887 00002F77 E9C0FE                     JMP     RENAME_CLEAN ; 10/08/2018
19888
19889                                   ;--------------------------------------------------------------------------------
19890                                   ; Check if file or directory. If file, delete file from the Fastopen cache,
19891                                   ; if directory, rename directory name in the Fastopen cache.
19892                                   ;--------------------------------------------------------------------------------
19893
19894                                   REN_OK2:
19895                                       ; MSDOS 6.0
19896                                       ;mov    al,[RENAMEDMA+32]
19897 00002F7A A0[4006]                   MOV     AL,[RENAMEDMA+21+dir_entry.dir_attr] ; PTR P5622
19898                                     ;test   al,10h
19899 00002F7D A810                       TEST    AL,attr_directory       ;;BN00X directory
19900 00002F7F 7408                       JZ      short Ren_Directory     ;;BN00X no - file, delete it
19901
19902                                     ; MSDOS 3.3 & MSDOS 6.0
19903 00002F81 E8BBFD                     call    FastOpen_Delete         ;;BN00X delete dir info in fastopen DOS 3.3
19904                                   jmp_to_builddest: ; 28/12/2022
19905                                       ; 31/01/2024
19906 00002F84 16                         push    ss
19907 00002F85 07                         pop     es
19908 00002F86 E9F9FE                     JMP     BUILDDEST               ;;BN00X
19909
19910                                     ; MSDOS 6.0
19911                                   Ren_Directory:
19912 00002F89 E8C1FD                     call    FastOpen_Rename         ;;BN00X delete dir info in fastopen DOS 3.3
19913                                     ;JMP    BUILDDEST
19914                                     ; 28/12/2022
19915 00002F8C EBF6                       jmp     short jmp_to_builddest
19916
19917                                   RENAME_OVER:
19918 00002F8E F8                         CLC
19919                                     ;JMP    RENAME_CLEAN ; 10/08/2018
19920                                     ; 28/12/2022
19921 00002F8F EBE6                       jmp     short jmp_to_rename_clean
19922
19923                                   ;--------------------------------------------------------------------------
19924                                   ; Procedure: GET_SOURCE
19925                                   ;
19926                                   ; Inputs:
19927                                   ;   RENAMEDMA has source info
19928                                   ; Function:
19929                                   ;   Re-find the source
19930                                   ; Output:
19931                                   ;   [CURBUF] set
19932                                   ;   [CURBUF+2]:BX points to entry
19933                                   ;   Carry set if error (currently user FAILed to I 24)
19934                                   ; DS preserved, others destroyed
19935                                   ;--------------------------------------------------------------------------
19936
19937                                   GET_SOURCE:
19938                                       ;;;
19939                                       ; 01/02/2024 - Retro DOS v5.0
19940                                       ; (PCDOS 7.1 IBMDOS.COM)
19941 00002F91 C42E[8A05]                 les     bp,[THISDPB]
19942 00002F95 31DB                       xor     bx,bx ; 0
19943                                     ;cmp    [es:bp+0Fh],bx
19944 00002F97 26395E0F                   cmp     [es:bp+DPB.FAT_SIZE],bx ; > 0 ?
19945 00002F9B 7504                       jnz     short gs_cont           ; yes, it is not FAT32
19946 00002F9D 8B1E[3106]                 mov     bx,[RENAMEDMA+17]       ; DirStart+2
19947                                   gs_cont:
19948 00002FA1 891E[EE0A]                 mov     [ROOTCLUST_HW],bx       ; hw of cluster number
19949                                     ;;;
```

```
00002FA5 8B1E[2F06]                 MOV     BX,[RENAMEDMA+15]     ; DirStart
                                    ; 01/02/2024
                                    ;LES    BP,[THISDPB]
00002FA9 E8B319                     call    SETDIRSRCH
00002FAC 7214                       JC      short gs_ret_label    ; retc
00002FAE E8021E                     call    STARTSRCH
00002FB1 A1[2D06]                   MOV     AX,[RENAMEDMA+13]     ; Lastent
                                    ;call   GETENT
                                    ; 18/12/2022
00002FB4 E9EB18                     jmp     GETENT
                                ;gs_ret_label:
                                    ;retn

                                ; MSDOS 6.0
                                ;--------------------------------------------------------------------------
                                ;Procedure: NEW_RENAME
                                ;
                                ;Input: DS:SI -> raw string with ?
                                ;   ES:DI -> destination string
                                ;   DS:BX -> source string
                                ;Function: replace ? chars of raw string with chars in source string and
                                ;      put in destination string
                                ;Output: ES:DI-> new string
                                ;--------------------------------------------------------------------------

                                NEW_RENAME:
                                    ; 17/05/2019 - Retro DOS v4.0
                                NEWNAM:
                                    ; DOSCODE:680Eh (MSDOS 6.21, MSDOS.SYS)
00002FB7 AC                         LODSB
00002FB8 3C3F                       CMP     AL,"?" ; 3Fh
00002FBA 7502                       JNZ     short NOCHG
00002FBC 8A07                       MOV     AL,[BX]               ; Get replace char
                                NOCHG:
00002FBE AA                         STOSB
00002FBF 43                         INC     BX                    ; Next replace char
00002FC0 E2F5                       LOOP    NEWNAM
                                    ; MSDOS 6.0
                                gs_ret_label:       ; 18/12/2022
00002FC2 C3                         retn

                                ;==========================================================================
                                ; FINFO.ASM, MSDOS 6.0, 1991
                                ;==========================================================================
                                ; 08/08/2018 - Retro DOS v3.0
                                ; 17/05/2019 - Retro DOS v4.0

                                ;** Low level routines for returning file information and setting file
                                ;   attributes
                                ;
                                ;   GET_FILE_INFO
                                ;   SET_FILE_ATTRIBUTE
                                ;
                                ;   Modification history:
                                ;
                                ;       Created: ARR 30 March 1983
                                ;
                                ;   M025: Return access_denied if attempting to set
                                ;         attribute of root directory.
                                ;

                                ;SUBTTL GET_FILE_INFO -- Get File Information

                                ;--------------------------------------------------------------------------
                                ; Procedure Name : GET_FILE_INFO
                                ;
                                ; Inputs:
                                ;   [WFP_START] Points to WFP string ("d:/" must be first 3 chars, NUL
                                ;           terminated)
                                ;   [CURR_DIR_END] Points to end of Current dir part of string
                                ;           ( = -1 if current dir not involved, else
                                ;            Points to first char after last "/" of current dir part)
                                ;   [THISCDS] Points to CDS being used
                                ;           (Low word = -1 if NUL CDS (Net direct request))
                                ;   [SATTRIB] Is attribute of search, determines what files can be found
                                ; Function:
                                ;   Get Information about a file
                                ; Returns:
                                ;   CARRY CLEAR
                                ;       AX = Attribute of file
                                ;       CX = Time stamp of file
                                ;       DX = Date stamp of file
                                ;       BX:DI = Size of file (32 bit)
                                ;   CARRY SET
                                ;       AX is error code
                                ;           error_file_not_found
                                ;                   Last element of path not found
                                ;           error_path_not_found
                                ;                   Bad path (not in curr dir part if present)
                                ;           error_bad_curr_dir
                                ;                   Bad path in current directory part of path
                                ; DS preserved, others destroyed
                                ;--------------------------------------------------------------------------

                                    ; 14/11/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)

                                GET_FILE_INFO:

                                ;hkn; get_file_info is called from file.asm and fcbio.asm. DS has been set
                                ;hkn; to DOSDATA at this point. So DOSassume is OK.

00002FC3 E863E8                     call    TestNet
00002FC6 7306                       JNC     short LOCAL_INFO

                                ;IF NOT Installed
                                ;   transfer NET_GET_FILE_INFO
                                ;ELSE
                                ;   MOV     AX,(MultNET SHL 8) OR 15
                                ;   INT     2FH
                                ;   return

00002FC8 B80F11                     mov     ax, 110Fh
00002FCB CD2F                       int     2Fh     ; Multiplex - NETWORK REDIRECTOR - GET REMOTE FILE'S ATTRIBUTES
                                                    ; SS = DOS CS, SDA first filename pointer -> fully-qualified name of file
                                                    ; SDA CDS pointer -> current directory
                                                    ; Return: CF set on error, AX = file attributes
00002FCD C3                         retn
                                ;ENDIF

                                LOCAL_INFO:
00002FCE E811E9                     call    ECritDisk
00002FD1 C606[4C03]01               MOV     byte [NoSetDir],1     ; if we find a dir, don't change to it
                                    ; MSDOS 3.3
                                    ;call   GETPATH
```

```
                                    ; MSDOS 6.0
00002FD6 E8C900                     call    GET_FAST_PATH
                                    ; MSDOS 3.3 & MSDOS 6.0
00002FD9 7312                       JNC     short info_check_dev
                                NO_PATH:
00002FDB 750B                       JNZ     short bad_path1
00002FDD 08C9                       OR      CL,CL
00002FDF 7407                       JZ      short bad_path1
                                info_no_file:
00002FE1 B80200                     MOV     AX,error_file_not_found
                                BadRet:
00002FE4 F9                         STC
                                JustRet:
                                    ;call   LCritDisk
                                    ;retn
                                    ; 18/12/2022
00002FE5 E927E9                     jmp     LCritDisk

                                bad_path1:
00002FE8 B80300                     MOV     AX,error_path_not_found
00002FEB EBF7                       jmp     short BadRet

                                info_check_dev:
00002FED 08E4                       OR      AH,AH
00002FEF 78F0                       JS      short info_no_file    ; device

                                    ; MSDOS 6.0
                                ;SR;
                                ; If root dir then CurBuf == -1. Check for this case and return subdir attr
                                ;for a root dir

00002FF1 833E[E205]FF               cmp     word [CURBUF],-1      ;is it a root dir?
00002FF6 7506                       jne     short not_root        ;no, CurBuf ptr is valid

00002FF8 30E4                       xor     ah,ah
00002FFA B010                       mov     al,attr_directory ; 10h
                                    ;clc
                                    ; 14/11/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
                                    ; (DOSCODE:683Eh)
                                    ; 16/12/2022
                                    ;clc
00002FFC EBE7                       jmp     short JustRet

                                not_root:
                                    ; MSDOS 3.3 (& MSDOS 6.0)
00002FFE 1E                         PUSH    DS
00002FFF 8E1E[E405]                 MOV     DS,[CURBUF+2]
00003003 89DE                       MOV     SI,BX
00003005 31DB                       XOR     BX,BX                 ; Assume size=0 (dir)
00003007 89DF                       MOV     DI,BX
                                    ;mov    cx,[si+16h]
00003009 8B4C16                     MOV     CX,[SI+dir_entry.dir_time]
                                    ;mov    dx,[si+18h]
0000300C 8B5418                     MOV     DX,[SI+dir_entry.dir_date]
0000300F 30E4                       XOR     AH,AH
                                    ;mov    al,[si+0Bh]
00003011 8A440B                     MOV     AL,[SI+dir_entry.dir_attr]
                                    ;test   al,10h
00003014 A810                       TEST    AL,attr_directory
00003016 7506                       JNZ     short NO_SIZE
                                    ;mov    di,[si+1Ch]
00003018 8B7C1C                     MOV     DI,[SI+dir_entry.dir_size_l]
                                    ;mov    bx,[si+1Eh]
0000301B 8B5C1E                     MOV     BX,[SI+dir_entry.dir_size_h]
                                NO_SIZE:
0000301E 1F                         POP     DS
                                    ;CLC
                                    ; 14/11/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
                                    ; (DOSCODE:6864h)
                                    ; 16/12/2022
                                    ;clc
0000301F EBC4                       jmp     short JustRet

                                ;Break      <SET_FILE_ATTRIBUTE -- Set File Attribute>
                                ;-----------------------------------------------------------------------------
                                ; Procedure Name : SET_FILE_ATTRIBUTE
                                ; Inputs:
                                ;   [WFP_START] Points to WFP string ("d:/" must be first 3 chars, NUL
                                ;           terminated)
                                ;   [CURR_DIR_END] Points to end of Current dir part of string
                                ;           ( = -1 if current dir not involved, else
                                ;            Points to first char after last "/" of current dir part)
                                ;   [THISCDS] Points to CDS being used
                                ;           (Low word = -1 if NUL CDS (Net direct request))
                                ;   [SATTRIB] is attribute of search (determines what files may be found)
                                ;   AX is new attributes to give to file
                                ; Function:
                                ;   Set File Attributes
                                ; Returns:
                                ;   CARRY CLEAR
                                ;       No error
                                ;   CARRY SET
                                ;       AX is error code
                                ;           error_file_not_found
                                ;                   Last element of path not found
                                ;           error_path_not_found
                                ;                   Bad path (not in curr dir part if present)
                                ;           error_bad_curr_dir
                                ;                   Bad path in current directory part of path
                                ;           error_access_denied
                                ;                   Attempt to set an attribute which cannot be set
                                ;                   (attr_directory, attr_volume_ID)
                                ;           error_sharing_violation
                                ;                   Sharing mode of file did not allow the change
                                ;                   (this request requires exclusive write/read access)
                                ;                   (INT 24H generated)
                                ; DS preserved, others destroyed
                                ;-----------------------------------------------------------------------------

                                    ; 01/02/2024 - Retro DOS v5.0

                                SET_FILE_ATTRIBUTE:

                                ;hkn; set_file_attr is called from file.asm. DS has been set
                                ;hkn; to DOSDATA at this point. So DOSassume is OK.

00003021 A9D8FF                     TEST    AX,~attr_changeable ; 0FFD8h
00003024 7412                       JZ      short set_look
                                _BAD_ACC:
                                    ;MOV    byte [EXTERR_LOCUS],errLOC_Unk ; 1
                                    ;;;
                                    ; 01/02/2024 - Retro DOS v5.0
                                    ; (PCDOS 7.1 IBMDOS.COM)
00003026 E8B4E2                     call    set_exerr_locus_unk
```

```
                                    ;;;
00003029 C606[2703]07               MOV     byte [EXTERR_CLASS],errCLASS_Apperr ; 7
0000302E C606[2603]04               MOV     byte [EXTERR_ACTION],errACT_Abort ; 4
00003033 B80500                     MOV     AX,error_access_denied ; 5
00003036 F9                         STC
00003037 C3                         retn

                                set_look:
00003038 E8EEE7                     call    TestNet
0000303B 7308                       JNC     short LOCAL_SET

                                ;IF NOT Installed
                                ;   transfer NET_SEQ_SET_FILE_ATTRIBUTE
                                ;ELSE
0000303D 50                         PUSH    AX

                                    ;MOV    AX,(MultNET SHL 8) OR 14
                                    ;INT    2FH

0000303E B80E11                     mov     ax, 110Eh
00003041 CD2F                       int     2Fh     ; Multiplex - NETWORK REDIRECTOR - SET REMOTE FILE'S ATTRIBUTES
                                                    ; SS = DOS CS, SDA first filename pointer -> fully-qualified name of file
                                                    ; SDA CDS pointer -> current directory
                                                    ; STACK: WORD new file attributes
                                                    ; Return: CF set on error

00003043 5B                         POP     BX                  ; clean stack
00003044 C3                         retn
                                ;ENDIF

                                LOCAL_SET:
00003045 E89AE8                     call    ECritDisk
00003048 50                         PUSH    AX                  ; Save new attributes
00003049 C606[4C03]01               MOV     byte [NoSetDir],1   ; if we find a dir, don't change to it
0000304E E8C21A                     call    GETPATH             ; get path through fastopen if there ;AC000;
00003051 7308                       JNC     short set_check_device
00003053 5B                         POP     BX                  ; Clean stack (don't zap AX)
00003054 EB85                       JMP     short NO_PATH

                                    ; MSDOS 6.0
                                cannot_set_root:                ; M025:
00003056 B80500                     mov     ax,error_access_denied; M025: return error is attempting
                                    ;stc                        ; M025: to set attr. of root
                                    ;jmp    short OK_BYE        ; M025:
                                    ; 01/02/2024
00003059 EB89                       jmp     short BadRet

                                set_check_device:
0000305B 08E4                       OR      AH,AH
0000305D 7906                       JNS     short set_check_share
0000305F 58                         POP     AX
00003060 E8ACE8                     call    LCritDisk
00003063 EBC1                       JMP     short _BAD_ACC              ; device

                                set_check_share:
00003065 58                         POP     AX                  ; Get new attributes

                                    ; MSDOS 6.0
00003066 833E[E205]FF               cmp     word [CURBUF], -1   ; M025: Q: is this the root dir
0000306B 74E9                       je      short cannot_set_root ; M025: Y: return error

                                    ; MSDOS 3.3 & MSDOS 6.0
0000306D E851FC                     call    REN_DEL_Check
00003070 7305                       JNC     short set_do
00003072 B82000                     MOV     AX,error_sharing_violation ; 32
00003075 EB28                       jmp     short OK_BYE

                                set_do:
                                    ; MSDOS 3.3 & MSDOS 6.0
00003077 C43E[E205]                 LES     DI,[CURBUF]
0000307B 2680670BD8                 AND     BYTE [ES:BX+dir_entry.dir_attr],~attr_changeable ; 0D8h
00003080 2608470B                   OR      BYTE [ES:BX+dir_entry.dir_attr],AL

                                ; 01/02/2024 - Retro DOS v5.0
                                %if 0
                                    ; MSDOS 6.0
                                    TEST    byte [ES:DI+BUFFINFO.buf_flags],buf_dirty
                                                                    ;LB. if already dirty           ;AN000;
                                    JNZ     short yesdirty3         ;LB.  don't increment dirty count ;AN000;
                                    call    INC_DIRTY_COUNT         ;LB.                            ;AN000;

                                    OR      byte [ES:DI+BUFFINFO.buf_flags],buf_dirty
                                yesdirty3:
                                %else
                                    ; 01/02/2024
                                    ; (PCDOS 7.1 IBMDOS.COM)
00003084 E8373B                     call    SET_BUF_DIRTY
                                %endif
00003087 A0[7605]                   MOV     AL,[THISDRV]
                                ;;;; 10/1/86 F.C update fastopen cache
0000308A 52                         PUSH    DX
0000308B 57                         PUSH    DI
0000308C B400                       MOV     AH,0            ; dir entry update
0000308E 88C2                       MOV     DL,AL           ; drive number A=0,B=1,,
00003090 89DF                       MOV     DI,BX           ; ES:DI -> dir entry
00003092 E8D2FC                     call    FastOpen_Update
00003095 5F                         POP     DI
00003096 5A                         POP     DX
                                ;;;; 9/11/86 F.C update fastopen cache
00003097 E8FE39                     call    FLUSHBUF
0000309A 7303                       JNC     short OK_BYE
0000309C B80200                     MOV     AX,error_file_not_found
                                OK_BYE:
                                    ;call   LCritDisk
                                    ;retn
                                    ; 16/12/2022
0000309F E96DE8                     jmp     LCritDisk

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

                                    ; 17/05/2019 - Retro DOS v4.0

                                    ; MSDOS 6.0
                                GET_FAST_PATH:
                                ;hkn; use SS override for FastOpenFlg
000030A2 36800E[4611]01             OR      byte [ss:FastOpenFlg],FastOpen_Set
                                                                    ;FO. trigger fastopen ;AN000;
000030A8 E8681A                     call    GETPATH
000030AB 9C                         PUSHF                           ;FO.                  ;AN000;
000030AC 368026[4611]80             AND     byte [ss:FastOpenFlg],Fast_yes
                                                                    ;FO. clear all fastopen flags ;AN000;
000030B2 9D                         POPF                            ;FO.                  ;AN000;
000030B3 C3                         retn

```

```
                        ;============================================================================
                        ; DUP.ASM, MSDOS 6.0, 1991
                        ;============================================================================
                        ; 08/08/2018 - Retro DOS v3.0
                        ; 17/05/2019 - Retro DOS v4.0

                        ;** Low level DUP routine for use by EXEC when creating a new process.
                        ;   Exports the DUP to the server machine and increments the SFT ref count
                        ;
                        ;   DOS_DUP
                        ;
                        ;   Modification history:
                        ;
                        ;     Created: ARR 30 March 1983

                        ;BREAK <DOS_DUP -- DUP SFT across network>
                        ;----------------------------------------------------------------------------
                        ; Procedure Name : DOS_DUP
                        ;
                        ; Inputs:
                        ;   [THISSFT] set to the SFT for the file being DUPed
                        ;           (a non net SFT is OK, in this case the ref
                        ;            count is simply incremented)
                        ; Function:
                        ;   Signal to the devices that a logical open is occurring
                        ; Returns:
                        ;   ES:DI point to SFT
                        ;    Carry clear
                        ;   SFT ref_count is incremented
                        ; Registers modified: None.
                        ; NOTE:
                        ;   This routine is called from $CREATE_PROCESS_DATA_BLOCK at DOSINIT
                        ;   time with SS NOT DOSGROUP. There will be no Network handles at
                        ;   that time.
                        ;----------------------------------------------------------------------------

                        DOS_DUP:
                            ;LES    DI,[CS:THISSFT]  ; MSDOS 3.3
                            ; MSDOS 6.0
 000030B4 2E8E06[0700]      mov     es,[cs:DosDSeg]
 000030B9 26C43E[9E05]      les     di,[es:THISSFT]

                            ;Entry  Dos_Dup_Direct
                        DOS_Dup_Direct:
 000030BE E881E7            call    IsSFTNet
 000030C1 7503              JNZ     short DO_INC
 000030C3 E84E21            call    DEV_OPEN_SFT
                        DO_INC:
                            ;INC    word [ES:DI+SF_ENTRY.sf_ref_count]
 000030C6 26FF05            inc     word [ES:DI]            ; Clears carry (if this ever wraps
                                                            ;   we're in big trouble anyway)
 000030C9 C3                retn

                        ;============================================================================
                        ; CREATE.ASM, MSDOS 6.0, 1991
                        ;============================================================================
                        ; 08/08/2018 - Retro DOS v3.0
                        ; 18/05/2019 - Retro DOS v4.0

                        ;TITLE      DOS_CREATE/DOS_CREATE_NEW - Internal CREATE calls for MS-DOS
                        ;NAME       DOS_CREATE
                        ;----------------------------------------------------------------------------
                        ;** Internal Create and Create new to create a local or NET file and SFT.
                        ;
                        ;   DOS_CREATE
                        ;   DOS_CREATE_NEW
                        ;   SET_MKND_ERR
                        ;   SET_Media_ID
                        ;   SET_EXT_Mode
                        ;
                        ;   Revision history:
                        ;
                        ;       A000 version 4.00   Jan. 1988
                        ;       A001  D490 -- Change IOCTL subfunctios from 63h,43h to 66h, 46h

                        ;Installed = TRUE

                        ;   i_need  THISSFT,DWORD
                        ;   i_need  THISCDS,DWORD
                        ;   I_need  EXTERR,WORD
                        ;   I_Need  ExtErr_locus,BYTE
                        ;   I_need  JShare,DWORD
                        ;   I_need  VOLCHNG_FLAG,BYTE
                        ;   I_need  SATTRIB,BYTE
                        ;   I_need  CALLVIDM,DWORD
                        ;   I_need  EXTOPEN_ON,BYTE                 ;AN000; extended open
                        ;   I_need  NAME1,BYTE                      ;AN000;
                        ;   I_need  NO_NAME_ID,BYTE                 ;AN000;
                        ;   I_need  Packet_Temp,WORD                ;AN000;
                        ;   I_need  DOS34_FLAG,WORD                 ;AN000;
                        ;   I_need  SAVE_BX,WORD                    ;AN000;

                        ;***DOS_CREATE - Create a File
                        ;----------------------------------------------------------------------------
                        ;   DOS_Create is called to create the specified file, truncating
                        ;   the old one if it exists.
                        ;
                        ;   ENTRY   AX is Attribute to create
                        ;           (ds) = DOSDATA
                        ;           [WFP_START] Points to WFP string ("d:/" must be first 3 chars, NUL
                        ;                   terminated)
                        ;           [CURR_DIR_END] Points to end of Current dir part of string
                        ;                   ( = -1 if current dir not involved, else
                        ;                    Points to first char after last "/" of current dir part)
                        ;           [THISCDS] Points to CDS being used
                        ;                   (Low word = -1 if NUL CDS (Net direct request))
                        ;           [THISSFT] Points to SFT to fill in if file created
                        ;                   (sf_mode field set so that FCB may be detected)
                        ;           [SATTRIB] Is attribute of search, determines what files can be found
                        ;
                        ;   EXIT    sf_ref_count is NOT altered
                        ;           CARRY CLEAR
                        ;               THISSFT filled in.
                        ;                   sf_mode = unchanged for FCB, sharing_compat + open_for_both
                        ;           CARRY SET
                        ;               AX is error code
                        ;                   error_path_not_found
                        ;                           Bad path (not in curr dir part if present)
                        ;                   error_bad_curr_dir
                        ;                           Bad path in current directory part of path
                        ;                   error_access_denied
                        ;                           Attempt to re-create read only file , or
                        ;                           create a second volume id or create a dir
```

```
                                ;                   error_sharing_violation
                                ;                           The sharing mode was correct but not allowed
                                ;                           generates an INT 24
                                ;   USES    all but DS
                                ;--------------------------------------------------------------------------

                                    ; 14/11/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
                                    ; DOSCODE:6920h (MSDOS 5.0, MSDOS.SYS)

                                    ; 01/02/2024 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMDOS.COM)
                                    ; DOSCODE:708Ah (PCDOS 7.1, IBMDOS.COM)

                                DOS_CREATE:
                                    ; 18/05/2019 - Retro DOS v4.0
                                    ; DOSCODE:6934h (MSDOS 6.21, MSDOS.SYS)

                                ;hkn; dispatched to from file.asm and fcbio.asm. DS set up to DOSDATA at
                                ;hkn; this point.

000030CA 30E4                       XOR     AH,AH           ; Truncate is OK

                                ;   Enter here from Dos_Create_New
                                ;
                                ;   (ah) = 0 iff truncate OK

                                Create_inter:
000030CC A8C0                       TEST    AL,~(attr_all+attr_ignore+attr_volume_id) ; 80h
                                                            ; Mask out any meaningless bits
000030CE 7511                       JNZ     short AttErr
000030D0 A808                       TEST    AL,attr_volume_id
000030D2 7407                       JZ      short NoReset

                                    ; MSDOS 6.0
                                    ; 16/12/2022
000030D4 800E[1106]80               OR      byte [DOS34_FLAG],DBCS_VOLID ; 80h ;AN000;FOR dbcs volid
                                    ; 07/12/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
                                    ;or     word [DOS34_FLAG],DBCS_VOLID ; 80h

000030D9 B008                       MOV     AL,attr_volume_id ; 8
                                NoReset:
000030DB 0C20                       OR      AL,attr_archive ; File changed  ; 20h
000030DD A850                       TEST    AL,attr_directory+attr_device ; 50h
000030DF 7408                       JZ      short ATT_OK
                                AttErr:
000030E1 B80500                     MOV     AX,5            ; Attribute problem
                                    ;MOV    byte [EXTERR_LOCUS],errLOC_Unk ; 1
                                    ; 01/02/2024
000030E4 E8F6E1                     call    set_exerr_locus_unk
000030E7 EB62                       JMP     SHORT SET_MKND_ERR ; Gotta use MKDIR to make dirs, NEVER allow
                                                                ;   attr_device to be set.
                                ATT_OK:
000030E9 C43E[9E05]                 LES     DI,[THISSFT]
000030ED 06                         PUSH    ES
000030EE C436[A205]                 LES     SI,[THISCDS]
000030F2 83FEFF                     CMP     SI,-1
000030F5 751B                       JNE     short TEST_RE_NET

                                ;   No CDS, it must be redirected.

000030F7 07                         POP     ES

                                    ; MSDOS 6.0
                                ;Extended open hooks
                                    ;test   byte [EXTOPEN_ON],1
000030F8 F606[F605]01               TEST    byte [EXTOPEN_ON],EXT_OPEN_ON ;AN000;EO. from extended open
000030FD 740D                       JZ      short NOEXTOP               ;AN000;EO. no, do normal
                                IFS_extopen:                                    ;AN000;EO.
000030FF 50                         PUSH    AX                          ;AN000;EO. pass create attr
                                    ;MOV    AX,(MultNET SHL 8) OR 46    ;AN000;EO. issue extended open verb
00003100 B82E11                     mov     ax,112Eh
                                NOEXTOP2:   ; 01/02/2024 (PCDOS 7.1 IBMDOS.COM)
00003103 CD2F                       INT     2FH                         ;AN000;EO.
00003105 5B                         POP     BX                          ;AN000;EO. trash bx
00003106 C606[F605]00               MOV     byte [EXTOPEN_ON],0         ;AN000;EO.
0000310B C3                         retn                                ;AN000;EO.
                                NOEXTOP:                                ;AN000;
                                ;Extended open hooks

                                ;IF NOT Installed
                                ;   transfer NET_SEQ_CREATE
                                ;ELSE
0000310C 50                         PUSH    AX

                                    ;MOV    AX,(MultNET SHL 8) OR 24
                                    ;INT    2FH

0000310D B81811                     mov     ax,1118h
                                    ; 01/02/2024
                                    ;int     2Fh    ; Multiplex - NETWORK REDIRECTOR - CREATE/TRUNCATE FILE
                                                    ; ES:DI -> uninitialized SFT, SS = DOS CS
                                                    ; SDA first filename pointer -> fully-qualified name of file
                                                    ; STACK: WORD file creation mode???

                                    ;POP    BX                  ; BX is trashed anyway
                                    ;retn
00003110 EBF1                       jmp     short NOEXTOP2 ; 01/02/2024
                                ;ENDIF

                                ;   We have a CDS. See if it's network

                                TEST_RE_NET:
                                    ;;test  word [es:si+43h],8000h
                                    ;TEST   word [ES:SI+curdir.flags],curdir_isnet
                                    ; 07/12/2022
                                    ;test   byte [es:si+44h],80h
                                    ; 17/12/2022
00003112 26F6444480                 test    byte [ES:SI+curdir.flags+1],curdir_isnet>>8
00003117 07                         POP     ES
00003118 7417                       JZ      short LOCAL_CREATE

                                    ; MSDOS 6.0
0000311A E8CA00                     CALL    Set_EXT_mode                ;AN000;EO.
0000311D 7205                       JC      SHORT dochk                 ;AN000;EO.
                                    ;;or    word [es:di+2],2
                                    ;OR     word [ES:DI+SF_ENTRY.sf_mode],SHARING_COMPAT+open_for_both ;IFS.
                                    ; 17/12/2022
0000311F 26804D0202                 or      byte [ES:DI+SF_ENTRY.sf_mode],SHARING_COMPAT+open_for_both ;IFS.

                                ;Extended open hooks
                                dochk:
00003124 F606[F605]01               TEST    byte [EXTOPEN_ON],EXT_OPEN_ON ;AN000;EO. from extended open
00003129 75D4                       JNZ     short IFS_extopen           ;AN000;EO. yes, issue extended open
                                ;Extended open hooks
```

```
                                ;IF NOT Installed
                                ;   transfer NET_CREATE
                                ;ELSE
0000312B 50                         PUSH    AX

                                    ;MOV    AX,(MultNET SHL 8) OR 23
                                    ;INT    2FH

0000312C B81711                     mov     ax,1117h

                                    ; 01/02/2024
                                    ;int     2Fh   ; Multiplex - NETWORK REDIRECTOR - CREATE/TRUNCATE REMOTE FILE
                                                   ; ES:DI -> uninitialized SFT, SS = DOS CS
                                                   ; SDA first filename pointer -> fully-qualified name of file to open
                                                   ; SDA CDS pointer -> current directory
                                                   ; Return: CF set on error

                                    ;POP    BX                  ; BX is trashed anyway
                                ;nomore:
                                    ;retn
0000312F EBD2                       jmp     short NOEXTOP2 ; 01/02/2024
                                ;ENDIF

                                ;** It's a local create. We have a local CDS for it.

                                LOCAL_CREATE:
                                    ; MSDOS 6.0
00003131 E8B300                     CALL    Set_EXT_mode   ;AN000;EO. set mode if from extended open
00003134 7205                       JC      short setdone  ;AN000;EO.

                                    ; MSDOS 3.3 & MSDOS 6.0
                                    ; 17/12/2022
                                    ;;or    word [es:di+2],2
                                    ;OR     word [ES:DI+SF_ENTRY.sf_mode],SHARING_COMPAT+open_for_both
                                    ;or     byte [es:di+2],2
00003136 26804D0202                 or      byte [ES:DI+SF_ENTRY.sf_mode],SHARING_COMPAT+open_for_both
                                setdone:
0000313B E8A4E7                     call    ECritDisk
0000313E E81A23                     call    MakeNode
00003141 7317                       JNC     short Create_ok
00003143 C606[A10A]FF               mov     byte [VOLCHNG_FLAG],-1; indicate no change in volume label
00003148 E8C4E7                     call    LCritDisk

                                    ;entry  SET_MKND_ERR
                                SET_MKND_ERR:

                                ;   Looks up MakeNode errors and converts them. AL is MakeNode
                                ;   error, SI is GETPATH bad spot return if path_not_found error.

                                ;hkn; CRTERRTAB is in TABLE seg (DOSCODE)
0000314B BB[5231]                   MOV     BX,CRTERRTAB
                                    ;XLAT  ; MSDOS 3.3
                                    ; 18/05/2019 - Retro DOS v4.0
0000314E 2E                         CS
0000314F D7                         XLAT
                                CreatBadRet:
00003150 F9                         STC
00003151 C3                         retn

                                ; 13/05/2019 - Retro DOS v4.0
                                ; DOSCODE:69C4h (MSDOS 6.21, MSDOS.SYS)
                                ; ---------------------------------------------------------------------------

                                ;** Internal Create and Create new to create a local or NET file and SFT.

                                ; 17/07/2018 - Retro DOS v3.0
                                ; Offset 12B1h of IBMDOS.COM (MSDOS 3.3), 1987

                                ;CRTERRTAB: ; 19/07/2018 - MSDOS 3.3
                                ;   db      0,5,52h,50h,3,5,20h

                                ;CRTERRTAB: ; 18/05/2019 - MSDOS 6.0
                                ;   db      0,5,52h,50h,3,5,20h,2

                                ; 08/08/2018

                                CRTERRTAB:  ;LABEL BYTE    ; Lookup table for MakeNode returns
00003152 00                         DB      0                       ; none
00003153 05                         DB      error_access_denied     ; MakeNode error 1
00003154 52                         DB      error_cannot_make       ; MakeNode error 2
00003155 50                         DB      error_file_exists       ; MakeNode error 3
00003156 03                         DB      error_path_not_found    ; MakeNode error 4
00003157 05                         DB      error_access_denied     ; MakeNode error 5
00003158 20                         DB      error_sharing_violation ; MakeNode error 6
                                    ; MSDOS 6.0
00003159 02                         DB      error_file_not_found    ; MakeNode error 7

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

                                ; We have just created a new file. This results in the truncation of old
                                ; files. We must inform the sharer to slash all the open SFT's for this
                                ; file to the current size.

                                ; If we created a volume id on the diskette, set the VOLCHNG_FLAG to logical
                                ; drive number to force a Build BPB after Media Check.

                                ;;; FASTOPEN 8/29/86
                                Create_ok:
0000315A E8E2FB                     call    FastOpen_Delete
                                ;;; FASTOPEN 8/29/86
0000315D A0[6D05]                   mov     al,[SATTRIB]
00003160 A808                       test    al,attr_volume_id
00003162 741C                       jz      short NoVolLabel
00003164 C43E[A205]                 LES     DI,[THISCDS]
                                    ;mov    ah,[ES:DI+curdir.text]; get drive letter
00003168 268A25                     mov     ah,[ES:DI] ; 09/08/2018
0000316B 80EC41                     sub     ah,'A' ; 41h            ; convert to drive number
0000316E 8826[A10A]                 mov     [VOLCHNG_FLAG],ah       ;Set flag to indicate volid change

                                    ; 18/05/2019 - Retro DOS v4.0

                                    ; MSDOS 6.0
00003172 B701                       MOV     BH,1                    ;AN000;>32mb set volume id to boot record
00003174 E81F00                     CALL    Set_Media_ID            ;AN000;>32mb

00003177 E868E7                     call    ECritDisk
0000317A E8DE33                     call    FATREAD_CDS             ; force a media check
0000317D E88FE7                     call    LCritDisk

                                NoVolLabel:
00003180 B80200                     MOV     ax,2
00003183 C43E[9E05]                 LES     DI,[THISSFT]
                                ;if installed
                                    ;call   JShare + 14 * 4
```

```
20694 00003187 FF1E[C800]               call    far [JShare+(14*4)] ; 14 = ShSU
20695                                  ;else
20696                                  ;   Call    ShSU
20697                                  ;endif
20698 0000318B E881E7                   call    LCritDisk
20699 0000318E E95601                   jmp     SET_SFT_MODE
20700
20701                                  ;--------------------------------------------------------------------------
20702                                  ; Procedure Name : Dos_Create_New
20703                                  ;
20704                                  ; Inputs:
20705                                  ;   [WFP_START] Points to WFP string ("d:/" must be first 3 chars, NUL
20706                                  ;           terminated)
20707                                  ;   [CURR_DIR_END] Points to end of Current dir part of string
20708                                  ;           ( = -1 if current dir not involved, else
20709                                  ;            Points to first char after last "/" of current dir part)
20710                                  ;   [THISCDS] Points to CDS being used
20711                                  ;           (Low word = -1 if NUL CDS (Net direct request))
20712                                  ;   [THISSFT] Points to SFT to fill in if file created
20713                                  ;           (sf_mode field set so that FCB may be detected)
20714                                  ;   [SATTRIB] Is attribute of search, determines what files can be found
20715                                  ;   AX is Attribute to create
20716                                  ; Function:
20717                                  ;   Try to create the specified file truncating an old one that exists
20718                                  ; Outputs:
20719                                  ;   sf_ref_count is NOT altered
20720                                  ;   CARRY CLEAR
20721                                  ;       THISSFT filled in.
20722                                  ;           sf_mode = sharing_compat + open_for_both for Non-FCB SFT
20723                                  ;   CARRY SET
20724                                  ;       AX is error code
20725                                  ;           error_path_not_found
20726                                  ;                   Bad path (not in curr dir part if present)
20727                                  ;           error_bad_curr_dir
20728                                  ;                   Bad path in current directory part of path
20729                                  ;           error_access_denied
20730                                  ;                   Create a second volume id or create a dir
20731                                  ;           error_file_exists
20732                                  ;                   Already a file by this name
20733                                  ; DS preserved, others destroyed
20734                                  ;--------------------------------------------------------------------------
20735
20736                                  DOS_Create_New:
20737 00003191 B401                       MOV     AH,1            ; Truncate is NOT OK
20738 00003193 E936FF                   JMP     Create_inter
20739
20740                                  ; MSDOS 6.0
20741                                  ;--------------------------------------------------------------------------
20742                                  ; Procedure Name : Set_Media_ID
20743                                  ;
20744                                  ; Inputs:
20745                                  ;   NAME1= Volume ID
20746                                  ;   BH= 0, delete volume id
20747                                  ;       1, set new volume id
20748                                  ;   DS= DOSGROUP
20749                                  ; Function:
20750                                  ;   Set Volume ID to DOS 4.00 Boot record.
20751                                  ; Outputs:
20752                                  ;   CARRY CLEAR
20753                                  ;       volume id set
20754                                  ;   CARRY SET
20755                                  ;       AX is error code
20756                                  ;--------------------------------------------------------------------------
20757
20758                                      ; 18/05/2019 - Retro DOS v4.0
20759                                      ; 01/02/2024 - Retro DOS v5.0
20760                                  Set_Media_ID:
20761 00003196 50                       PUSH    AX              ;AN000;;>32mb
20762 00003197 06                       PUSH    ES              ;AN000;;>32mb
20763 00003198 57                       PUSH    DI              ;AN000;;>32mb
20764
20765 00003199 FEC4                     INC     AH              ;AN000;;>32mb  bl=drive #
20766 0000319B 88E3                     MOV     BL,AH           ;AN000;;>32mb  bl=drive # (A=1,B=2,,,)
20767 0000319D B00D                     MOV     AL,0DH          ;AN000;;>32mb  generic IOCTL
20768                                   ;MOV    CX,0866H        ;AN001;;>32mb  get media id
20769                                   ; 01/02/2024
20770                                   ; (PCDOS 7.1 IBMDOS.COM)
20771 0000319F B96648                   mov     cx,4866h        ; ch = FAT32 disk drive (CATEGORY CODE)
20772                                                           ; cl = minor code,
20773                                                           ;      get volume serial number (and name)
20774
20775                                  ;hkn; PACKET_TEMP is in DOSDATA
20776 000031A2 BA[A00D]                 MOV     DX,Packet_Temp ;AN000;>32mb
20777
20778                                  Set_Media_ID_1:
20779                                      ; 01/02/2024
20780 000031A5 51                       push    cx
20781
20782 000031A6 53                       PUSH    BX              ;AN000;;>32mb
20783 000031A7 52                       PUSH    DX              ;AN000;;>32mb
20784 000031A8 30FF                     XOR     BH,BH           ;AN000;;>32mb
20785
20786                                   ;invoke $IOCTL          ;AN000;;>32mb
20787 000031AA E8D7F6                   call    _$IOCTL
20788
20789 000031AD 5A                       POP     DX              ;AN000;;>32mb
20790 000031AE 5B                       POP     BX              ;AN000;;>32mb
20791
20792                                   ; 01/02/2024
20793 000031AF 59                       pop     cx
20794                                   ;JC     short geterr    ;AN000;;>32mb
20795 000031B0 730A                     jnc     short Set_Media_ID_2
20796 000031B2 80FD48                   cmp     ch,48h          ; is it FAT32 disk drive request?
20797 000031B5 F9                       stc
20798 000031B6 7529                     jne     short geterr    ; (ch=8 request failed!)
20799 000031B8 B508                     mov     ch,8            ; set category code for (old) FAT disk drive
20800                                                           ; (except FAT32)
20801 000031BA EBE9                     jmp     short Set_Media_ID_1 ; and try again
20802
20803                                  Set_Media_ID_2:
20804 000031BC 08FF                     OR      BH,BH           ;AN000;;>32mb delete volume id
20805 000031BE 7405                     JZ      short NoName    ;AN000;>32mb yes
20806
20807                                  ;hkn; NAME1 is in DOSDATA
20808 000031C0 BE[4B05]                 MOV     SI,NAME1        ;AN000;>32mb
20809
20810 000031C3 EB03                     JMP     SHORT doset     ;AN000;>32mb yes
20811                                  NoName:                  ;AN000;
20812
20813                                  ;hkn; NO_NAME_ID is in DOSDATA
20814 000031C5 BE[C10D]                 MOV     SI,NO_NAME_ID   ;AN000;>32mb
20815
20816                                  doset:                           ;AN000;
20817 000031C8 89D7                     MOV     DI,DX           ;AN000;;>32mb
```

```
                                    ;add    di,6
000031CA 83C706                     ADD     DI,MEDIA_ID_INFO.MEDIA_Label ;AN000;;>32mb

                                ;hkn; ES & DS must point to SS
                                ;hkn;       PUSH    CS              ;AN000;;>32mb  move new volume id to packet
000031CD 16                         PUSH    SS              ;AN000;;>32mb  move new volume id to packet

000031CE 1F                         POP     DS              ;AN000;;>32mb

                                ;hkn;       PUSH    CS              ;AN000;;>32mb
000031CF 16                         PUSH    SS              ;AN000;;>32mb

000031D0 07                         POP     ES              ;AN000;;>32mb

                                    ; 01/02/2024
000031D1 51                         push    cx

000031D2 B90B00                     MOV     CX,11           ;AN000;;>32mb
000031D5 F3A4                       REP     MOVSB           ;AN000;;>32mb

                                    ;MOV    CX,0846H        ;AN001;;>32mb
                                    ; 01/02/2024
000031D7 59                         pop     cx
000031D8 B146                       mov     cl,46h          ; set volume serial number (and name)
                                    ;
000031DA B00D                       MOV     AL,0DH          ;AN000;;>32mb
000031DC 30FF                       XOR     BH,BH           ;AN000;;>32mb
                                    ;invoke $IOCTL          ;AN000;;>32mb  set volume id
000031DE E8A3F6                     call    _$IOCTL
                                geterr:                     ;AN000;
                                ;hkn;       PUSH    CS              ;AN000;>32mb
000031E1 16                         PUSH    SS              ;AN000;>32mb

000031E2 1F                         POP     DS              ;AN000;>32mb   ds= dosgroup

000031E3 5F                         POP     DI              ;AN000;;>32mb
000031E4 07                         POP     ES              ;AN000;;>32mb
000031E5 58                         POP     AX              ;AN000;;>32mb
000031E6 C3                         retn                    ;AN000;>32mb

                                ; MSDOS 6.0
                                ;--------------------------------------------------------------------------
                                ; Procedure Name : Set_EXT_mode
                                ;
                                ; Inputs:
                                ;   [EXTOPEN_ON]= flag for extended open
                                ;   SAVE_BX= mode specified in Extended Open
                                ; Function:
                                ;   Set mode in ThisSFT
                                ; Outputs:
                                ;   carry set,mode is set if from Extended Open
                                ;   carry clear, mode not set yet
                                ;--------------------------------------------------------------------------

                                ; 13/05/2019 - Retro DOS v4.0

                                Set_EXT_mode:

                                ;hkn; SS override
000031E7 36F606[F605]01             TEST    byte [ss:EXTOPEN_ON],EXT_OPEN_ON ;AN000;EO. from extended open
000031ED 740B                       JZ      short NOTEX             ;AN000;EO. no, do normal
000031EF 50                         PUSH    AX                      ;AN000;EO.

                                ;hkn; SS override
000031F0 36A1[0106]                 MOV     AX,[ss:SAVE_BX]                 ;AN000;EO.
                                    ;or     [es:di+2],ax
000031F4 26094502                   OR      [ES:DI+SF_ENTRY.sf_mode],AX ;AN000;EO.
000031F8 58                         POP     AX                      ;AN000;EO.
000031F9 F9                         STC                             ;AN000;EO.
                                NOTEX:                              ;AN000;
000031FA C3                         retn                    ;AN000;EO.

                                ;============================================================================
                                ; OPEN.ASM, MSDOS 6.0, 1991
                                ;============================================================================
                                ; 08/08/2018 - Retro DOS v3.0
                                ; 18/05/2019 - Retro DOS v4.0

                                ;   TITLE   DOS_OPEN - Internal OPEN call for MS-DOS
                                ;   NAME    DOS_OPEN

                                ;** OPEN.ASM - File Open
                                ;--------------------------------------------------------------------------
                                ;   Low level routines for openning a file from a file spec.
                                ;   Also misc routines for sharing errors
                                ;
                                ;   DOS_Open
                                ;   Check_Access_AX
                                ;   SHARE_ERROR
                                ;   SET_SFT_MODE
                                ;   Code_Page_Mismatched_Error              ; DOS 4.00
                                ;
                                ;   Revision history:
                                ;
                                ;       Created: ARR 30 March 1983
                                ;       A000       version 4.00   Jan. 1988
                                ;
                                ;   M034 - The value in save_bx must be pushed on to the stack for
                                ;          remote extended opens and not save_cx.
                                ;
                                ;   M035 - if open made from exec then we must set the appropriate bits
                                ;          on the stack before calling off to the redir.
                                ;   M042 - Bit 11 of DOS34_FLAG set indicates that the redir knows how
                                ;          to handle open from exec. In this case set the appropriate bit
                                ;          else do not.
                                ;--------------------------------------------------------------------------

                                ;Installed = TRUE

                                ;   i_need  NoSetDir,BYTE
                                ;   i_need  THISSFT,DWORD
                                ;   i_need  THISCDS,DWORD
                                ;   i_need  CURBUF,DWORD
                                ;   i_need  CurrentPDB,WORD
                                ;   i_need  CURR_DIR_END,WORD
                                ;   I_need  RetryCount,WORD
                                ;   I_need  Open_Access,BYTE
                                ;   I_need  fSharing,BYTE
                                ;   i_need  JShare,DWORD
                                ;   I_need  FastOpenFlg,byte
                                ;   I_need  EXTOPEN_ON,BYTE                 ;AN000;; DOS 4.00
                                ;   I_need  ALLOWED,BYTE                    ;AN000;; DOS 4.00
                                ;   I_need  EXTERR,WORD                     ;AN000;; DOS 4.00
                                ;   I_need  EXTERR_LOCUS,BYTE               ;AN000;; DOS 4.00
```

```
20942                                   ;   I_need  EXTERR_ACTION,BYTE             ;AN000;; DOS 4.00
20943                                   ;   I_need  EXTERR_CLASS,BYTE              ;AN000;; DOS 4.00
20944                                   ;   I_need  CPSWFLAG,BYTE                  ;AN000;; DOS 4.00
20945                                   ;   I_need  EXITHOLD,DWORD                 ;AN000;; DOS 4.00
20946                                   ;   I_need  THISDPB,DWORD                   ;AN000;; DOS 4.00
20947                                   ;   I_need  SAVE_CX,WORD                    ;AN000;; DOS 4.00
20948                                   ;   I_need  SAVE_BX,WORD                    ;M034
20949                                   ;
20950                                   ;   I_need  DOS_FLAG,BYTE
20951                                   ;   I_need  DOS34_FLAG,WORD                       ;M042
20952
20953                                   ;Break      <DOS_Open - internal file access>
20954                                   ;-----------------------------------------------------------------------------
20955                                   ; Procedure Name : DOS_Open
20956                                   ;
20957                                   ; Inputs:
20958                                   ;   [WFP_START] Points to WFP string ("d:/" must be first 3 chars, NUL
20959                                   ;           terminated)
20960                                   ;   [CURR_DIR_END] Points to end of Current dir part of string
20961                                   ;           ( = -1 if current dir not involved, else
20962                                   ;            Points to first char after last "/" of current dir part)
20963                                   ;   [THISCDS] Points to CDS being used
20964                                   ;           (Low word = -1 if NUL CDS (Net direct request))
20965                                   ;   [THISSFT] Points to SFT to fill in if file found
20966                                   ;           (sf_mode field set so that FCB may be detected)
20967                                   ;   [SATTRIB] Is attribute of search, determines what files can be found
20968                                   ;   AX is Access and Sharing mode
20969                                   ;     High NIBBLE of AL (Sharing Mode)
20970                                   ;           sharing_compat    file is opened in compatibility mode
20971                                   ;           sharing_deny_none  file is opened Multi reader, Multi writer
20972                                   ;           sharing_deny_read  file is opened Only reader, Multi writer
20973                                   ;           sharing_deny_write file is opened Multi reader, Only writer
20974                                   ;           sharing_deny_both  file is opened Only reader, Only writer
20975                                   ;     Low NIBBLE of AL (Access Mode)
20976                                   ;           open_for_read  file is opened for reading
20977                                   ;           open_for_write file is opened for writing
20978                                   ;           open_for_both  file is opened for both reading and writing.
20979                                   ;
20980                                   ;      For FCB SFTs AL should = sharing_compat + open_for_both
20981                                   ;           (not checked)
20982                                   ; Function:
20983                                   ;   Try to open the specified file
20984                                   ; Outputs:
20985                                   ;   sf_ref_count is NOT altered
20986                                   ;   CARRY CLEAR
20987                                   ;       THISSFT filled in.
20988                                   ;   CARRY SET
20989                                   ;       AX is error code
20990                                   ;           error_file_not_found
20991                                   ;                   Last element of path not found
20992                                   ;           error_path_not_found
20993                                   ;                   Bad path (not in curr dir part if present)
20994                                   ;           error_bad_curr_dir
20995                                   ;                   Bad path in current directory part of path
20996                                   ;           error_invalid_access
20997                                   ;                   Bad sharing mode or bad access mode or bad combination
20998                                   ;           error_access_denied
20999                                   ;                   Attempt to open read only file for writting, or
21000                                   ;                   open a directory
21001                                   ;           error_sharing_violation
21002                                   ;                   The sharing mode was correct but not allowed
21003                                   ;                   generates an INT 24 on compatibility mode SFTs
21004                                   ; DS preserved, others destroyed
21005                                   ;-----------------------------------------------------------------------------
21006
21007                                   ; 18/05/2019 - Retro DOS v4.0
21008                                   ; DOSCODE:6A60h (MSDOS 6.21, MSDOS.SYS)
21009                                   ; 14/11/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
21010                                   ; DOSCODE:6A4Ch (MSDOS 5.0, MSDOS.SYS)
21011
21012                                   ; 01/02/2024 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMDOS.COM)
21013                                   ; DOSCODE:71BBh (PCDOS 7.1, IBMDOS.COM)
21014
21015                                   DOS_OPEN:
21016                                       ; DS has been set up to DOSDATA in file.asm and fcbio2.asm.
21017
21018 000031FB C606[4C03]00               MOV     byte [NoSetDir],0
21019 00003200 E83301                     CALL    Check_Access_AX
21020 00003203 722B                       JC      short do_ret_label        ; retc
21021
21022 00003205 C43E[9E05]                 LES     DI,[THISSFT]
21023 00003209 30E4                       XOR     AH,AH
21024
21025                                     ; sleaze! move only access/sharing mode in. Leave sf_isFCB unchanged
21026
21027 0000320B 26884502                   MOV     [ES:DI+SF_ENTRY.sf_mode],AL ; For moment do this on FCBs too
21028 0000320F 06                         PUSH    ES
21029 00003210 C436[A205]                 LES     SI,[THISCDS]
21030                                     ; 18/08/2018
21031 00003214 83FEFF                     CMP     SI,-1 ; 0FFFFh
21032 00003217 7530                       JNZ     short TEST_RE_NET1
21033 00003219 07                         POP     ES
21034
21035                                     ; MSDOS 6.0
21036                                   ;Extended open hooks
21037 0000321A F606[F605]01               TEST    byte [EXTOPEN_ON],EXT_OPEN_ON ;FT. from extended open             ;AN000;
21038 0000321F 7410                       JZ      short _NOEXTOP                    ;FT. no, do normal              ;AN000;
21039                                   _IFS_extopen:                                                               ;AN000;
21040 00003221 A0[0106]                   MOV     AL,[SAVE_BX]              ; M034 - save_bx has original bx
21041                                                                       ; with which call was made. This
21042                                                                       ; has the open access bits.
21043                                     ;;MOV   AL,[SAVE_CX]              ; M034 - FT. al= create attribute
21044
21045 00003224 50                         PUSH    AX                        ;FT. pass create attr to IFS     ;AN000;
21046                                     ;mov    ax,112Eh
21047                                     ;MOV    AX,(MultNET SHL 8) OR 46    ;FT. issue extended open verb  ;AN000;
21048 00003225 B82E11                     mov     ax,(MultNET*256)+46
21049 00003228 CD2F                       INT     2FH                       ;FT.                             ;AN000;
21050 0000322A 5B                         POP     BX                        ;FT. trash bx                    ;AN000;
21051 0000322B C606[F605]00               MOV     byte [EXTOPEN_ON],0       ;FT.                             ;AN000;
21052
21053                                   do_ret_label:
21054 00003230 C3                         retn                              ;FT.                             ;AN000;
21055                                   _NOEXTOP:
21056                                   ;Extended open hooks
21057                                       ;
21058                                   ;IF NOT Installed
21059                                       ;transfer NET_SEQ_OPEN
21060                                   ;ELSE
21061
21062                                   do_net_int2f:
21063 00003231 F606[8600]01               test    byte [DOS_FLAG],EXECOPEN ; Q: was this open call made from exec
21064 00003236 7409                       jz      short not_exec_open   ; N: just do net open
21065                                                                     ; Y: check to see if redir is aware
```

```
                                                        ;    of this
                                                        ;
                                                        ; M042 - start
                                    ;test   word [DOS34_FLAG],EXEC_AWARE_REDIR ; 800h
00003238 F606[1206]08               test    byte [DOS34_FLAG+1],(EXEC_AWARE_REDIR>>8)
                                                        ; Q: does this redir know how to
                                                        ;    this
0000323D 7402                       jz      short not_exec_open   ; N: just do net open
                                                        ; Y: set bit 3 of access byte and
                                                        ;    set sharing mode to DENY_WRITE
                                                        ; M042 - end

                                    ; NOTE: This specific mode has not been set for the code assembled
                                    ; under the "NOT Installed" conditional. Currently Installed is
                                    ; always one.
                                                        ; M035 - set the bits on the stack
                                    ;mov    al,23h
0000323F B023                       mov     AL,SHARING_DENY_WRITE+EXEC_OPEN

                                not_exec_open:
                                    ; MSDOS 3.3 & MSDOS 6.0
00003241 50                         PUSH    AX

                                    ;MOV    AX,(MultNET SHL 8) OR 22
                                    ;INT    2FH

00003242 B81611                     mov     ax,1116h
00003245 CD2F                       int     2Fh     ; Multiplex - NETWORK REDIRECTOR - OPEN EXISTING REMOTE FILE
                                                    ; ES:DI -> uninitialized SFT, SS = DOS CS
                                                    ; SDA first filename pointer -> fully-qualified name of file to open
                                                    ; STACK: WORD file open mode
                                                    ; Return: CF set on error

00003247 5B                         POP     BX                    ; clean stack
                                ;do_ret_label: ; 09/08/2018
00003248 C3                         retn
                                ;ENDIF

                                TEST_RE_NET1:
                                    ;TEST   word [ES:SI+curdir.flags],curdir_isnet
                                    ; 17/12/2022
00003249 26F6444480                 test    byte [ES:SI+curdir.flags+1],curdir_isnet>>8
0000324E 07                         POP     ES
                                    ; 18/05/2019
0000324F 7409                       JZ      short LOCAL_OPEN

                                ;Extended open hooks
                                    ; MSDOS 6.0
00003251 F606[F605]01               TEST    byte [EXTOPEN_ON],EXT_OPEN_ON ;FT. from extended open      ;AN000;
00003256 75C9                       JNZ     short _IFS_extopen          ;FT. issue extended open       ;AN000;
                                ;Extended open hooks

                                ;IF NOT Installed
                                ;   transfer NET_OPEN
                                ;ELSE
00003258 EBD7                       jmp     short do_net_int2f
                                ;ENDIF

                                LOCAL_OPEN:
                                    ; MSDOS 3.3 & MSDOS 6.0
0000325A E885E6                     call    ECritDisk

                                ; DOS 3.3 FastOPen 6/16/86

                                    ;or     byte [FastOpenFlg],5
0000325D 800E[4611]05               OR      byte [FastOpenFlg],FastOpen_Set+Special_Fill_Set ; only open can

00003262 E8AE18                     call    GETPATH

                                ; DOS 3.3 FastOPen 6/16/86

00003265 731D                       JNC     short Open_found
00003267 7511                       JNZ     short bad_path2
00003269 08C9                       OR      CL,CL
0000326B 740D                       JZ      short bad_path2
                                OpenFNF:
0000326D B80200                     MOV     AX,error_file_not_found       ; 2
                                OpenBadRet:
                                ;hkn; FastOpenFlg is in DOSDATA use SS override
                                    ; 12/08/2018
                                    ;mov    byte [cs:FastOpenFlg],0 ; IBMDOS.COM (MSDOS 3.3) offset 36CAh
                                    ; MSDOS 6.0
00003270 368026[4611]80             AND     BYTE [SS:FastOpenFlg],Fast_yes    ;; DOS 3.3
00003276 F9                         STC
                                    ;call   LCritDisk
                                    ; 16/12/2022
00003277 E995E6                     jmp     LCritDisk
                                    ;;JMP   Clear_FastOpen ; 10/08/2018
                                    ;retn   ; 08/09/2018
                                    ; 14/11/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
                                    ;jmp    Clear_FastOpen

                                bad_path2:
0000327A B80300                     MOV     AX,error_path_not_found       ; 3
0000327D EBF1                       JMP     short OpenBadRet

                                Open_Bad_Access:
0000327F B80500                     MOV     AX,error_access_denied; 5
00003282 EBEC                       JMP     short OpenBadRet

                                Open_found:
00003284 74F9                       JZ      short Open_Bad_Access ; test for directories
00003286 08E4                       OR      AH,AH
00003288 783E                       JS      short open_ok         ; Devices don't have attributes
0000328A 8E06[E405]                 MOV     ES,[CURBUF+2]         ; get buffer location
                                    ;mov    al,[es:bx+0Bh]
0000328E 268A470B                   MOV     AL,[ES:BX+dir_entry.dir_attr]
00003292 A808                       TEST    AL,attr_volume_id     ; can't open volume ids
00003294 75E9                       JNZ     short Open_Bad_Access
00003296 A801                       TEST    AL,attr_read_only     ; check write on read only
00003298 742E                       JZ      short open_ok

                                ; The file is marked READ-ONLY. We verify that the open mode allows access to
                                ; the read-only file. Unfortunately, with FCB's and net-FCB's we cannot
                                ; determine at the OPEN time if such access is allowed. Thus, we defer such
                                ; processing until the actual write operation:
                                ;
                                ; If FCB, then we change the mode to be read_only.
                                ; If net_FCB, then we change the mode to be read_only.
                                ; If not open for read then error.

0000329A 1E                         push    ds
0000329B 56                         push    si
0000329C C536[9E05]                 LDS     SI,[THISSFT]
```

```
21190                                   ;mov    cx,[si+2]
21191 000032A0 8B4C02                   MOV     CX,[SI+SF_ENTRY.sf_mode]
21192                                   ; 17/12/2022
21193                                   ;test   ch,80h
21194 000032A3 F6C580                   test    ch,sf_isFCB>>8
21195                                   ;TEST   CX,sf_isFCB ; 8000h   ; is it FCB?
21196 000032A6 750A                     JNZ     short ResetAccess     ; yes, reset the access
21197 000032A8 88CA                     MOV     DL,CL
21198 000032AA 80E2F0                   AND     DL,SHARING_MASK       ; 0F0h
21199 000032AD 80FA70                   CMP     DL,SHARING_NET_FCB ; 70h ; is it net FCB?
21200 000032B0 7508                     JNZ     short NormalOpen      ; no
21201                                 ResetAccess:
21202                                   ; 14/11/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
21203                                   ;AND    CX,~access_mask       ; 0FFF0h ; clear access
21204                                   ; 16/12/2022
21205                                   ;and    cl,0F0h ; 18/05/2019
21206                                   ;;;
21207                                   ; 01/02/2024 - Retro DOS v5.0
21208                                   ; (PCDOS 7.1 IBMDOS.COM)
21209                                   ;and    cx,0FFFCh
21210 000032B2 80E1FC                   and     cl,0FCh ; ~3 ; ~open_mode_mask ; clear access
21211                                   ;;;
21212                                 ;   OR      CX,open_for_read ; 0  ; stick in open_for_read
21213 000032B5 894C02                   MOV     [SI+SF_ENTRY.sf_mode],CX
21214 000032B8 EB0C                     JMP     SHORT FillSFT
21215
21216                                 ; The SFT is normal. See if the requested access is open_for_read
21217
21218                                 NormalOpen:
21219                                   ;AND    CL,access_mask ; 0Fh   ; remove extras
21220                                   ;;;
21221                                   ; 01/02/2024 - Retro DOS v5.0
21222                                   ; (PCDOS 7.1 IBMDOS.COM)
21223 000032BA 80E103                   and     cl,3                  ; it was 'and cl,0Fh' in MSDOS 6.22
21224                                                                 ; (and cl,access_mask)
21225                                                                 ; this may be open_mode_mask
21226                                   ;;;
21227 000032BD 80F900                   CMP     CL,open_for_read ; 0  ; is it open for read?
21228 000032C0 7404                     JZ      short FillSFT  ; yes
21229 000032C2 5E                       pop     si
21230 000032C3 1F                       pop     ds
21231 000032C4 EBB9                     JMP     short Open_Bad_Access
21232                                 ;
21233                                 ; All done, restore registers and fill the SFT.
21234                                 ;
21235                                 FillSFT:
21236 000032C6 5E                       pop     si
21237 000032C7 1F                       pop     ds
21238                                 open_ok:
21239 000032C8 E83823                   call    DOOPEN                ; Fill in SFT
21240
21241                                 ;hkn; FastOpenFlg is in DOSDATA. use SS override
21242                                   ; 18/05/2019
21243                                   ;and    byte [ss:FastOpenFlag],80h
21244 000032CB 368026[4611]80           AND     BYTE [SS:FastOpenFlg],Fast_yes       ;; DOS 3.3
21245                                   ; 12/08/2018
21246                                   ;and    byte [FastOpenFlg],Fast_yes
21247
21248                                   ; MSDOS 6.0
21249 000032D1 E84300                   CALL    DO_SHARE_CHECK
21250 000032D4 7303                     JNC     short SHARE_OK
21251                                   ;call   LCritDisk
21252                                   ; 16/12/2022
21253 000032D6 E936E6                   jmp     LCritDisk
21254                                   ;;JMP   short Clear_FastOpen
21255                                   ;retn   ; 18/05/2019
21256                                   ; 14/11/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
21257                                   ;jmp    short Clear_FastOpen
21258
21259                                   ; MSDOS 3.3
21260                                 ;DO_SHARE_CHECK:
21261                                 ;   MOV     CX,[RetryCount]             ; Get # tries to do
21262                                 ;OpenShareRetry:
21263                                 ;   push    cx                    ; Save number left to do
21264                                 ;   call    SHARE_CHECK           ; Final Check
21265                                 ;   pop     cx                    ; CX = # left
21266                                 ;   JNC     short SHARE_OK        ; No problem with access
21267                                 ;   call    Idle
21268                                 ;   LOOP    OpenShareRetry        ; One more retry used up
21269                                 ;OpenShareFail:
21270                                 ;   LES     DI,[THISSFT]
21271                                 ;   call    SHARE_ERROR
21272                                 ;   JNC     short DO_SHARE_CHECK  ; User wants more retry
21273
21274                                   ;12/08/2018
21275                                   ;mov    byte [ss:FastOpenFlg],0
21276                                   ;08/09/2018
21277                                   ;mov    byte [FastOpenFlg],0
21278                                   ;call   LCritDisk
21279                                   ;JMP    short Clear_FastOpen
21280                                   ;retn
21281
21282                                 SHARE_OK:
21283                                   ; MSDOS 3.3 & MSDOS 6.0
21284 000032D9 B80300                   MOV     AX,3
21285 000032DC C43E[9E05]               LES     DI,[THISSFT]
21286                                 ;if installed
21287                                   ;call   JShare + 14 * 4
21288 000032E0 FF1E[C800]               call    far [JShare+(14*4)]  ; 14 = ShSU
21289                                 ;else
21290                                 ;   Call    ShSU
21291                                 ;endif
21292 000032E4 E828E6                   call    LCritDisk
21293
21294                                   ;FallThru Set_SFT_Mode
21295
21296                                 ;--------------------------------------------------------------------------
21297                                 ; Procedure Name : SET_SFT_MODE
21298                                 ;
21299                                 ; Finish SFT initialization for new reference. Set the correct mode.
21300                                 ;
21301                                 ;   Inputs:
21302                                 ;   ThisSFT points to SFT
21303                                 ;
21304                                 ;   Outputs:
21305                                 ;   Carry clear
21306                                 ;   Registers modified: AX.
21307                                 ;--------------------------------------------------------------------------
21308
21309                                 ;hkn; called from create. DS already set up to DOSDATA.
21310
21311                                 SET_SFT_MODE:
21312 000032E7 C43E[9E05]               LES     DI,[THISSFT]
21313 000032EB E8261F                   call    DEV_OPEN_SFT
```

```
21314                                   ;test   word [es:di+2],8000h
21315                                   ; 17/12/2022
21316                                   ;test   byte [es:di+3],80h
21317 000032EE 26F6450380               test    byte [ES:DI+SF_ENTRY.sf_mode+1],sf_isFCB>>8
21318                                   ;TEST   word [ES:DI+SF_ENTRY.sf_mode],sf_isFCB ; Clears carry
21319 000032F3 7407                     JZ      short Clear_FastOpen  ; sf_mode correct (retz)
21320 000032F5 A1[3003]                 MOV     AX,[CurrentPDB]
21321                                   ;mov    [es:di+31h],ax
21322 000032F8 26894531                 MOV     [ES:DI+SF_ENTRY.sf_PID],AX ; For FCB sf_PID=PDB
21323
21324                                 Clear_FastOpen:
21325 000032FC C3                       retn                            ;;;;; DOS 3.3
21326
21327                                 ;--------------------------------------------------------------------------
21328                                 ; Procedure Name : SHARE_ERROR
21329                                 ;
21330                                 ; Called on sharing violations. ES:DI points to SFT. AX has error code
21331                                 ; If SFT is FCB or compatibility mode gens INT 24 error.
21332                                 ; Returns carry set AX=error_sharing_violation if user says ignore (can't
21333                                 ; really ignore). Carry clear if user wants a retry. ES, DI, DS preserved
21334                                 ;--------------------------------------------------------------------------
21335
21336                                 SHARE_ERROR:
21337                                     ; 17/12/2022
21338                                     ;test   byte [es:di+3],80h
21339 000032FD 26F6450380               test    byte [ES:DI+SF_ENTRY.sf_mode+1],sf_isFCB>>8 ; 80h
21340                                   ;TEST   word [ES:DI+SF_ENTRY.sf_mode],sf_isFCB ; 8000h
21341 00003302 7509                     JNZ     short _HARD_ERR
21342 00003304 268A4D02                 MOV     CL,[ES:DI+SF_ENTRY.sf_mode]
21343 00003308 80E1F0                   AND     CL,SHARING_MASK  ; 0F0h
21344                                   ;CMP    CL,SHARING_COMPAT ; 0
21345                                   ;JNE    short _NO_HARD_ERR
21346                                   ; 21/09/2023
21347 0000330B 7505                     jnz     short _NO_HARD_ERR
21348                                 _HARD_ERR:
21349 0000330D E83C51                   call    SHARE_VIOLATION
21350                                   ;retnc                          ; User wants retry
21351 00003310 73EA                     jnc     short Clear_FastOpen
21352                                 _NO_HARD_ERR:
21353 00003312 B82000                   MOV     AX,error_sharing_violation  ; 20h
21354 00003315 F9                       STC
21355 00003316 C3                       retn
21356
21357                                 ; MSDOS 6.0
21358                                 ;--------------------------------------------------------------------------
21359                                 ; Procedure Name : DO_SHARE_CHECK
21360                                 ;
21361                                 ; Input: THISDPB, WFP_Start, THISSFT set
21362                                 ; Functions: check file sharing mode is valid
21363                                 ; Output: carry set, error
21364                                 ;     carry clear, share ok
21365                                 ;--------------------------------------------------------------------------
21366
21367                                     ; 18/05/2019 - Retro DOS v4.0
21368                                 DO_SHARE_CHECK:
21369 00003317 E8C8E5                   call    ECritDisk               ; enter critical section
21370                                 OPN_RETRY:
21371 0000331A 8B0E[1A00]               MOV     CX,[RetryCount]                 ; Get # tries to do
21372                                 OpenShareRetry:
21373 0000331E 51                       push    cx                      ; Save number left to do
21374 0000331F E82551                   call    SHARE_CHECK             ; Final Check
21375 00003322 59                       pop     cx                      ; CX = # left
21376 00003323 730E                     JNC     short Share_Ok2                 ; No problem with access
21377 00003325 E8BCE4                   call    Idle
21378 00003328 E2F4                     LOOP    OpenShareRetry          ; One more retry used up
21379                                 OpenShareFail:
21380 0000332A C43E[9E05]               LES     DI,[THISSFT]
21381 0000332E E8CCFF                   call    SHARE_ERROR
21382 00003331 73E7                     JNC     short OPN_RETRY                 ; User wants more retry
21383                                 Share_Ok2:
21384                                   ;call   LCritDisk               ; leave critical section
21385                                   ;retn
21386                                   ; 18/12/2022
21387 00003333 E9D9E5                   jmp     LCritDisk
21388
21389                                 ;--------------------------------------------------------------------------
21390                                 ; Procedure Name : Check_Access
21391                                 ;
21392                                 ; Inputs:
21393                                 ;   AX is mode
21394                                 ;     High NIBBLE of AL (Sharing Mode)
21395                                 ;           sharing_compat    file is opened in compatibility mode
21396                                 ;           sharing_deny_none  file is opened Multi reader, Multi writer
21397                                 ;           sharing_deny_read  file is opened Only reader, Multi writer
21398                                 ;           sharing_deny_write file is opened Multi reader, Only writer
21399                                 ;           sharing_deny_both  file is opened Only reader, Only writer
21400                                 ;     Low NIBBLE of AL (Access Mode)
21401                                 ;           open_for_read  file is opened for reading
21402                                 ;           open_for_write file is opened for writing
21403                                 ;           open_for_both  file is opened for both reading and writing.
21404                                 ; Function:
21405                                 ;   Check this access mode for correctness
21406                                 ; Outputs:
21407                                 ;   [open_access] = AL input
21408                                 ;   Carry Clear
21409                                 ;           Mode is correct
21410                                 ;           AX unchanged
21411                                 ;   Carry Set
21412                                 ;           Mode is bad
21413                                 ;           AX = error_invalid_access
21414                                 ; No other registers effected
21415                                 ;--------------------------------------------------------------------------
21416
21417                                     ; 23/01/2024 - Retro DOS v5.0
21418                                 Check_Access_AX:
21419 00003336 A2[6E05]                 MOV     [OPEN_ACCESS],AL
21420 00003339 53                       PUSH    BX
21421
21422                                 ;   If sharing, then test for special sharing mode for FCBs
21423
21424 0000333A 88C3                     MOV     BL,AL
21425 0000333C 80E3F0                   AND     BL,SHARING_MASK ; 0F0h
21426
21427                                   ;CMP    byte [FSHARING],-1
21428                                   ;JNZ    short CheckShareMode  ; not through server call, must be ok
21429                                   ; 23/01/2024
21430                                   ; PCDOS 7.1 IBMDOS.COM
21431 0000333F 803E[7205]00             cmp     byte [FSHARING],0
21432 00003344 7405                     jz      short CheckShareMode  ; not through server call, must be ok
21433
21434 00003346 80FB70                   CMP     BL,SHARING_NET_FCB ; 70h
21435 00003349 7405                     JZ      short CheckAccessMode ; yes, we have an FCB
21436                                 CheckShareMode:
21437 0000334B 80FB40                   CMP     BL,40h                  ; is this a good sharing mode?
```

```
21438 0000334E 770D                     JA      short Make_Bad_Access
21439                                 CheckAccessMode:
21440 00003350 88C3                     MOV     BL,AL
21441                                   ; 23/01/2024
21442 00003352 80E303                   and     bl,3 ; PCDOS 7.1 IBMDOS.COM
21443                                   ;AND    BL,access_mask ; 0Fh
21444 00003355 80FB02                   CMP     BL,2
21445 00003358 7703                     JA      short Make_Bad_Access
21446 0000335A 5B                       POP     BX
21447 0000335B F8                       CLC
21448 0000335C C3                       retn
21449
21450                                 Make_Bad_Access:
21451 0000335D B80C00                   MOV     AX,error_invalid_access ; 0Ch
21452 00003360 5B                       POP     BX
21453 00003361 F9                       STC
21454 00003362 C3                       retn
21455
21456                                 ;============================================================================
21457                                 ; DINFO.ASM, MSDOS 6.0, 1991
21458                                 ;============================================================================
21459                                 ; 08/08/2018 - Retro DOS v3.0
21460                                 ; 18/05/2019 - Retro DOS v4.0
21461                                 ; 02/02/2024 - Retro DOS v5.0
21462
21463                                 ;** Low level routine for returning disk drive information from a local
21464                                 ;     or NET device
21465                                 ;
21466                                 ;   DISK_INFO
21467                                 ;
21468                                 ;     Modification history:
21469                                 ;
21470                                 ;           Created: ARR 30 March 1983
21471
21472                                 ;   Break  <DISK_INFO -- Get Disk Drive Information>
21473                                 ;--------------------------------------------------------------------------
21474                                 ; Procedure Name : DISK_INFO
21475                                 ;
21476                                 ; Inputs:
21477                                 ;   [THISCDS] Points to the Macro List Structure of interest
21478                                 ;           (It MAY NOT be NUL, error not detected)
21479                                 ; Function:
21480                                 ;   Get Interesting Drive Information
21481                                 ; Returns:
21482                                 ;   DX = Number of free allocation units
21483                                 ;   BX = Total Number of allocation units on disk
21484                                 ;   CX = Sector size
21485                                 ;   AL = Sectors per allocation unit
21486                                 ;   AH = FAT ID BYTE
21487                                 ;   Carry set if error (currently user FAILed to I 24)
21488                                 ; Segs except ES preserved, others destroyed
21489                                 ;--------------------------------------------------------------------------
21490
21491                                 ;hkn; called from getset.asm and misc.asm. DS has already been set up to
21492                                 ;hkn; DOSDATA.
21493
21494                                     ; 02/02/2024 - Retro DOS v5.0
21495                                     ; PCDOS 7.1 IBMDOS.COM - DOSCODE:732Bh
21496
21497                                 DISK_INFO:
21498                                     ; 08/08/2018 - Retro DOS v3.0
21499                                     ; IBM DOS.COM (MSDOS 3.3, 1987) - Offset 37C5h
21500
21501 00003363 E8C3E4                    call    TestNet
21502 00003366 7318                     JNC     short LOCAL_DSK_INFO
21503
21504                                     ;;;
21505                                     ; 02/02/2024 (PCDOS 7.1 IBMDOS.COM)
21506 00003368 31F6                     xor     si,si  ; free cluster count hw = 0
21507 0000336A 31FF                     xor     di,di  ; number of clusters hw = 0
21508                                     ;;;
21509
21510                                 ;IF NOT Installed
21511                                 ;   transfer NET_DISK_INFO
21512                                 ;ELSE
21513                                     ;MOV    AX,(MultNET SHL 8) OR 12
21514                                     ;INT    2FH
21515                                     ;return
21516
21517 0000336C B80C11                    mov     ax,110Ch
21518 0000336F CD2F                     int     2Fh    ; Multiplex - NETWORK REDIRECTOR - GET DISK SPACE
21519                                                    ; ES:DI -> current directory
21520                                                    ; Return: AL = sectors per cluster, BX = total clusters
21521                                                    ; CX = bytes per sector, DX = number of available clusters
21522                                     ;;;
21523                                     ; 02/02/2024 (PCDOS 7.1 IBMDOS.COM)
21524 00003371 83FBFF                   cmp     bx,0FFFFh
21525 00003374 7502                     jne     short dsk_info_1
21526 00003376 89DF                     mov     di,bx
21527                                 dsk_info_1:
21528 00003378 83FAFF                   cmp     dx,0FFFFh
21529 0000337B 7502                     jne     short disk_info_retn
21530 0000337D 89D6                     mov     si,dx
21531                                 disk_info_retn:
21532                                     ;;;
21533 0000337F C3                       retn
21534                                 ;ENDIF
21535
21536                                 LOCAL_DSK_INFO:
21537                                     ;MOV    byte [EXTERR_LOCUS],errLOC_Disk
21538                                     ;;;
21539                                     ; 02/02/2024 (PCDOS 7.1 IBMDOS.COM)
21540 00003380 E863DF                   call    set_exerr_locus_disk
21541                                     ;;;
21542 00003383 E85CE5                   call    ECritDisk
21543 00003386 E8D231                   call    FATREAD_CDS            ; perform media check.
21544                                     ;JC     short CRIT_LEAVE
21545                                     ;;; 02/02/2024
21546 00003389 720D                     jc      short dsk_info_2
21547 0000338B 31C0                     xor     ax,ax
21548 0000338D A3[E80A]                 mov     [CLUSTNUM_HW],ax ; 0  ; clear high word of cluster number
21549                                     ;;;
21550 00003390 BB0200                   MOV     BX,2
21551 00003393 E84B2F                   call    UNPACK                 ; Get first FAT sector into CURBUF
21552                                     ;JC     short CRIT_LEAVE
21553                                     ;;;
21554                                     ; 02/02/2024 - Retro DOS v5.0
21555 00003396 7303                     jnc     short dsk_info_3
21556                                 dsk_info_2:
21557                                     ;jmp    CRIT_LEAVE
21558 00003398 E974E5                   jmp     LCritDisk
21559                                 dsk_info_3:
21560                                     ;;;
21561 0000339B C536[E205]               LDS     SI,[CURBUF]
```

```
                                    ; 02/02/2024
                                    ;;mov   ah,[si+20]
                                    ;mov    ah,[si+24] ; PCDOS 7.1 IBMDOS.COM
000033 9F 8A6418                    MOV     AH,[SI+BUFINSIZ]      ; get FAT ID BYTE

                                ;hkn; SS is DOSDATA
000033 A2 16                        push    ss
000033 A3 1F                        pop     ds

                                ; 02/02/2024 - Retro DOS v5.0
                                %if 0
                                    ;mov    cx,[es:bp+0Dh]
                                    MOV     CX,[ES:BP+DPB.MAX_CLUSTER]

                                ; Examine the current free count. If it indicates that we have an invalid
                                ; count, do the expensive calculation.

                                    ;mov    dx,[es:bp+1Fh]
                                    MOV     DX,[ES:BP+DPB.FREE_CNT] ; get free count
                                    CMP     DX,-1                 ; is it valid?
                                    JZ      short DoScan

                                ; Check to see if it is in a reasonable range. If so, trust it and return.
                                ; Otherwise, we need to blast out an internal error message and then recompute
                                ; the count.

                                    CMP     DX,CX                 ; is it in a reasonable range?
                                    JB      short GotVal          ; yes, trust it.
                                DoScan:
                                    XOR     DX,DX
                                    DEC     CX
                                SCANFREE:
                                    call    UNPACK
                                    JC      short CRIT_LEAVE
                                    JNZ     short NOTFREECLUS
                                    INC     DX                    ; A free one
                                NOTFREECLUS:
                                    INC     BX                    ; Next cluster
                                    LOOP    SCANFREE
                                    DEC     BX                    ; BX was next cluster. Convert to
                                ReturnVals:
                                    DEC     BX                    ; count
                                    ;mov    al,[es:bp+4]
                                    MOV     AL,[ES:BP+DPB.CLUSTER_MASK]
                                    INC     AL                    ; Sectors/cluster
                                    ;mov    cx,[es:bp+2]
                                    MOV     CX,[ES:BP+DPB.SECTOR_SIZE] ; Bytes/sector
                                    ;mov    [es:bp+1Fh],dx
                                    MOV     [ES:BP+DPB.FREE_CNT],DX
                                    CLC
                                CRIT_LEAVE:
                                    ;call   LCritDisk
                                    ;retn
                                    ; 17/12/2022
                                    jmp     LCritDisk

                                ; We have correctly computed everything previously. Load up registers for
                                ; return.

                                GotVal:
                                    MOV     BX,CX                 ; get cluster count
                                    JMP     short ReturnVals

                                %else
                                    ; 02/02/2024 (PCDOS 7.1 IBMDOS.COM)
000033 A4 31F6                      xor     si,si   ; 0
000033 A6 89F7                      mov     di,si   ; 0
                                    ;mov    dx,[es:bp+1Fh]
000033 A8 268B561F                  mov     dx,[es:bp+DPB.FREE_CNT]       ; get free count
                                    ;cmp    [es:bp+0Fh],si
000033 AC 2639760F                  cmp     [es:bp+DPB.FAT_SIZE],si ; FAT32 (16 bit FAT size = 0) ?
000033 B0 7406                      jz      short dsk_info_4      ; yes
                                    ;mov    cx,[es:bp+0Dh]
000033 B2 268B4E0D                  mov     cx,[es:bp+DPB.MAX_CLUSTER]
000033 B6 EB18                      jmp     short dsk_info_5 ; zf=0, si=di=0

                                dsk_info_4:
                                    ;mov    di,[es:bp+2Fh]
000033 B8 268B7E2F                  mov     di,[es:bp+DPB.LAST_CLUSTER+2]
                                    ;mov    cx,[es:bp+2Dh]
000033 BC 268B4E2D                  mov     cx,[es:bp+DPB.LAST_CLUSTER]
                                    ;mov    si,[es:bp+21h]
000033 C0 268B7621                  mov     si,[es:bp+DPB.FREE_CNT_HW] ; hw of free cluster count
000033 C4 39F2                      cmp     dx,si                 ; same (zero) ?
                                ;dsk_info_5:
000033 C6 7504                      jnz     short dsk_info_6      ; not same (not zero)
000033 C8 42                        inc     dx
000033 C9 740B                      jz      short dsk_info_8      ; 0FFFFh -> 0 (free count is invalid/initial)
                                                                  ; free count calculation is needed
000033 CB 4A                        dec     dx
                                dsk_info_6:
000033 CC 39FE                      cmp     si,di                 ; same hw ?
000033 CE 7502                      jne     short dsk_info_7      ; no
                                dsk_info_5: ; 02/02/2024 - Retro DOS v5.0
000033 D0 39CA                      cmp     dx,cx                 ; same lw ?
                                dsk_info_7:
000033 D2 7268                      jb      short GotVal          ; free cluster count < last cluster number
000033 D4 31D2                      xor     dx,dx   ; 0
                                dsk_info_8:
000033 D6 31F6                      xor     si,si   ; 0
000033 D8 83E901                    sub     cx,1                  ; last cluster number - 1 = number of clusters
000033 DB 19F7                      sbb     di,si
                                    ;or     byte [es:bp+18h],1
000033 DD 26804E1801                or      byte [es:bp+DPB.FIRST_ACCESS],1 ; set first access bit 0
                                                                  ; (Update flag for FSINFO sector)
                                SCANFREE:
000033 E2 56                        push    si
000033 E3 FF36[EC0A]                push    word [CCONTENT_HW]
000033 E7 57                        push    di
000033 E8 E8F62E                    call    UNPACK
000033 EB 5F                        pop     di
000033 EC 8F06[EC0A]                pop     word [CCONTENT_HW]
000033 F0 5E                        pop     si
000033 F1 7246                      jc      short CRIT_LEAVE
000033 F3 7504                      jnz     short NOTFREECLUS
000033 F5 42                        inc     dx                    ; a free one
000033 F6 7501                      jnz     short NOTFREECLUS
000033 F8 46                        inc     si                    ; increase hw of free cluster count

                                NOTFREECLUS:
000033 F9 43                        inc     bx                    ; next cluster
000033 FA 7504                      jnz     short NOTFREECLUS2
000033 FC FF06[E80A]                inc     word [CLUSTNUM_HW]    ; increase hw of (next) cluster number
```

```
21686                                   NOTFREECLUS2:
21687 00003400 83E901                       sub     cx,1                ; decrease remain cluster count for calculation
21688 00003403 83DF00                       sbb     di,0
21689 00003406 75DA                         jnz     short SCANFREE
21690 00003408 E302                         jcxz    NOTFREECLUS3        ; calculation completed
21691 0000340A EBD6                         jmp     short SCANFREE
21692
21693                                   NOTFREECLUS3:
21694 0000340C 8B3E[E80A]                   mov     di,[CLUSTNUM_HW]
21695 00003410 83EB01                       sub     bx,1
21696 00003413 83DF00                       sbb     di,0                ; di:bx = last cluster number
21697
21698                                   ReturnVals:
21699 00003416 31C9                         xor     cx,cx
21700 00003418 83EB01                       sub     bx,1
21701 0000341B 19CF                         sbb     di,cx               ; di:bx = number of clusters
21702                                       ;mov    al,[es:bp+4]
21703 0000341D 268A4604                     mov     al,[es:bp+DPB.CLUSTER_MASK] ; spc - 1
21704 00003421 FEC0                         inc     al                  ; sectors per cluster
21705                                       ;mov    [es:bp+1Fh],dx
21706 00003423 2689561F                     mov     [es:bp+DPB.FREE_CNT],dx     ; free cluster count,lw
21707                                       ;cmp    [es:bp+0Fh],cx ; 0
21708 00003427 26394E0F                     cmp     [es:bp+DPB.FAT_SIZE],cx ; 0 ; FAT32 (16 bit FAT size = 0) ?
21709 0000342B 7507                         jnz     short ReturnVals2   ; no
21710                                       ;mov    [es:bp+21h],si
21711 0000342D 26897621                     mov     [es:bp+DPB.FREE_CNT_HW],si ; hw of free cluster count
21712
21713 00003431 E83800                       call    update_fat32_fsinfo
21714
21715                                   ReturnVals2:
21716                                       ;mov    cx,[es:bp+2]
21717 00003434 268B4E02                     mov     cx,[es:bp+DPB.SECTOR_SIZE] ; bytes per sector
21718 00003438 F8                           clc
21719
21720                                   CRIT_LEAVE:
21721                                       ;call   LCritDisk
21722                                       ;retn
21723 00003439 E9D3E4                       jmp     LCritDisk
21724
21725                                   GotVal:
21726 0000343C 89CB                         mov     bx,cx
21727 0000343E EBD6                         jmp     short ReturnVals
21728
21729                                   %endif
21730
21731                                   ; =============== S U B R O U T I N E =======================================
21732
21733                                   ; 03/02/2024 - Retro DOS v5.0
21734
21735                                   modify_spc:
21736 00003440 50                           push    ax                  ; ax = sectors per cluster
21737 00003441 52                           push    dx
21738 00003442 F7E1                         mul     cx                  ; bytes per sector
21739                                       ;cmp    dx,0
21740 00003444 21D2                         and     dx,dx
21741 00003446 7503                         jnz     short mspc_1
21742 00003448 3D0040                       cmp     ax,16384            ; 16 kilobytes (per cluster)
21743                                                                   ; ***
21744                                                                   ; actual disk size limit
21745                                                                   ; without invalidating cluster counts is
21746                                                                   ; 2 GB (512K clusters * 8 sectors per cluster)
21747                                                                   ;      (128K clusters * 32 sectors per cluster)
21748                                   mspc_1:
21749 0000344B 5A                           pop     dx
21750 0000344C 58                           pop     ax
21751 0000344D 760E                         jbe     short mspc_3        ; bytes per cluster <= 16 KB
21752                                                                         ; ***
21753                                                                         ; bytes per cluster > 16 KB
21754 0000344F 31FF                         xor     di,di               ; 0
21755 00003451 BBFEFF                       mov     bx,0FFFEh           ; (invalidated)
21756 00003454 09F6                         or      si,si               ; hw of free cluster count
21757 00003456 7404                         jz      short mspc_2        ; si = 0
21758 00003458 89FE                         mov     si,di               ; si = 0
21759 0000345A 89DA                         mov     dx,bx               ; dx = bx = 0FFFEh (invalidated)
21760                                   mspc_2:
21761 0000345C C3                           retn
21762
21763                                   mspc_3:
21764 0000345D D1E0                         shl     ax,1                ; sectors per clust = sectors per clust * 2
21765                                                                   ; (ax <= 32768) -modified spc limit-
21766 0000345F D1EF                         shr     di,1                ; cluster count = cluster count /2
21767                                                                   ; di:bx = modified value of total clusters
21768                                   mspc_4:
21769 00003461 D1DB                         rcr     bx,1
21770 00003463 D1EE                         shr     si,1                ; free clusters = free clusters / 2
21771 00003465 D1DA                         rcr     dx,1                ; si:dx = modified value of free clusters
21772
21773                                   ; ---------------------------------------------------------------------------
21774
21775                                       ; 03/02/2024
21776                                       ; PCDOS 7.1 IBMDOS.COM - DOSCODE:7431h
21777
21778                                   modify_cluster_count:
21779 00003467 09FF                         or      di,di               ; hw of cluster count
21780 00003469 75D5                         jnz     short modify_spc
21781 0000346B C3                           retn
21782
21783                                   ; =============== S U B R O U T I N E =======================================
21784
21785                                   ; write FSINFO sector onto disk
21786
21787                                   ; 03/02/2024 - Retro DOS v5.0
21788                                   ; --------------------------------
21789                                   ; FAT32 FSInfo Sector Structure
21790                                   ; --------------------------------
21791                                   ; ref: Microsoft FAT32 File System Specification (2000)
21792
21793                                   struc FSINFO                ; Offset ;
21794 00000000 ????????               .LeadSig:   resb 4    ; 0           ; Value 0x41615252. Lead Signature.
21795 00000004 <res 1E0h>              .Reserved1: resb 480  ; 4           ; Reserved. Must be 0. Never be used.
21796 000001E4 ????????               .StrucSig:  resb 4    ; 484         ; Value 0x61417272. Fields Signature.
21797 000001E8 ????????               .Free_Count:       resb 4   ; 488         ; Last known free cluster count. (*)
21798 000001EC ????????               .Nxt_Free:  resb 4    ; 492         ; Start clus for free clus srch. (**)
21799 000001F0 <res Ch>               .Reserved2: resb 12   ; 496         ; Reserved. Must be 0. Never be used.
21800 000001FC ????????               .TrailSig:  resb 4    ; 508         ; Value 0xAA550000. Trail Signature.
21801                                  .size:
21802                                  endstruc
21803
21804                                   ; (*) If the value is 0xFFFFFFFF, then the free count is unknown
21805                                   ;     and must be computed.
21806                                   ; (**) If the value is 0xFFFFFFFF, then free cluster search must be started
21807                                   ;     from cluster 2.
21808                                   ; Lead Signature, Fields (Structure) Signature and Trail Signature
21809                                   ; are used to validate FSInfo sector.
```

```
21810
21811                                   ; --------------------------------
21812
21813                                       ; 03/02/2024 - Retro DOS v5.0
21814                                       ; PCDOS 7.1 IBMDOS.COM - DOSCODE:7436h
21815                                       ; (Windows ME IO.SYS - BIOSCODE:7227h)
21816
21817                                   update_fat32_fsinfo:
21818 0000346C 51                         push  cx
21819 0000346D 52                         push  dx
21820 0000346E 31C9                       xor   cx,cx
21821                                     ;mov   dx,[es:bp+25h]
21822 00003470 268B5625                   mov   dx,[es:bp+DPB.FSINFO_SECTOR]
21823                                     ;cmp   [es:bp+0Fh],cx
21824 00003474 26394E0F                   cmp   [es:bp+DPB.FAT_SIZE],cx ; 0
21825                                                   ; (16bit FAT size field = 0 for FAT32 fs)
21826 00003478 7505                       jne   short u_fat32_inf_1
21827 0000347A 83FAFF                     cmp   dx,0FFFFh ; -1
21828 0000347D 7508                       jne   short u_fat32_inf_2
21829
21830                                   u_fat32_inf_1:
21831 0000347F 5A                         pop   dx
21832 00003480 59                         pop   cx
21833                                     ;and   byte [es:bp+18h],0F4h
21834 00003481 26806618F4                 and   byte [es:bp+DPB.FIRST_ACCESS],0F4h ; clear bit 0,1 and 3
21835                                                   ; bit 0 - FSINFO update (dirty) bit
21836                                                   ; bit 1 - BPB_RootClus update bit
21837                                                   ; bit 3 - BPB_ExtFlags update bit
21838 00003486 C3                         retn
21839
21840                                   u_fat32_inf_2:
21841 00003487 50                         push  ax
21842 00003488 53                         push  bx
21843 00003489 57                         push  di
21844 0000348A 56                         push  si
21845 0000348B 1E                         push  ds
21846 0000348C 36803E[7900]00             cmp   byte [ss:BuffInHMA],0 ; is buffers in HMA?
21847 00003492 740A                       jz    short u_fat32_inf_3   ; no
21848 00003494 36C53E[7A00]               lds   di,[ss:LoMemBuff]     ; read it into scratch buffer
21849                                     ;sub   di,24
21850 00003499 83EF18                     sub   di,BUFINSIZ           ; space for buffer header
21851                                                                 ; (buffer header size = 24)
21852                                     ;clc
21853 0000349C EB0E                       jmp   short u_fat32_inf_4
21854
21855                                   u_fat32_inf_3:
21856 0000349E 06                         push  es
21857 0000349F 55                         push  bp
21858 000034A0 E81634                     call  GETCURHEAD            ; ds:di = first buffer in queue
21859 000034A3 52                         push  dx
21860 000034A4 E87536                     call  BUFWRITE              ; BufWrite writes a buffer to the disk,
21861                                                                 ;  if it's dirty.
21862 000034A7 5A                         pop   dx
21863 000034A8 5D                         pop   bp
21864 000034A9 07                         pop   es
21865                                   ;u_fat32_inf_4:
21866 000034AA 726A                       jc    short u_fat32_inf_5
21867                                   u_fat32_inf_4:
21868 000034AC 31C9                       xor   cx,cx
21869                                     ;lea   bx,[di+24]
21870 000034AE 8D5D18                     lea   bx,[di+BUFINSIZ]      ; buffer data address
21871                                     ;mov   byte [ss:ALLOWED],18h
21872 000034B1 36C606[4B03]18             mov   byte [ss:ALLOWED],Allowed_FAIL+Allowed_RETRY
21873 000034B7 36890E[0706]               mov   [ss:HIGH_SECTOR],cx ; 0
21874 000034BC 41                         inc   cx                    ; cx = sector count = 1
21875                                                                 ; es:bp = DPB
21876 000034BD 53                         push  bx                    ; ds:bx = buffer (data) address
21877 000034BE 52                         push  dx                    ; HIGH_SECTOR:dx = disk sector address
21878 000034BF E82C0B                     call  DREAD                 ; read fs info sector
21879 000034C2 5A                         pop   dx
21880 000034C3 5B                         pop   bx
21881 000034C4 7250                       jc    short u_fat32_inf_5
21882
21883                                     ;cmp   word [bx+FSINFO.LeadSig],5252h
21884 000034C6 813F5252                   cmp   word [bx],5252h               ; 'RR'
21885 000034CA 754A                       jne   short u_fat32_inf_5
21886                                     ;cmp   word [bx+2],4161h      ; 'aA' ; (NASM syntax)
21887 000034CC 817F026141                 cmp   word [bx+FSINFO.LeadSig+2],4161h
21888 000034D1 7543                       jne   short u_fat32_inf_5
21889
21890                                     ;cmp   word [bx+1E4h],7272h  ; 'rr' at offset 484
21891 000034D3 81BFE4017272               cmp   word [bx+FSINFO.StrucSig],7272h
21892 000034D9 753B                       jne   short u_fat32_inf_5
21893
21894                                     ;cmp   word [bx+1E6h],6141h  ; 'Aa' at offset 486
21895 000034DB 81BFE6014161               cmp   word [bx+FSINFO.StrucSig+2],6141h
21896 000034E1 7533                       jne   short u_fat32_inf_5
21897
21898                                     ;cmp   word [bx+1FEh],0AA55h ; boot signature at offset 510
21899 000034E3 81BFFE0155AA               cmp   word [bx+FSINFO.TrailSig+2],0AA55h
21900 000034E9 752B                       jne   short u_fat32_inf_5
21901
21902                                     ;mov   ax,[es:bp+1Fh]
21903 000034EB 268B461F                   mov   ax,[es:bp+DPB.FREE_CNT]
21904                                     ;mov   [bx+1E8h],ax
21905 000034EF 8987E801                   mov   [bx+FSINFO.Free_Count],ax ; at offset 488
21906                                     ;mov   ax,[es:bp+21h]
21907 000034F3 268B4621                   mov   ax,[es:bp+DPB.FREE_CNT+2]
21908                                     ;mov   [bx+1EAh],ax
21909 000034F7 8987EA01                   mov   [bx+FSINFO.Free_Count+2],ax
21910
21911                                     ;mov   ax,[es:bp+39h]
21912 000034FB 268B4639                   mov   ax,[es:bp+DPB.FAT32_NXTFREE]
21913                                     ;mov   [bx+1ECh],ax
21914 000034FF 8987EC01                   mov   [bx+FSINFO.Nxt_Free],ax       ; at offset 492
21915                                     ;mov   ax,[es:bp+3Bh]
21916 00003503 268B463B                   mov   ax,[es:bp+DPB.FAT32_NXTFREE+2]
21917                                     ;mov   [bx+1EEh],ax
21918 00003507 8987EE01                   mov   [bx+FSINFO.Nxt_Free+2],ax
21919
21920 0000350B 31C9                       xor   cx,cx
21921 0000350D 36890E[0706]               mov   [ss:HIGH_SECTOR],cx ; 0
21922 00003512 41                         inc   cx      ; 1
21923 00003513 E83D0B                     call  DWRITE
21924
21925                                   u_fat32_inf_5:
21926 00003516 1F                         pop   ds
21927 00003517 5E                         pop   si
21928 00003518 5F                         pop   di
21929 00003519 5B                         pop   bx
21930 0000351A 58                         pop   ax
21931 0000351B E961FF                     jmp   u_fat32_inf_1
21932
21933                                   ;================================================================================
```

```
                                ; ISEARCH.ASM, MSDOS 6.0, 1991
                                ;==========================================================================
                                ; 22/07/2018 - Retro DOS v3.0

                                ;   TITLE   DOS_SEARCH - Internal SEARCH calls for MS-DOS
                                ;   NAME    DOS_SEARCH

                                ;** Low level routines for doing local and NET directory searches
                                ;
                                ;   DOS_SEARCH_FIRST
                                ;   DOS_SEARCH_NEXT
                                ;   RENAME_NEXT
                                ;
                                ;   Revision history:
                                ;
                                ;       Created: ARR 30 March 1983
                                ;       A000       version 4.00  Jan. 1988
                                ;       A001       PTM 3564 -- search for fastopen

                                ;Installed = TRUE

                                ;--------------------------------------------------------------------------
                                ;
                                ; Procedure Name : DOS_SEARCH_FIRST
                                ;
                                ; Inputs:
                                ;   [WFP_START] Points to WFP string ("d:/" must be first 3 chars, NUL
                                ;           terminated)
                                ;   [CURR_DIR_END] Points to end of Current dir part of string
                                ;           ( = -1 if current dir not involved, else
                                ;            Points to first char after last "/" of current dir part)
                                ;   [THISCDS] Points to CDS being used
                                ;           (Low word = -1 if NUL CDS (Net direct request))
                                ;   [SATTRIB] Is attribute of search, determines what files can be found
                                ;   [DMAADD] Points to 53 byte buffer
                                ; Function:
                                ;   Initiate a search for the given file spec
                                ; Outputs:
                                ;   CARRY CLEAR
                                ;       The 53 bytes ot DMAADD are filled in as follows:
                                ;
                                ;   LOCAL
                                ;       Drive Byte (A=1, B=2, ...) High bit clear
                                ;           NEVER STORE DRIVE BYTE AFTER  found_it
                                ;       11 byte search name with Meta chars in it
                                ;       Search Attribute Byte, attribute of search
                                ;       WORD LastEnt value
                                ;       WORD DirStart
                                ;       4 byte pad
                                ;       32 bytes of the directory entry found
                                ;   NET
                                ;       21 bytes First byte has high bit set
                                ;       32 bytes of the directory entry found
                                ;
                                ;   CARRY SET
                                ;       AX = error code
                                ;           error_no_more_files
                                ;                   No match for this file
                                ;           error_path_not_found
                                ;                   Bad path (not in curr dir part if present)
                                ;           error_bad_curr_dir
                                ;                   Bad path in current directory part of path
                                ; DS preserved, others destroyed
                                ;--------------------------------------------------------------------------

                                ; 24/01/2024
                                %if 1
                                    ; 17/05/2019 - Retro DOS v4.0
                                GET_FAST_SEARCH:
                                    ; 22/07/2018
                                    ; MSDOS 6.0
                                    ; 17/12/2022
0000351E 36800E[1206]04             OR      byte [ss:DOS34_FLAG+1],(SEARCH_FASTOPEN>>8)  ; 04h
                                    ;OR     word [ss:DOS34_FLAG],SEARCH_FASTOPEN  ; 400h
                                                                    ;FO.trigger fastopen ;AN000;
                                    ;call   DOS_SEARCH_FIRST
                                    ;retn
                                    ; 24/01/2024
                                    ; 17/12/2022
                                    ;jmp    DOS_SEARCH_FIRST
                                %endif

                                    ; 14/11/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
                                    ; DOSCODE:6C22h (MSDOS 5.0, MSDOS.SYS)

                                    ; 03/02/2024 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMDOS.COM)
                                    ; DOSCODE:74E9h (PCDOS 7.1, IBMDOS.COM)

                                DOS_SEARCH_FIRST:
                                    ; IBMDOS.COM (MSDOS 3.3 kernel) - Offset 3826h

00003524 C43E[A205]                 LES     DI,[THISCDS]
00003528 83FFFF                     CMP     DI,-1
0000352B 7506                       JNZ     short TEST_RE_NET2

                                ;IF NOT Installed
                                ;   transfer NET_SEQ_SEARCH_FIRST
                                ;ELSE
                                    ;mov    ax,1119h
0000352D B81911                     MOV     AX,(MultNET<<8)|25
00003530 CD2F                       INT     2Fh
00003532 C3                         retn
                                ;ENDIF

                                TEST_RE_NET2:
                                    ;test   word [es:di+43h],8000h
                                    ; 17/12/2022
                                    ;test   byte [es:di+44h],80h
                                    ; 28/12/2022
00003533 26F6454480                 test    byte [ES:DI+curdir.flags+1],curdir_isnet>>8
                                    ;TEST   word [ES:DI+curdir.flags],curdir_isnet
00003538 7406                       JZ      short LOCAL_SEARCH_FIRST

                                ;IF NOT Installed
                                ;   transfer NET_SEARCH_FIRST
                                ;ELSE
                                    ;mov    ax,111Bh
0000353A B81B11                     MOV     AX,(MultNET<<8)|27
0000353D CD2F                       INT     2FH
0000353F C3                         retn
                                ;ENDIF
                                    ; 18/05/2019 - Retro DOS v4.0
                                LOCAL_SEARCH_FIRST:
00003540 E89FE3                     call    ECritDisk
```

```
                                    ; MSDOS 6.0
                                    ;;test  word [DOS34_FLAG],400h
                                    ; 17/12/2022
                                    ;test   byte [DOS34_FLAG+1],04h
00003543 F606[1206]04               test    byte [DOS34_FLAG+1],(SEARCH_FASTOPEN>>8)
                                    ;TEST   word [DOS34_FLAG],SEARCH_FASTOPEN ;AN000;
00003548 7405                       JZ      short NOFN                  ;AN000;
                                    ;or     byte [FastOpenFlg],1
0000354A 800E[4611]01               OR      byte [FastOpenFlg],FastOpen_Set     ;AN000;
                                NOFN:                                           ;AN000;
0000354F C606[4C03]01               MOV     byte [NoSetDir],1     ; if we find a dir, don't change to it

                                ; 03/02/2024
                                %if 0
                                    ; MSDOS 6.0
                                    CALL    CHECK_QUESTION        ;AN000;;FO. is '?' in path
                                    JNC     short norm_GETPATH    ;AN000;;FO. no
                                %else
                                    ; 03/02/2024
00003554 16                         push    ss
00003555 1F                         pop     ds                    ;AN000;;FO. ds:si -> final path
00003556 8B36[B205]                 mov     si,[WFP_START]        ;AN000;;FO.
                                getnext:                          ;AN000;
0000355A AC                         lodsb                         ;AN000;;FO. get char
0000355B 08C0                       or      al,al                 ;AN000;;FO. is it null
0000355D 7409                       jz      short NO_Question     ;AN000;;FO. yes
0000355F 3C3F                       cmp     al,'?'                  ;AN000;;FO. is '?'
00003561 75F7                       jne     short getnext         ;AN000;;FO. no
                                %endif
00003563 8026[4611]80               ;and    byte [FastOpenFlg],80h
                                    AND     byte [FastOpenFlg],Fast_yes ;AN000;;FO. reset fastopen
                                NO_Question:       ; 03/02/2024
                                norm_GETPATH:
00003568 E8A815                     call    GETPATH
                                    ; BX = offset NAME1
                                ;_getdone:
0000356B 7318                       JNC     short find_check_dev
0000356D 7511                       JNZ     short bad_path3
0000356F 08C9                       OR      CL,CL
00003571 740D                       JZ      short bad_path3
                                find_no_more:
                                    ;mov    ax,12h
00003573 B81200                     MOV     AX,error_no_more_files
                                BadBye:
                                    ; MSDOS 6.0
00003576 368026[4611]80             AND     byte [SS:FastOpenFlg],Fast_yes  ;AN000;;FO. reset fastopen

0000357C F9                         STC
                                    ;call   LCritDisk
                                    ;retn
                                    ; 18/12/2022
0000357D E98FE3                     jmp     LCritDisk

                                bad_path3:
                                    ;mov    ax,3
00003580 B80300                     MOV     AX,error_path_not_found
00003583 EBF1                       JMP     short BadBye

                                find_check_dev:
00003585 08E4                       OR      AH,AH
00003587 790A                       JNS     short found_entry
00003589 C706[4803]FFFF             MOV     word [LASTENT],-1     ; Cause DOS_SEARCH_NEXT to fail
0000358F FE06[7005]                 INC     byte [FOUND_DEV]      ; Tell DOS_RENAME we found a device
                                found_entry:

                                ; We set the physical drive byte here Instead of after found_it; Doing
                                ; a search-next may not have wfp_start set correctly

00003593 C43E[2C03]                 LES     DI,[DMAADD]
00003597 8B36[B205]                 MOV     SI,[WFP_START]        ; get pointer to beginning
0000359B AC                         LODSB
0000359C 2C40                       SUB     AL,'A'-1                ; logical drive
0000359E AA                         STOSB                         ; High bit not set (local)
                                found_it:
0000359F C43E[2C03]                 LES     DI,[DMAADD]
000035A3 47                         INC     DI

                                    ; MSDOS 6.0
000035A4 1E                         PUSH    DS                              ;FO.;AN001; save ds
                                    ;test   byte [FastOpenFlg],10h
000035A5 F606[4611]10               TEST    byte [FastOpenFlg],Set_For_Search ;FO.;AN001; from fastopen
000035AA 7408                       JZ      short notfast                   ;FO.;AN001;
000035AC 89DE                       MOV     SI,BX                           ;FO.;AN001;
000035AE 8E1E[E405]                 MOV     DS,[CURBUF+2]                   ;FO.;AN001;
000035B2 EB03                       JMP     SHORT movmov                    ;FO.;AN001;

                                notfast:
000035B4 BE[4B05]                   MOV     SI,NAME1              ; find_buf 2 = formatted name
                                movmov:
                                ; Special E5 code
000035B7 A4                         MOVSB
000035B8 26807DFF05                 CMP     BYTE [ES:DI-1],5
000035BD 7505                       JNZ     short NOTKANJB
000035BF 26C645FFE5                 MOV     BYTE [ES:DI-1],0E5H
                                NOTKANJB:
                                    ;MOV    CX,10
                                    ;REP    MOVSB
                                    ; 03/02/2024
000035C4 B90500                     mov     cx,5
000035C7 F3A5                       rep     movsw

                                    ; 08/09/2018
000035C9 1F                         POP     DS                      ;FO.;AN001; restore ds

000035CA A0[6B05]                   MOV     AL,[ATTRIB]
000035CD AA                         STOSB
000035CE 50                         PUSH    AX                      ; Save AH device info
000035CF A1[4803]                   MOV     AX,[LASTENT]
000035D2 AB                         STOSW
000035D3 A1[C205]                   MOV     AX,[DIRSTART]
000035D6 AB                         STOSW

                                    ; 03/02/2024 - Retro DOS v5.0
                                    ; PCDOS 7.1 IBMDOS.COM
                                    ;;;
000035D7 A1[DC0A]                   MOV     AX,[DIRSTART_HW]
000035DA AB                         STOSW
000035DB 83C702                     add     di,2
                                ; 4 bytes of 21 byte cont structure left for NET stuff
                                    ;ADD    DI,4
                                    ;;;

000035DE 58                         POP     AX                      ; Recover AH device info
000035DF 08E4                       OR      AH,AH
```

```
22182 000035E1 781B                     JS      short DOSREL            ; Device entry is DOSGROUP relative
22183 000035E3 833E[E205]FF             CMP     WORD [CURBUF],-1
22184 000035E8 7510                     JNZ     short OKSTORE
22185
22186                                   ; MSDOS 6.0
22187 000035EA F606[4611]10             TEST    byte [FastOpenFlg],Set_For_Search
22188                                                           ;AN000;;FO. from fastopen and is good
22189 000035EF 7509                     JNZ     short OKSTORE           ;AN000;;FO.
22190
22191                                   ; The user has specified the root directory itself, rather than some
22192                                   ; contents of it. We can't "find" that.
22193
22194 000035F1 26C745F8FFFF              MOV     WORD [ES:DI-8],-1       ; Cause DOS_SEARCH_NEXT to fail by
22195                                                           ;   stuffing a -1 at Lastent
22196 000035F7 E979FF                   JMP     find_no_more
22197
22198                                 OKSTORE:
22199 000035FA 8E1E[E405]               MOV     DS,[CURBUF+2]
22200                                 DOSREL:
22201                                   ; BX = offset NAME1 (from GETPATH)
22202 000035FE 89DE                     MOV     SI,BX                   ; SI-> start of entry
22203
22204                                 ; NOTE: DOS_RENAME depends on BX not being altered after this point
22205
22206                                   ;;mov   cx,32
22207                                   ;MOV    CX,dir_entry.size
22208                                   ; 03/02/2024
22209 00003600 B91000                   mov     cx,dir_entry.size>>1
22210                                 ;;;;; 7/29/86
22211 00003603 89F8                     MOV     AX,DI                   ; save the 1st byte addr
22212                                   ;REP    MOVSB
22213 00003605 F3A5                     rep     movsw
22214                                   ;
22215 00003607 89C7                     MOV     DI,AX                   ; restore 1st byte addr
22216 00003609 26803D05                 CMP     BYTE [ES:DI],05H        ; special char check
22217 0000360D 7504                     JNZ     short NO05
22218 0000360F 26C605E5                 MOV     BYTE [ES:DI],0E5H       ; convert it back to E5
22219                                 NO05:
22220
22221                                 ;;;;; 7/29/86
22222
22223                                 ;hkn; FastOpenflg is in DOSDATA use SS
22224                                     ; 16/12/2022
22225                                     ; 14/11/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
22226                                     ; MSDOS 6.0
22227                                     ;AND    byte [SS:FastOpenFlg],Fast_yes ;AN000;;FO. reset fastopen
22228                                     ; 18/05/2019 - Retro DOS v4.0
22229 00003613 16                       push    ss
22230 00003614 1F                       pop     ds
22231                                   ; 16/12/2022
22232 00003615 8026[4611]80             AND     byte [FastOpenFlg],Fast_yes ; 80h
22233
22234                                 ;hkn; SS is DOSDATA
22235                                     ;push   ss
22236                                     ;pop    ds
22237
22238                                     ; 27/06/2024
22239                                     ; cf=0
22240                                     ;CLC
22241
22242                                     ;call   LCritDisk
22243                                     ;retn
22244                                     ; 16/12/2022
22245 0000361A E9F2E2                   jmp     LCritDisk
22246
22247                                 ;BREAK <DOS_SEARCH_NEXT - scan for subsequent matches>
22248                                 ;--------------------------------------------------------------------------
22249                                 ;
22250                                 ; Procedure Name : DOS_SEARCH_NEXT
22251                                 ;
22252                                 ; Inputs:
22253                                 ;   [DMAADD] Points to 53 byte buffer returned by DOS_SEARCH_FIRST
22254                                 ;       (only first 21 bytes must have valid information)
22255                                 ; Function:
22256                                 ;   Look for subsequent matches
22257                                 ; Outputs:
22258                                 ;   CARRY CLEAR
22259                                 ;       The 53 bytes at DMAADD are updated for next call
22260                                 ;           (see DOS_SEARCH_FIRST)
22261                                 ;   CARRY SET
22262                                 ;       AX = error code
22263                                 ;           error_no_more_files
22264                                 ;                   No more files to find
22265                                 ; DS preserved, others destroyed
22266                                 ;--------------------------------------------------------------------------
22267
22268                                 ;hkn; called from search.asm. DS already set up at this point.
22269
22270                                     ; 03/02/2024 - Retro DOS v5.0
22271                                     ; PCDOS 7.1 IBMDOS.COM - DOSCODE:75ECh
22272
22273                                 DOS_SEARCH_NEXT:
22274 0000361D C43E[2C03]                 LES     DI,[DMAADD]     ; 24/01/2024 (Retro DOS v5-v4)
22275 00003621 268A05                   MOV     AL,[ES:DI]
22276 00003624 A880                     TEST    AL,80H                  ; Test for NET
22277 00003626 7406                     JZ      short LOCAL_SEARCH_NEXT
22278                                 ;IF NOT Installed
22279                                 ;   transfer NET_SEARCH_NEXT
22280                                 ;ELSE
22281                                     ;mov    ax,111Ch
22282 00003628 B81C11                   MOV     AX,(MultNET<<8)|28
22283 0000362B CD2F                     INT     2FH  ; Multiplex - NETWORK REDIRECTOR - FINDNEXT
22284                                                ; SS = DS = DOS CS, [DTA] = 21-byte findfirst search data
22285                                                ; Return: CF set on error, AX = DOS error code
22286                                                ; CF clear if successful
22287 0000362D C3                       retn
22288                                 ;ENDIF
22289
22290                                 LOCAL_SEARCH_NEXT:
22291                                     ;AL is drive A=1
22292                                     ;mov    byte [EXTERR_LOCUS],2
22293 0000362E C606[2303]02             MOV     byte [EXTERR_LOCUS],errLOC_Disk
22294 00003633 E8ACE2                   call    ECritDisk
22295
22296                                 ;hkn; DummyCDS is in DOSDATA
22297 00003636 C706[A205][F304]         MOV     word [THISCDS],DUMMYCDS
22298                                 ;hkn; Segment address is DOSDATA - use ds
22299                                 ;hkn;       MOV     WORD [THISCDS+2],CS
22300 0000363C 8C1E[A405]               mov     [THISCDS+2],DS
22301
22302 00003640 0440                     ADD     AL,'A'-1
22303 00003642 E89044                   call    InitCDS
22304
22305                                 ;   call    GETTHISDRV              ; Set CDS pointer
```

```
                                    JC      short No_files          ; Bogus drive letter
00003645 7253                       LES     DI,[THISCDS]            ; Get CDS pointer
00003647 C43E[A205]                 ;les    bp,[es:di+45h]
                                    LES     BP,[ES:DI+curdir.devptr] ; Get DPB pointer
0000364B 26C46D45                   call    GOTDPB                  ; [THISDPB] = ES:BP
0000364F E813D0
                                    ; 16/12/2022
00003652 268A4600                   mov     al,[ES:BP]
                                    ; 14/11/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
                                    ;mov    AL,[ES:BP+DPB.DRIVE] ; mov al,[ES:BP+0]
00003656 A2[7605]                   mov     [THISDRV],AL
                                    ;mov    word [CREATING],0E500h
00003659 C706[7E05]00E5             MOV     WORD [CREATING],(DIRFREE*256)+0
0000365F C606[4C03]01               MOV     byte [NoSetDir],1       ; if we find a dir, don't change to it
00003664 C536[2C03]                 LDS     SI,[DMAADD]
00003668 AC                         LODSB                           ; Drive Byte

                                    ;entry  RENAME_NEXT             ; Entry used by DOS_RENAME
                                RENAME_NEXT:
                                    ;context ES
00003669 16                         push    ss
0000366A 07                         pop     es                      ; THIS BLOWS ES:BP POINTER TO DPB

                                ;hkn; NAME1 is in DOSDATA
0000366B BF[4B05]                   MOV     DI,NAME1

0000366E B90B00                     MOV     CX,11
00003671 F3A4                       REP     MOVSB                   ; Search name
00003673 AC                         LODSB                           ; Attribute

                                ;hkn; SS override
00003674 36A2[6B05]                 MOV     [SS:ATTRIB],AL
00003678 AD                         LODSW                           ; LastEnt
00003679 09C0                       OR      AX,AX
                                    ; 03/02/2024
                                    ;JNS    short cont_load
0000367B 781D                       js      short No_files
                                ;No_files:
                                    ;JMP    find_no_more

                                cont_load:
0000367D 50                         PUSH    AX                      ; Save LastEnt
0000367E AD                         LODSW                           ; DirStart
0000367F 89C3                       MOV     BX,AX

                                    ;;;
                                    ; 03/02/2024 - Retro DOS v5.0
                                    ; (PCDOS 7.1 IBMDOS.COM)
00003681 AD                         lodsw                           ; DIRSTART_HW
                                    ;;;

                                ;hkn; SS is DOSDATA
                                    ;context DS
00003682 16                         push    ss
00003683 1F                         pop     ds
00003684 C42E[8A05]                 LES     BP,[THISDPB]            ; Recover ES:BP

                                    ;;;
                                    ; 03/02/2024 - Retro DOS v5.0
                                    ; (PCDOS 7.1 IBMDOS.COM)

                                    ;cmp    word [es:bp+0Fh],0
00003688 26837E0F00                 cmp     word [es:bp+DPB.FAT_SIZE],0
0000368D 7402                       jz      short cont_load2 ; FAT32 fs
0000368F 31C0                       xor     ax,ax ; 0
                                cont_load2:
00003691 A3[EE0A]                   mov     [ROOTCLUST_HW],ax       ; 0 or DIRSTART_HW
                                    ;;;

                                    ;invoke SetDirSrch
00003694 E8C812                     call    SETDIRSRCH
00003697 7304                       JNC     short SEARCH_GOON
00003699 58                         POP     AX                      ; Clean stack
                                    ;JMP    short No_files
                                    ; 03/02/2024
                                No_files:
0000369A E9D6FE                     JMP     find_no_more

                                SEARCH_GOON:
0000369D E81317                     call    STARTSRCH
000036A0 58                         POP     AX                      ; Restore LastEnt
000036A1 E8FE11                     call    GETENT
000036A4 72F4                       JC      short No_files
000036A6 E8F210                     call    NEXTENT
000036A9 72EF                       JC      short No_files
000036AB 30E4                       XOR     AH,AH                   ; If Search_Next, can't be a DEV
000036AD E9EFFE                     JMP     found_it ; 10/08/2018

                                ; MSDOS 6.0
                                ;--------------------------------------------------------------------------
                                ;
                                ; Procedure Name : CHECK_QUESTION
                                ;
                                ; Input: [WFP_START]= pointer to final path
                                ; Function: check '?' char
                                ; Output: carry clear, if no '?'
                                ;    carry set, if '?' exists
                                ;--------------------------------------------------------------------------

                                ; 03/02/2024
                                %if 0
                                    ; 07/12/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
                                CHECK_QUESTION:
                                ;hkn;       wfp_start is in DOSDATA;hkn;  MOV     WORD PTR ThisCDS+2,CS
                                ;hkn;       PUSH    CS                    ;AN000;;FO.
                                    push    ss
                                    POP     DS                    ;AN000;;FO. ds:si -> final path
                                    ; 16/12/2022
                                    ; 07/12/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
                                    MOV     SI,[WFP_START]        ;AN000;;FO.
                                    ;mov    si,[ss:WFP_START]
                                getnext:                          ;AN000;
                                    LODSB                         ;AN000;;FO. get char
                                    OR      AL,AL                 ;AN000;;FO. is it null
                                    JZ      short NO_Question     ;AN000;;FO. yes
                                    CMP     AL,'?'                  ;AN000;;FO. is '?'
                                    JNZ     short getnext         ;AN000;;FO. no
                                    STC                           ;AN000;;FO.
                                NO_Question:                              ;AN000;
                                    retn                          ;AN000;;FO.
                                %endif

                                ;==========================================================================
```

```
                        ; ABORT.ASM, MSDOS 6.0, 1991
                        ;==========================================================================
                        ; 23/07/2018 - Retro DOS v3.0
                        ; 18/05/2019 - Retro DOS v4.0

                        ;**
                        ;
                        ; Internal Abort call closes all handles and FCBs associated with a process.
                        ;  If process has NET resources a close all is sent out over the net.
                        ;
                        ;   DOS_ABORT
                        ;
                        ;   Modification history:
                        ;
                        ;       Created: ARR 30 March 1983
                        ;
                        ;   M038    SR      10/16/90        Free SFT with the PSP of the process
                        ;                           being terminated only if it is busy.
                        ;

                        ;Break   <DOS_ABORT -- CLOSE all files for process>
                        ;--------------------------------------------------------------------------
                        ;
                        ; Procedure Name : DOS_ABORT
                        ;
                        ; Inputs:
                        ;       [CurrentPDB] set to PID of process aborting
                        ; Function:
                        ;       Close all files and free all SFTs for this PID
                        ; Returns:
                        ;       None
                        ; All destroyed except stack
                        ;--------------------------------------------------------------------------

                            ; 03/02/2024 - Retro DOS v5.0
                            ; PCDOS 7.1 IBMDOS.COM - DOSCODE:7685h
                            ; (Win ME IO.SY_ - BIOSCODE:74F4h)

                        DOS_ABORT:
000036B0 368E06[3003]       MOV     ES,[SS:CurrentPDB]    ; SS override
000036B5 268B0E3200         MOV     CX,[ES:PDB.JFN_Length]  ; Number of JFNs
                        reset_free_jfn:
000036BA 89CB               MOV     BX,CX
000036BC 51                 PUSH    CX
000036BD 4B                 DEC     BX                      ; get jfn (start with last one)

000036BE E8AC40             CALL    _$CLOSE
000036C1 59                 POP     CX
000036C2 E2F6               LOOP    reset_free_jfn          ; and do 'em all

                        ; Note: We do need to explicitly close FCBs. Reasons are as follows: If we
                        ; are running in the no-sharing no-network environment, we are simulating the
                        ; 2.0 world and thus if the user doesn't close the file, that is his problem
                        ; BUT... the cache remains in a state with garbage that may be reused by the
                        ; next process. We scan the set and blast the ref counts of the FCBs we own.
                        ;
                        ; If sharing is loaded, then the following call to close process will
                        ; correctly close all FCBs. We will then need to walk the list AFTER here.
                        ;
                        ; Finally, the following call to NET_Abort will cause an EOP to be sent to all
                        ; known network resources. These resources are then responsible for cleaning
                        ; up after this process.
                        ;
                        ; Sleazy, eh?

                            ;context DS                 ; SS is DOSDATA
000036C4 16                 push    ss
000036C5 1F                 pop     ds  ; 09/09/2018

                            ;CallInstall Net_Abort, MultNET, 29
000036C6 B81D11             mov     ax,111Dh
000036C9 CD2F               int     2Fh     ; Multiplex - NETWORK REDIRECTOR
                                            ;           - CLOSE ALL REMOTE FILES FOR PROCESS
                                            ; DS???, SS = DOS CS
                        ;if installed
000036CB FF1E[A000]         call    far [JShare+(4*4)]    ; 4 = MFTCloseP
                        ;else
                        ;   call    MFTCloseP
                        ;endif

                        ; Scan the FCB cache for guys that belong to this process and zap their ref
                        ; counts.
                                                        ; SS override
000036CF 36C43E[4000]       les     di,[ss:SFTFCB]      ; grab the pointer to the table

                            ;;;
                            ; 03/02/2024 - Retro DOS v5.0
                            ; PCDOS 7.1 IBMDOS.COM
                        SFTFCB_check:
000036D4 8CC1               mov     cx,es
000036D6 09F9               or      cx,di
000036D8 E325               jcxz    FCBScanDone
000036DA 57                 push    di
                        SFTFCB_OK:
                            ;;;
                            ;mov    cx,[es:di+4]
000036DB 268B4D04           mov     cx,[es:di+SFT.SFCount]
                            ;jcxz   FCBScanDone
                            ; 27/06/2024
                            ;;;
000036DF E315               jcxz    SFTFCB_check2
                            ;;;
                            ;lea    di,[di+6]
000036E1 8D7D06             LEA     DI,[DI+SFT.SFTable]   ; point at table
000036E4 36A1[3C03]         mov     ax,[SS:PROC_ID]             ; SS override
                        FCBTest:
                            ;cmp    [es:di+31h],ax
000036E8 26394531           cmp     [es:di+SF_ENTRY.sf_PID],ax ; is this one of ours
000036EC 7503               jnz     short FCBNext         ; no, skip it

                        ; 03/02/2024 - Retro DOS v5.0
                        %if 0
                            mov     word [es:di],0
                            ;mov    word [es:di+SF_ENTRY.sf_ref_count],0 ; yes, blast ref count
                        %else
                            ; 03/02/2024
                            ; PCDOS 7.1 IBMDOS.COM
000036EE E8FA13             call    SFT_FREE
                        %endif

                        FCBNext:
000036F1 83C73B             add     di,SF_ENTRY.size ; 59 (for MSDOS 6.0)
000036F4 E2F2               loop    FCBTest
```

```
                                    ; 27/06/2024 (PCDOS 7.1 IBMDOS.COM)
                                    ;;;
                                SFTFCB_check2:
000036F6 5F                         pop     di
000036F7 26C43D                     les     di,[es:di]
000036FA 83FFFF                     cmp     di,0FFFFh
000036FD 75D5                       jne     short SFTFCB_check
                                    ;;;

                                FCBScanDone:

                                ; Walk the SFT to eliminate all busy SFT's for this process.

000036FF 31DB                       XOR     BX,BX
                                Scan:
00003701 53                         push    bx
00003702 E8E43F                     call    SFFromSFN
00003705 5B                         pop     bx
                                    ;jnc    short Scan1
                                    ;retn

                                    ; 18/12/2022
                                    ;jc     short NO_Question ; retn
                                    ; 03/02/2024
00003706 7232                       jc      short RET2

                                ;M038
                                ; Do what the comment above says, check for busy state

                                Scan1:
                                    ;cmp    word [es:di],0
                                    ;jz     short scan_next  ; MSDOS 3.3
                                    ; MSDOS 6.0
00003708 26833DFF                   cmp     word [es:di],sf_busy ; -1
                                    ;cmp    word [es:di+SF_ENTRY.sf_ref_count],sf_busy
                                                                 ; Is Sft busy? ;M038
0000370C 7517                       jnz     short scan_next ; no
                                ;
                                ; we have a SFT that is busy. See if it is for the current process
                                ;
0000370E 36A1[3C03]                 mov     ax,[SS:PROC_ID]             ; SS override
                                    ;cmp    [es:di+31h],ax
00003712 26394531                   cmp     [es:di+SF_ENTRY.sf_PID],ax
00003716 750D                       jnz     short scan_next
00003718 36A1[3E03]                 mov     ax,[SS:USER_ID]             ; SS override
                                    ;sub    ax,[es:di+2Fh] ; PCDOS 7.1 IBMDOS.COM ; 03/02/2024
                                    ;cmp    [es:di+2Fh],ax
0000371C 2639452F                   cmp     [es:di+SF_ENTRY.sf_UID],ax
00003720 7503                       jnz     short scan_next

                                ; This SFT is labelled as ours.

                                ; 03/02/2024 - Retro DOS v5.0
                                %if 0
                                    mov     word [es:di],0
                                    ;mov    word [es:di+SF_ENTRY.sf_ref_count],0
                                %else
                                    ; 03/02/2024
                                    ; PCDOS 7.1 IBMDOS.COM
00003722 E8C613                     call    SFT_FREE
                                %endif

                                scan_next:
00003725 43                         inc     bx
00003726 EBD9                       jmp     short Scan

                                ;============================================================================
                                ; CLOSE.ASM, MSDOS 6.0, 1991
                                ;============================================================================
                                ; 23/07/2018 - Retro DOS v3.0
                                ; 18/05/2019 - Retro DOS v4.0

                                ;** Internal Close and Commit calls to close a local or NET SFT.
                                ;
                                ;   DOS_CLOSE
                                ;   DOS_COMMIT
                                ;   FREE_SFT
                                ;   SetSFTTimes
                                ;
                                ;   Revision history:
                                ;
                                ;      AN000  version 4.00Jan. 1988
                                ;      A005   PTM 3718 --- lost clusters when fastopen installed
                                ;      A011   PTM 4766 --- C2 fastopen problem

                                ;Installed = TRUE

                                ;Break <DOS_CLOSE -- CLOSE FILE from SFT>
                                ;----------------------------------------------------------------------------
                                ;
                                ; Procedure Name : DOS_CLOSE
                                ;
                                ; Inputs:
                                ;   [THISSFT] set to the SFT for the file being used
                                ; Function:
                                ;   Close the indicated file via the SFT
                                ; Returns:
                                ;   sf_ref_count decremented otherwise
                                ;   ES:DI point to SFT
                                ;   Carry set if error
                                ;       AX has error code
                                ; DS preserved, others destroyed
                                ;----------------------------------------------------------------------------

                                ;hkn; DOS_CLOSE called from fcbio.asm and handle.asm. DS alreday set up.

                                ; 18/05/2019 - Retro DOS v4.0
                                ; DOSCODE:6E2Eh (MSDOS 6.21, MSDOS.SYS)

                                ; 14/11/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
                                ; DOSCODE:6E1Ah (MSDOS 5.0, MSDOS.SYS)

                                ; 23/07/2018 - IBMDOS.COM (MSDOS 3.3), 1987 - Offset 39D0h

                                    ; 03/02/2024 - Retro DOS v5.0
                                    ; PCDOS 7.1 IBMDOS.COM - DOSCODE:76FEh
                                    ; (Win ME IO.SYS - BIOSCODE:7579h)

                                DOS_CLOSE:
00003728 C43E[9E05]                 LES     DI,[THISSFT]
                                    ;mov    bx,[ES:DI+5]
0000372C 268B5D05                   MOV     BX,[ES:DI+SF_ENTRY.sf_flags]

                                ; Network closes are handled entirely by the net code.
```

```
22678
22679                                   ;;test  bx,8000h
22680                                   ;TEST   BX,sf_isnet
22681                                   ; 17/12/2022
22682                                   ;test   bh,80h
22683 00003730 F6C780                   test    bh,(sf_isnet>>8)
22684 00003733 7406                     JZ      short LocalClose
22685
22686                                   ;CallInstall Net_Close,MultNET,6
22687 00003735 B80611                   mov     ax,1106h
22688 00003738 CD2F                     int     2Fh     ; Multiplex - NETWORK REDIRECTOR - CLOSE REMOTE FILE
22689                                                   ; ES:DI -> SFT
22690                                                   ; SFT DPB field -> DPB of drive containing file
22691                                                   ; Return: CF set on error, AX = DOS error code
22692                                                   ; CF clear if successful
22693                                 RET2:             ; 03/02/2024
22694 0000373A C3                       retn
22695
22696                                 ; All closes release the sharing information.
22697                                 ; No commit releases sharing information
22698                                 ;
22699                                 ; All closes decrement the ref count.
22700                                 ; No commit decrements the ref count.
22701
22702                                 LocalClose:
22703 0000373B E8A4E1                   call    ECritDisk
22704 0000373E E8C001                   CALL    SetSFTTimes
22705 00003741 E84801                   CALL    FREE_SFT              ; dec ref count or mark as busy
22706
22707                                 ;hkn; SS is DOSDATA
22708                                   ;Context DS
22709 00003744 16                       push    ss
22710 00003745 1F                       pop     ds
22711
22712 00003746 50                       push    ax
22713 00003747 53                       push    bx
22714 00003748 E8394D                   call    ShareEnd
22715 0000374B 5B                       pop     bx
22716 0000374C 58                       pop     ax
22717
22718                                 ; Commit enters here. AX from commit MUST be <> 1, BX is flags word
22719
22720                                 CloseEntry:
22721 0000374D 50                       PUSH    AX
22722
22723                                 ; File clean or device does not get stamped nor disk looked at.
22724
22725                                   ;test   bx,0C0h
22726                                   ; 17/12/2022
22727 0000374E F6C3C0                   test    bl,devid_file_clean+devid_device
22728                                   ;TEST   BX,devid_file_clean+devid_device
22729 00003751 7403                     JZ      short rdir
22730                                   ; 14/11/2022
22731 00003753 E9FD00                   JMP     FREE_SFT_OK           ; either clean or device
22732                                   ;jnz    short FREE_SFT_OK ; 24/07/2019
22733
22734                                 ; Retrieve the directory entry for the file
22735
22736                                 rdir:
22737 00003756 E84001                   CALL    DirFromSFT
22738                                   ;mov    al,5
22739 00003759 B005                     MOV     AL,error_access_denied
22740
22741                                   ; 03/02/2024
22742                                   ;JNC    short clook
22743                                   ;; 14/11/2022
22744                                   ;JMP    CloseFinish           ; pretend the close worked.
22745                                   ;;jc    short CloseFinish ; 24/07/2019
22746
22747                                   ;;;
22748                                   ; 03/02/2024 - Retro DOS v5.0
22749                                   ; (PCDOS 7.1 IBMDOS.COM)
22750 0000375B 723D                     jc      short jmp_to_CloseFinish ; pretend the close worked.
22751                                 rdir2:
22752                                   ;test   word [si+2],4
22753 0000375D F6440204                 test    byte [si+SF_ENTRY.sf_mode],4 ; devid_device_null
22754                                                                        ; bit 2 - null device
22755 00003761 751C                     jnz     short clook
22756
22757 00003763 1E                        push    ds
22758 00003764 53                       push    bx
22759                                   ; 27/06/2024
22760                                   ;lds    bx,[si+7]
22761 00003765 C55C07                   lds     bx,[si+SF_ENTRY.sf_devptr] ; pointer to DPB
22762
22763 00003768 8A1F                     mov     bl,[bx]               ; DPB.DRIVE
22764 0000376A 30FF                     xor     bh,bh  ; 0
22765 0000376C 36F687[0813]04           test    byte [ss:bx+drive_flags],4
22766                                                                 ; bit 2 - last access date/time flag?
22767                                                                 ; or disk accessed flag !?
22768 00003772 5B                       pop     bx
22769 00003773 7409                     jz      short skip_upd_laccdt ; no support for last access date&time
22770 00003775 1F                       pop     ds
22771 00003776 1E                       push    ds
22772 00003777 E8E6D3                   call    DATE16
22773                                   ;mov    [es:di+12h],ax
22774 0000377A 26894512                 mov     [es:di+dir_entry.dir_lstaccdate],ax
22775                                 skip_upd_laccdt:
22776 0000377E 1F                       pop     ds
22777                                   ;;;
22778
22779                                 clook:
22780
22781                                 ; ES:DI points to entry
22782                                 ; DS:SI points to SFT
22783                                 ; ES:BX points to buffer header
22784
22785 0000377F 57                        push    di
22786 00003780 56                       push    si
22787                                   ;lea    si,[si+20h]
22788 00003781 8D7420                   LEA     SI,[SI+SF_ENTRY.sf_name]
22789
22790                                 ; ES:DI point to directory entry
22791                                 ; DS:SI point to unpacked name
22792
22793 00003784 E856E0                    call    XCHGP
22794
22795                                 ; ES:DI point to unpacked name
22796                                 ; DS:SI point to directory entry
22797
22798 00003787 E88F10                    call    MetaCompare
22799 0000378A E850E0                   call    XCHGP
22800 0000378D 5E                       pop     si
22801 0000378E 5F                       pop     di
```

```
22802 0000378F 740C                     JZ      short CLOSE_GO          ; Name OK
22803                                 Bye:
22804 00003791 89F7                     MOV     DI,SI
22805 00003793 1E                       PUSH    DS
22806 00003794 07                       POP     ES                      ; ES:DI points to SFT
22807 00003795 16                       PUSH    SS
22808 00003796 1F                       POP     DS
22809 00003797 F9                       STC
22810                                   ;mov    al,2
22811 00003798 B002                     MOV     AL,error_file_not_found
22812
22813                                 jmp_to_CloseFinish: ; 03/02/2024
22814                                   ;JMP    CloseFinish ; 24/07/2019
22815                                   ; 03/02/2024
22816                                   ; (PCDOS 7.1 IBMDOS.COM)
22817 0000379A E9DE00                   jmp     CloseFinish2
22818
22819                                   ; 18/05/2019 - Retro DOS v4.0
22820                                 CLOSE_GO:
22821                                   ; 03/02/2024
22822                                   ;mov    al,[si+4]
22823 0000379D 8A4404                   mov     al,[si+SF_ENTRY.sf_attr]
22824
22825                                   ; MSDOS 6.0
22826                                   ;test   word [si+2],8000h
22827                                   ;TEST   word [SI+SF_ENTRY.sf_mode],sf_isFCB ; FCB ?
22828                                   ; 17/12/2022
22829                                   ;test   byte [si+3],80h
22830 000037A0 F6440380                 test    byte [SI+SF_ENTRY.sf_mode+1],(sf_isFCB>>8) ; FCB ?
22831 000037A4 740A                     JZ      short nofcb             ; no, set dir attr, sf_attr
22832                                   ; MSDOS 3.3 & MSDOS 6.0
22833                                   ;mov    ch,[es:di+0Bh]
22834 000037A6 268A6D0B                 MOV     CH,[ES:DI+dir_entry.dir_attr]
22835
22836                                   ; 03/02/2024
22837                                   ;;mov   al,[si+4]
22838                                   ;MOV    AL,[SI+SF_ENTRY.sf_attr]
22839
22840                                 ;hkn; SS override
22841 000037AA 36A2[6B05]               MOV     [SS:ATTRIB],AL
22842                                   ; MSDOS 3.3
22843                                   ;;call  MatchAttributes
22844                                   ;;JNZ   short Bye               ; attributes do not match
22845                                   ; 18/05/2019
22846 000037AE EB04                     JMP     SHORT setattr           ;FT.
22847                                 nofcb:
22848                                   ; 03/02/2024
22849                                   ; MSDOS 6.0
22850                                   ;;mov   al,[si+4]
22851                                   ;MOV    AL,[SI+SF_ENTRY.sf_attr] ;FT.         ;AN000;
22852
22853 000037B0 2688450B                 MOV     [ES:DI+dir_entry.dir_attr],AL ;FT.    ;AN000;
22854                                 setattr:
22855                                   ; MSDOS 3.3 (& MSDOS 6.0)
22856                                   ;or     byte [es:di+0Bh],20h
22857 000037B4 26804D0B20               OR      BYTE [ES:DI+dir_entry.dir_attr],attr_archive ;Set archive
22858                                   ; MSDOS 6.0
22859                                   ;mov    ax,[es:di+1Ah]
22860 000037B9 268B451A                 MOV     AX,[ES:DI+dir_entry.dir_first] ;AN011
22861                                                                   ;F.O. save old first cluster
22862                                 ;hkn; SS override
22863 000037BD 36A3[BD0E]               MOV     [SS:OLD_FIRSTCLUS],AX ;AN011;F.O. save old first cluster
22864
22865                                   ;;;
22866                                   ; 03/02/2024
22867                                   ; PCDOS 7.1 IBMDOS.COM
22868                                   ;mov    ax,[es:di+14h]
22869 000037C1 268B4514                 mov     ax,[ES:DI+dir_entry.dir_fclus_hi] ; old first cluster, hw
22870                                   ; 27/06/2024
22871 000037C5 36A3[3A11]               MOV     [ss:OLD_FIRSTCLUS_HW],ax
22872                                   ;;;
22873
22874                                   ;;mov   ax,[si+0Bh]
22875                                   ;MOV    AX,[SI+SF_ENTRY.sf_firclus]
22876                                   ;;;
22877                                   ; 03/02/2024
22878                                   ; PCDOS 7.1 IBMDOS.COM
22879 000037C9 8B442B                   mov     ax,[si+2Bh]    ; mov ax,[si+SF_ENTRY.sf_chain]
22880                                                          ; first cluster (32 bit) low word !
22881                                   ;;;
22882                                   ;mov    [es:di+1Ah],ax
22883 000037CC 2689451A                 MOV     [ES:DI+dir_entry.dir_first],AX       ;Set firclus pointer
22884                                   ;;; 03/02/2024
22885 000037D0 8B442D                   mov     ax,[si+2Dh]    ; mov ax,[si+SF_ENTRY.sf_chain+2]
22886                                                          ; first cluster (32 bit) high word !
22887                                   ;mov    [es:di+14h],ax
22888 000037D3 26894514                 mov     [es:di+dir_entry.dir_fclus_hi],ax
22889                                   ;;;
22890
22891                                 ; 03/02/2024
22892                                 %if 0
22893                                   ;mov    ax,[si+11h]
22894                                   MOV     AX,[SI+SF_ENTRY.sf_size]
22895                                   ;mov    [es:di+1Ch],ax
22896                                   MOV     [ES:DI+dir_entry.dir_size_l],AX      ;Set size
22897                                   ;mov    ax,[si+13h]
22898                                   MOV     AX,[SI+SF_ENTRY.sf_size+2]
22899                                   ;mov    [es:di+1Eh],ax
22900                                   MOV     [ES:DI+dir_entry.dir_size_h],AX
22901                                   ;mov    ax,[si+0Fh]
22902                                   MOV     AX,[SI+SF_ENTRY.sf_date]
22903                                   ;mov    [es:di+18h],ax
22904                                   MOV     [ES:DI+dir_entry.dir_date],AX ;Set date
22905                                   ;mov    ax,[si+0Dh]
22906                                   MOV     AX,[SI+SF_ENTRY.sf_time]
22907                                   ;mov    [es:di+16h],ax
22908                                   MOV     [ES:DI+dir_entry.dir_time],AX ;Set time
22909                                 %else
22910                                   ; 03/02/2024 - Retro DOS v5.0
22911 000037D7 56                       push    si
22912 000037D8 83C60D                   add     si,0Dh
22913 000037DB AD                       lodsw   ; [si+SF_ENTRY.sf_time]
22914 000037DC 26894516                 mov     [es:di+dir_entry.dir_time],ax ; Set time
22915 000037E0 AD                       lodsw   ; [si+SF_ENTRY.sf_date]
22916 000037E1 26894518                 mov     [es:di+dir_entry.dir_date],ax ; Set date
22917 000037E5 AD                       lodsw   ; [si+SF_ENTRY.sf_size]
22918 000037E6 2689451C                 mov     [es:di+dir_entry.dir_size_l],ax      ; Set size
22919 000037EA AD                       lodsw   ; [si+SF_ENTRY.sf_size+2]
22920 000037EB 2689451E                 mov     [es:di+dir_entry.dir_size_h],ax
22921 000037EF 5E                       pop     si
22922                                 %endif
22923
22924                                   ; MSDOS 6.0
22925                                 ;; File Tagging
```

```
                                    TEST    byte [ES:BX+BUFFINFO.buf_flags],buf_dirty
                                                            ;LB. if already dirty                   ;AN000;
                                    JNZ     short yesdirty4         ;LB.  don't increment dirty count ;AN000;
                                    ; 02/06/2019
                                    call    INC_DIRTY_COUNT   ;LB.                          ;AN000;
                                    ; MSDOS 3.3 (& MSDOS 6.0)
                                    ;or     byte [es:bx+5],40h
                                    OR      byte [ES:BX+BUFFINFO.buf_flags],buf_dirty ;Buffer dirty
                                yesdirty4:
                                    push    ds
                                    push    si

                                ; 03/02/2024
                                %if 0
                                    ; MSDOS 6.0
                                    ;mov    cx,[si+0Bh]
                                    ; 07/12/2022
                                    MOV     CX,[SI+SF_ENTRY.sf_firclus] ; do this for Fastopen
                                %else
                                    ; 03/02/2024
                                    ; PCDOS 7.1
                                    mov     cx,[si+2Bh]             ; [es:di+SF_ENTRY.sf_chain]
                                                                    ; first cluster (32 bit) low word !?
                                %endif

                                ;hkn; SS override
                                    MOV     AL,[SS:THISDRV]
                                    ; MSDOS 3.3
                                    ;push   ss
                                    ;pop    ds
                                    ;MOV    AL,[THISDRV]
                                ;;; 10/1/86  update fastopen cache
                                    ; MSDOS 3.3 & MSDOS 6.0
                                    PUSH    DX
                                    MOV     AH,0                    ; dir entry update
                                    MOV     DL,AL                   ; drive number A=0, B=1,,,
                                    ;;;
                                    ; 03/02/2024 - Retro DOS v5.0
                                    ; PCDOS 7.1
                                    push    bx ; *
                                    mov     bx,[si+2Dh]             ; [es:di+SF_ENTRY.sf_chain+2]
                                                                    ; first cluster (32 bit) high word !?
                                    or      bx,bx
                                    jnz     short do_update2
                                    ;;;
                                    ; MSDOS 6.0
                                    OR      CX,CX                   ;AN005; first cluster 0; may be truncated
                                    JNZ     short do_update2        ;AN005; no, do update
                                    MOV     AH,3                    ;AN005; do a delete cache entry
                                    ; 27/06/2024
                                    ;;;
                                    ;;mov   di,[si+1Bh]
                                    ;MOV    DI,[SI+SF_ENTRY.sf_dirsec] ;AN005; cx:di = dir sector
                                    ;;mov   cx,[si+1Dh]
                                    ;MOV    CX,[SI+SF_ENTRY.sf_dirsec+2] ;AN005;
                                    lds     di,[si+SF_ENTRY.sf_dirsec]
                                    mov     cx,ds
                                    ;;;
                                    ;mov    dh,[si+1Fh]
                                    MOV     DH,[SI+SF_ENTRY.sf_dirpos] ;AN005; dh = dir pos
                                    JMP     SHORT do_update         ;AN011;F.O.

                                do_update2:                         ;AN011;F.O.
                                    ;;;
                                    ; 03/02/2024
                                    ; PCDOS 7.1
                                    cmp     bx,[ss:OLD_FIRSTCLUS_HW] ; same as old first cluster?
                                    jnz     short do_update3 ; no
                                    ;;;

                                ;hkn; SS override fort OLD_FIRSTCLUS
                                    ;
                                    CMP     CX,[SS:OLD_FIRSTCLUS] ;AN011;F.O. same as old first cluster?
                                    JZ      short do_update             ;AN011;F.O. yes
                                do_update3: ; 03/02/2024
                                    MOV     AH,2                    ;AN011;F.O. delete the old entry
                                    MOV     CX,[SS:OLD_FIRSTCLUS] ;AN011;F.O.
                                    ;;;
                                    ; 03/02/2024
                                    ; PCDOS 7.1
                                    mov     bx,[ss:OLD_FIRSTCLUS_HW]
                                    ;;;
                                do_update:                          ;AN005;
                                ;hkn; SS is DOSDATA
                                    ;Context DS
                                    push    ss
                                    pop     ds
                                    ;;;
                                    ; 03/02/2024 - Retro DOS v5.0
                                    xchg    bx,si   ; PCDOS 7.1 IBMDOS.COM
                                    ;;;
                                    ; MSDOS 3.3 & MSDOS 6.0
                                    call    FastOpen_Update         ; invoke fastopen
                                    ;;;
                                    ; 03/02/2024
                                    xchg    bx,si   ; PCDOS 7.1 IBMDOS.COM
                                    pop     bx ; *
                                    ;;;
                                    POP     DX

                                ;;; 10/1/86  update fastopen cache
                                    call    FLUSHBUF                ; flush all relevant buffers
                                    pop     di
                                    pop     es
                                    ;mov    al,5
                                    MOV     AL,error_access_denied
                                    JC      short CloseFinish
                                FREE_SFT_OK:
                                    ; 03/02/2024
                                    ;CLC                            ; signal no error.

                                    ;;;
                                    ; 03/02/2024 - Retro DOS v5.0
                                    ; PCDOS 7.1 IBMDOS.COM
                                    ;test   word [es:di+5],8080h
                                    test    word [es:di+SF_ENTRY.sf_flags],sf_isnet+devid_device
                                    jnz     short CloseFinish2
                                    push    es
                                    push    bp
                                    ;les    bp,[es:di+7]
                                    les     bp,[es:di+SF_ENTRY.sf_devptr] ; DPB
                                    call    update_fat32_fsinfo
                                    pop     bp
                                    pop     es
```

```
23050 00003866 EB13                     jmp     short CloseFinish2
23051
23052                                   CloseFinish:        ; 03/02/2024 - Retro DOS v5.0
23053 00003868 1E                       push    ds
23054 00003869 53                       push    bx
23055 0000386A 26C55D07                 lds     bx,[es:di+7]            ; [es:di+SF_ENTRY.sf_devptr] ; DPB
23056 0000386E 8A1F                     mov     bl,[bx]                         ; DPB.DRIVE
23057 00003870 30FF                     xor     bh,bh ; 0
23058 00003872 3680A7[0813]FB           and     byte [ss:bx+drive_flags],0FBh ; clear bit 2
23059                                                                   ; bit 2 - last access date/time flag?
23060                                                                   ; or disk accessed (successful) flag !?
23061 00003878 5B                       pop     bx
23062 00003879 1F                       pop     ds
23063 0000387A F9                       stc
23064                                   ;;;
23065
23066                                   CloseFinish2:       ; 03/02/2024 - Retro DOS v5.0
23067                                   ;Closefinish:
23068
23069                                   ; Indicate to the device that the SFT is being closed.
23070
23071                                   ;;;; 7/21/86
23072 0000387B 9C                       PUSHF                           ; save flag from DirFromSFT
23073 0000387C E89D19                   call    DEV_CLOSE_SFT
23074 0000387F 9D                       POPF
23075                                   ;;;; 7/21/86
23076                                   ;
23077                                   ; See if the ref count indicates that we have busied the SFT. If so, mark the
23078                                   ; SFT as being free. Note that we do NOT need to be in critSFT as we are ONLY
23079                                   ; going to be moving from busy to free.
23080                                   ;
23081 00003880 59                       POP     CX                      ; get old ref count
23082 00003881 9C                       PUSHF
23083                                   ; 03/02/2024
23084                                   ;DEC    CX                      ; if cx != 1
23085                                   ;JNZ    short NoFree            ; then do NOT free SFT
23086 00003882 E203                     loop    NoFree ; PCDOS 7.1 IBMDOS.COM
23087
23088                                   ; 03/02/2024 - Retro DOS v5.0
23089                                   %if 0
23090                                       mov     [es:di],cx
23091                                       ;MOV    [ES:DI+SF_ENTRY.sf_ref_Count],CX ; mov [es:di+0],cx
23092                                   %else
23093                                       ; 03/02/2024
23094                                       ; PCDOS 7.1 IBMDOS.COM
23095 00003884 E86412                   call    SFT_FREE
23096                                   %endif
23097
23098                                   NoFree:
23099 00003887 E885E0                   call    LCritDisk
23100 0000388A 9D                       POPF
23101 0000388B C3                       retn
23102
23103                                   ;--------------------------------------------------------------------------
23104                                   ;
23105                                   ; Procedure Name : FREE_SFT
23106                                   ;
23107                                   ; ES:DI -> SFT. Decs sft_ref_count. If the count goes to 0, mark it as busy.
23108                                   ; Flags preserved. Return old ref count in AX
23109                                   ;
23110                                   ; Note that busy is indicated by the SFT ref count being -1.
23111                                   ;
23112                                   ;--------------------------------------------------------------------------
23113
23114                                   FREE_SFT:
23115 0000388C 9C                       PUSHF           ; Save carry state
23116 0000388D 268B05                   mov     ax,[es:di]
23117                                   ;MOV    AX,[ES:DI+SF_ENTRY.sf_ref_count]
23118 00003890 48                       DEC     AX
23119 00003891 7501                     JNZ     short SetCount
23120 00003893 48                       DEC     AX
23121                                   SetCount:
23122 00003894 268705                   xchg    ax,[es:di]
23123                                   ;XCHG   AX,[ES:DI+SF_ENTRY.sf_ref_count]
23124 00003897 9D                       POPF
23125 00003898 C3                       retn
23126
23127                                       ; 18/05/2019 - Retro DOS v4.0
23128
23129                                   ;--------------------------------------------------------------------------
23130                                   ;
23131                                   ; Procedure Name : DirFromSFT
23132                                   ;
23133                                   ;   DirFromSFT - locate a directory entry given an SFT.
23134                                   ;
23135                                   ;   Inputs: ES:DI point to SFT
23136                                   ;           DS = DOSDATA
23137                                   ;   Outputs:
23138                                   ;           EXTERR_LOCUS = errLOC_Disk
23139                                   ;           CurBuf points to buffer
23140                                   ;           Carry Clear -> operation OK
23141                                   ;               ES:DI point to entry
23142                                   ;               ES:BX point to buffer
23143                                   ;               DS:SI point to SFT
23144                                   ;           Carry SET   -> operation failed
23145                                   ;               registers trashified
23146                                   ;   Registers modified: ALL
23147                                   ;--------------------------------------------------------------------------
23148
23149                                       ; 04/02/2024 - Retro DOS v5.0
23150                                       ; PCDOS 7.1 IBMDOS.COM
23151
23152                                   DirFromSFT:
23153                                       ;;mov   byte [EXTERR_LOCUS],2
23154                                       ;MOV    byte [EXTERR_LOCUS],errLOC_Disk
23155                                       ; 04/02/2024
23156 00003899 E84ADA                   call    set_exerr_locus_disk
23157                                       ;
23158 0000389C 06                       push    es
23159 0000389D 57                       push    di
23160                                       ; MSDOS 3.3
23161                                       ;;mov   dx,[es:di+1Dh]
23162                                       ;MOV    dx,[ES:DI+SF_ENTRY.sf_dirsec]
23163                                       ; MSDOS 6.0
23164                                       ;mov    dx,[es:[di+1Dh]
23165 0000389E 268B551D                   MOV     DX,[ES:DI+SF_ENTRY.sf_dirsec+2]  ;F.C. >32mb
23166 000038A2 8916[0706]                 MOV     [HIGH_SECTOR],DX                ;F.C. >32mb
23167                                       ; 04/02/2024
23168 000038A6 52                       push    dx
23169                                       ;mov    dx,[es:di+1Bh]
23170 000038A7 268B551B                   MOV     DX,[ES:DI+SF_ENTRY.sf_dirsec]
23171                                       ; 04/02/2024
23172                                       ; 19/05/2019
23173                                       ;PUSH   word [HIGH_SECTOR]      ;F.C. >32mb
```

```
                                    ; MSDOS 3.3 & MSDOS 6.0
000038AB 52                         PUSH    DX
000038AC E89B2C                     call    FATREAD_SFT             ; ES:BP points to DPB, [THISDRV] set
                                                                    ; [THISDPB] set
000038AF 5A                         POP     DX
000038B0 8F06[0706]                 POP     word [HIGH_SECTOR]      ;F.C. >32mb
000038B4 721E                       JC      short PopDone
                                    ; 22/09/2023
                                    ;XOR    AL,AL   ; *             ; Pre read
                                    ;;mov   byte [ALLOWED],18h
                                    ;MOV    byte [ALLOWED],Allowed_FAIL+Allowed_RETRY ; *
                                    ;call   GETBUFFR
                                    ; 22/09/2023
000038B6 E87030                     call    GETBUFFER ; *           ; Pre read
000038B9 7219                       JC      short PopDone
000038BB 5E                         pop     si
000038BC 1F                         pop     ds                      ; Get back SFT pointer

                                ;hkn; SS override
000038BD 36C43E[E205]               LES     DI,[SS:CURBUF]
                                    ;or     byte [es:di+5],4
000038C2 26804D0504                 OR      byte [ES:DI+BUFFINFO.buf_flags],buf_isDIR
000038C7 89FB                       MOV     BX,DI                   ; ES:BX point to buffer header
                                    ;;;lea  di,[di+16] ; MSDOS 3.3
                                    ;;lea   di,[di+20] ; MSDOS 6.0
                                    ; 04/02/2024
                                    ;lea    di,[di+24] ; MSDOS 7.1
000038C9 8D7D18                     LEA     DI,[DI+BUFINSIZ]        ; Point to buffer
                                    ;mov    al,32
000038CC B020                       MOV     AL,dir_entry.size
                                    ;mul    byte [si+1Fh] ; MSDOS 6.0
000038CE F6641F                     MUL     byte [SI+SF_ENTRY.sf_dirpos]
000038D1 01C7                       ADD     DI,AX                   ; Point at the entry
000038D3 C3                         retn                            ; carry is clear
                                PopDone:
000038D4 5F                         pop     di
000038D5 07                         pop     es
                                PopDone_retn:
000038D6 C3                         retn

                                ;---------------------------------------------------------------------------
                                ;
                                ;** DOS_Commit - UPdate Directory Entries
                                ;
                                ;   ENTRY   same as DOS_CLOSE (??? BUGBUG - update this jgl)
                                ;           (DS) = DOSGROUP
                                ;   EXIT    Same as DOS_CLOSE except ref_count field is not altered
                                ;   USES    all but DS
                                ;
                                ;---------------------------------------------------------------------------

                                ; 14/11/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
                                ; DOSCODE:6F72h (MSDOS 5.0, MSDOS.SYS)

                                ; 04/02/2024 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMDOS.COM)
                                ; DOSCODE:78B8h (PCDOS 7.1 IBMDOS.COM)

                                DOS_COMMIT:
                                    ;hkn; called from srvcall. DS already set up.
000038D7 C43E[9E05]                 LES     DI,[THISSFT]
                                    ;mov    bx,[es:di+5]
000038DB 268B5D05                   MOV     BX,[ES:DI+SF_ENTRY.sf_flags]
                                    ;test   bx,0C0h
                                    ; 17/12/2022
000038DF F6C3C0                     test    bl,devid_file_clean+devid_device ;Clears carry
                                    ;TEST   BX,devid_file_clean+devid_device ;Clears carry
000038E2 75F2                       jnz     short PopDone_retn
                                    ;test   bx,8000h
                                    ; 17/12/2022
                                    ;test   bh,80h
000038E4 F6C780                     test    bh,(sf_isnet>>8) ; 80h
                                    ;TEST   BX,sf_isnet ; 8000h
000038E7 7406                       JZ      short LOCAL_COMMIT

                                ;IF NOT Installed
                                ;   transfer NET_COMMIT
                                ;ELSE
                                    ;mov    ax,1107h
000038E9 B80711                     MOV     AX,(MultNET<<8)|7
000038EC CD2F                       int      2Fh    ; Multiplex - NETWORK REDIRECTOR - COMMIT REMOTE FILE
                                                    ; ES:DI -> SFT
                                                    ; SFT DPB field -> DPB of drive containing file
                                                    ; Return: CF set on error, AX = DOS error code
                                                    ; CF clear if successful
                                localcommit_retn: ; 18/12/2022
000038EE C3                         retn
                                ;ENDIF

                                ; Perform local commit operation by doing a close but not releaseing the SFT.
                                ; There are three ways we can do this. One is to enter a critical section to
                                ; protect a potential free. The second is to increment the ref count to mask
                                ; the close decrementing.
                                ;
                                ; The proper way is to let the caller's of close decide if a decrement should
                                ; be done. We do this by providing another entry into close after the
                                ; decrement and after the share information release.

                                ; DOSCODE:6FA0h (MSDOS 6.21, MSDOS.SYS)
                                ; DOSCODE:6F8Ch (MSDOS 5.0, MSDOS.SYS)
                                ; 06/08/2025
                                ; DOSCODE:78D0h (PCDOS 7.1, IBMDOS.COM)

                                LOCAL_COMMIT:
                                    ; 06/08/2025
                                    ;call   ECritDisk
                                    ; MSDOS 6.0
000038EF E8F0DF                     call    ECritDisk       ;PTM.
000038F2 E80C00                     call    SetSFTTimes
000038F5 B8FFFF                     MOV     AX,-1 ; 0FFFFh
000038F8 E852FE                     call    CloseEntry
                                    ; MSDOS 6.0
000038FB 9C                         PUSHF                   ;PTM.                           ;AN000;
000038FC E81519                     call    DEV_OPEN_SFT    ;PTM.  increment device count ;AN000;
                                    ; 06/08/2025
                                    ;POPF                   ;PTM.                           ;AN000;
                                    ;call   LCritDisk       ;PTM.                           ;AN000;
                                    ; 18/12/2022
                                    ;jmp    LCritDisk
                                    ; 06/08/2025
000038FF EB86                       jmp     NoFree
                                ;localcommit_retn:
                                ;   retn

                                ;Break      <SetSFTTimes - signal a change in the times for an SFT>
```

```
                        ;--------------------------------------------------------------------------
                        ;
                        ; Procedure Name : SetSFTTimes
                        ;
                        ;   SetSFTTimes - Examine the flags for a SFT and set the time appropriately.
                        ;   Reflect these times in other SFT's for the same file.
                        ;
                        ;   Inputs: ES:DI point to SFT
                        ;           BX = sf_flags set appropriately
                        ;   Outputs:       Set sft times to current time if File & dirty & !nodate
                        ;   Registers modified: All except ES:DI, BX, AX
                        ;
                        ;--------------------------------------------------------------------------

                            ; 04/02/2024 - Retro DOS v5.0
                            ; PCDOS 7.1 IBMDOS.COM

                        SetSFTTimes:

                        ; 04/02/2024
                        %if 0
                        ;   File clean or device does not get stamped nor disk looked at.

                            ;test   bx,0C0h
                            ; 17/12/2022
                            test    bl,devid_file_clean+devid_device
                            ;TEST   BX,devid_file_clean+devid_device
                            ;retnz                          ; clean or device => no timestamp
                            jnz     short localcommit_retn

                        ;   file and dirty. See if date is good

                            ;test   bx,4000h
                            ; 17/12/2022
                            ;test   bh,40h
                            test    bh,(sf_close_nodate>>8)
                            ;TEST   BX,sf_close_nodate
                            ;retnz                          ; nodate => no timestamp
                            jnz     short localcommit_retn
                        %else
                            ; 04/02/2024
                            ; (PCDOS 7.1 IBMDOS.COM)
                            ;test   bx,40C0h
 00003901 F7C3C040          test    bx,sf_close_nodate+devid_file_clean+devid_device
 00003905 75E7              jnz     short localcommit_retn
                        %endif

 00003907 50                push    ax
 00003908 53                push    bx
 00003909 E854D2            call    DATE16                  ; Date/Time to AX/DX
                            ;mov    [es:di+0Fh],ax
 0000390C 2689450F          MOV     [ES:DI+SF_ENTRY.sf_date],AX
                            ;mov    [es:di+0Dh],dx
 00003910 2689550D          MOV     [ES:DI+SF_ENTRY.sf_time],DX
 00003914 31C0              XOR     AX,AX
                        ;if installed
                            ;call   JShare + 14 * 4
 00003916 FF1E[C800]        call    far [JShare+(14*4)]   ; 14 = ShSU
                        ;else
                        ;   call    ShSU
                        ;endif
 0000391A 5B                pop     bx
 0000391B 58                pop     ax
 0000391C C3                retn

                        ;==========================================================================
                        ; DIRCALL.ASM, MSDOS 6.0, 1991
                        ;==========================================================================
                        ; 23/07/2018 - Retro DOS v3.0
                        ; 18/05/2019 - Retro DOS v4.0

                        ; DOSCODE:6FDAh (MSDOS 6.21, MSDOS.SYS)

                        ;TITLE DIRCALL - Directory manipulation internal calls
                        ;NAME  DIRCALL

                        ;** Low level directory manipulation routines for making removing and
                        ;     verifying local or NET directories
                        ;
                        ;   DOS_MKDIR
                        ;   DOS_CHDIR
                        ;   DOS_RMDIR
                        ;
                        ;   Modification history:
                        ;
                        ;           Created: ARR 30 March 1983

                        ;BREAK <DOS_MkDir - Make a directory entry>
                        ;--------------------------------------------------------------------------
                        ;
                        ; Procedure Name : DOS_MkDir
                        ;
                        ; Inputs:
                        ;   [WFP_START] Points to WFP string ("d:/" must be first 3 chars, NUL
                        ;           terminated)
                        ;   [CURR_DIR_END] Points to end of Current dir part of string
                        ;           ( = -1 if current dir not involved, else
                        ;            Points to first char after last "/" of current dir part)
                        ;   [THISCDS] Points to CDS being used
                        ;           (Low word = -1 if NUL CDS (Net direct request))
                        ; Function:
                        ;   Make a new directory
                        ; Returns:
                        ;   Carry Clear
                        ;           No error
                        ;   Carry Set
                        ;       AX is error code
                        ;           error_path_not_found
                        ;                   Bad path (not in curr dir part if present)
                        ;           error_bad_curr_dir
                        ;                   Bad path in current directory part of path
                        ;           error_access_denied
                        ;                   Already exists, device name
                        ; DS preserved, Others destroyed
                        ;--------------------------------------------------------------------------

                        ;hkn; called from path.asm. DS already set up.

                        ; 17/11/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
                        ; DOSCODE:6FC6h (MSDOS 5.0, MSDOS.SYS)

                        ; 04/02/2024 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMDOS.COM)
                        ; DOSCODE:78FFh (PCDOS 7.1, IBMDOS.COM)
```

```
                                DOS_MKDIR:
0000391D E809DF                     call    TestNet
00003920 7313                       JNC     short LOCAL_MKDIR

                                ;IF NOT Installed
                                ;   transfer NET_MKDIR
                                ;ELSE
                                    ;mov    ax,1103h
00003922 B80311                     MOV     AX,(MultNET<<8)|3
00003925 CD2F                       int      2Fh    ; Multiplex - NETWORK REDIRECTOR - MAKE REMOTE DIRECTORY
                                                    ; SS = DOS CS
                                                    ; SDA first filename pointer -> fully-qualified directory name
                                                    ; SDA CDS pointer -> current directory
                                                    ; Return: CF set on error, AX = DOS error code
                                                    ; CF clear if successful
00003927 C3                         retn
                                ;ENDIF

                                NODEACCERRJ:
                                    ;mov    ax,5
00003928 B80500                     MOV     AX,error_access_denied
                                _BadRet:
0000392B F9                         STC
                                    ;call   LCritDisk
                                    ;retn
                                    ; 18/12/2022
0000392C E9E0DF                     jmp     LCritDisk

                                PATHNFJ:
0000392F E8DDDF                     call    LCritDisk
00003932 E916F8                     jmp     SET_MKND_ERR   ; Map the MakeNode error and return

                                LOCAL_MKDIR:
00003935 E8AADF                     call    ECritDisk

                                ; MakeNode requires an SFT to fiddle with. We Use a temp spot (RENBUF)

00003938 8C16[A005]                 MOV     [THISSFT+2],SS

                                ;hkn; DOSDATA
0000393C C706[9E05][3E04]           MOV     WORD [THISSFT],RENBUF

                                ;  NOTE: Need WORD PTR because MASM takes type of
                                ;   TempSFT (byte) instead of type of sf_mft (word).

                                    ;mov    word [RENBUF+33h],0 ; MSDOS 6.0
00003942 C706[7104]0000             MOV     WORD [RENBUF+SF_ENTRY.sf_MFT],0
                                                    ; make sure SHARER won't complain.
                                    ;mov    al,10h
00003948 B010                       MOV     AL,attr_directory
0000394A E80E1B                     call    MakeNode
0000394D 72E0                       JC      short PATHNFJ
0000394F 83F803                     CMP     AX,3
00003952 74D4                       JZ      short NODEACCERRJ ; Can't make a device into a directory
00003954 C42E[8A05]                 LES     BP,[THISDPB]   ; Makenode zaps this
00003958 C53E[E205]                 LDS     DI,[CURBUF]
0000395C 29FE                       SUB     SI,DI
0000395E 56                         PUSH    SI             ; Pointer to dir_first

                                ; 04/02/2024
                                %if 0
                                    ; MSDOS 6.0
                                    ;push   word [DI+8]
                                    PUSH    WORD [DI+BUFFINFO.buf_sector+2]      ;F.C. >32mb
                                    ; MSDOS 3.3 & MSDOS 6.0
                                    ;push   word [di+6]
                                    PUSH    WORD [DI+BUFFINFO.buf_sector] ; Sector of new node
                                %else
                                    ; 04/02/2024
                                    ; (PCDOS 7.1 IBMDOS.COM)
0000395F C54506                     lds     ax,[di+BUFFINFO.buf_sector] ; Sector of new node
00003962 1E                         push    ds
00003963 50                         push    ax
                                %endif

00003964 16                         push    ss
00003965 1F                         pop     ds

                                    ; 04/02/2024
                                    ;PUSH   word [DIRSTART]       ; Parent for .. entry
00003966 31C0                       XOR     AX,AX
                                    ;MOV    [DIRSTART],AX  ; Null directory
                                    ;;;
                                    ; Retro DOS v5.0
                                    ; PCDOS 7.1 IBMDOS.COM
00003968 89C2                       mov     dx,ax ; 0
0000396A 8716[DC0A]                 xchg    dx,[DIRSTART_HW] ; Null directory
0000396E 8706[C205]                 xchg    ax,[DIRSTART]
                                    ;
                                    ;cmp    word [es:bp+0Fh],0
00003972 26837E0F00                 cmp     word [es:bp+DPB.FAT_SIZE],0
00003977 7510                       jnz     short LOCAL_MKDIR_cont ; not FAT32
                                    ;cmp    [es:bp+35h],ax
00003979 26394635                   cmp     [es:bp+DPB.ROOT_CLUSTER],ax
0000397D 750A                       jne      short LOCAL_MKDIR_cont
                                    ;cmp    [es:bp+37h],dx
0000397F 26395637                   cmp     [es:bp+DPB.ROOT_CLUSTER+2],dx
00003983 7504                       jne     short LOCAL_MKDIR_cont
                                    ;
00003985 31D2                       xor     dx,dx
00003987 31C0                       xor     ax,ax
                                                    ; dx:ax = 0 for root directory
                                LOCAL_MKDIR_cont:
00003989 52                         push    dx
                                    ;;;
0000398A 50                         push    ax

0000398B E8DF19                     call    NEWDIR
0000398E 7278                       JC      short NODEEXISTSPOPDEL ; No room
00003990 E80F0F                     call    GETENT         ; First entry
00003993 7273                       JC      short NODEEXISTSPOPDEL ; Screw up
00003995 C43E[E205]                 LES     DI,[CURBUF]

                                ; 04/02/2024
                                %if 0
                                    ; MSDOS 6.0
                                    TEST    byte [ES:DI+BUFFINFO.buf_flags],buf_dirty
                                                        ;LB. if already dirty           ;AN000;
                                    JNZ     short yesdirty5        ;LB.   don't increment dirty count ;AN000;
                                    call    INC_DIRTY_COUNT  ;LB.                          ;AN000;

                                    ; MSDOS 3.3 & MSDOS 6.0
                                    ;or     byte [es:di+5],40h  ; 07/12/2022
                                    OR      byte [ES:DI+BUFFINFO.buf_flags],buf_dirty
```

```
                                %else
                                    ; 04/02/2024
                                    ; (PCDOS 7.1 IBMDOS.COM)
00003999 E82232                     call    SET_BUF_DIRTY
                                %endif

                                yesdirty5:
                                    ;;;add  di,16 ; MSDOS 3.3
                                    ;;add   di,20 ; MSDOS 6.0
                                    ; 04/02/2024
                                    ;add    di,24 ; PCDOS 7.1
0000399C 83C718                     ADD     DI,BUFINSIZ     ; Point at buffer
0000399F B82E20                     MOV     AX,202EH        ; ". "
                                    ;;;
                                    ; 04/02/2024 (PCDOS 7.1 IBMDOS.COM)
000039A2 8B16[DC0A]                 mov     dx,[DIRSTART_HW]
000039A6 8916[F20A]                 mov     [CLUSTERS_HW],dx
                                    ;;;
000039AA 8B16[C205]                 MOV     DX,[DIRSTART]  ; Point at itself
000039AE E8831A                     call    SETDOTENT
000039B1 B82E2E                     MOV     AX,2E2EH        ; ".."
000039B4 5A                         POP     DX              ; Parent
                                    ;;;
                                    ; 04/02/2024 (PCDOS 7.1 IBMDOS.COM)
000039B5 8F06[F20A]                 pop     word [CLUSTERS_HW]
                                    ;;;
000039B9 E8781A                     call    SETDOTENT
000039BC C42E[8A05]                 LES     BP,[THISDPB]
                                    ; 22/09/2023
                                    ;;mov   byte [ALLOWED],18h
                                    ;MOV    byte [ALLOWED],Allowed_FAIL+Allowed_RETRY ; *
000039C0 5A                         POP     DX              ; Entry sector
                                    ; MSDOS 6.0
000039C1 8F06[0706]                 POP     word [HIGH_SECTOR] ;F.C. >32mb

                                    ;XOR    AL,AL ; *       ; Pre read
                                    ;call   GETBUFFR
                                    ; 22/09/2023
000039C5 E8612F                     call    GETBUFFER ; *  ; Pre read
000039C8 7265                       JC      short NODEEXISTSP
                                    ;;;
                                    ; 04/02/2024 (PCDOS 7.1 IBMDOS.COM)
000039CA A1[DC0A]                   mov     ax,[DIRSTART_HW]
                                    ;;;
000039CD 8B16[C205]                 MOV     DX,[DIRSTART]
000039D1 C53E[E205]                 LDS     DI,[CURBUF]
                                    ;or     byte [di+5],4
000039D5 804D0504                   OR      byte [DI+BUFFINFO.buf_flags],buf_isDIR
000039D9 5E                         POP     SI              ; dir_first pointer
000039DA 01FE                       ADD     SI,DI
000039DC 8914                       MOV     [SI],DX         ; dir_entry.dir_first
                                    ;;;
                                    ; 04/02/2024 (PCDOS 7.1 IBMDOS.COM)
000039DE 8944FA                     mov     [si-6],ax       ; dir_entry.dir_fclus_hi
                                    ;;;

000039E1 31D2                       XOR     DX,DX
000039E3 895402                     MOV     [SI+2],DX       ; Zero size
000039E6 895404                     MOV     [SI+4],DX
                                DIRUP:
                                    ; MSDOS 6.0
000039E9 F6450540                   TEST    byte [DI+BUFFINFO.buf_flags],buf_dirty
                                    ;                       ;LB. if already dirty                   ;AN000;
000039ED 7507                       JNZ     short yesdirty6         ;LB.  don't increment dirty count ;AN000;
000039EF E8D831                     call    INC_DIRTY_COUNT  ;LB.                           ;AN000;

                                    ; MSDOS 3.3 & MSDOS 6.0
                                    ;or     byte [di+5],40h
000039F2 804D0540                   OR      byte [DI+BUFFINFO.buf_flags],buf_dirty      ; Dirty buffer
                                yesdirty6:
000039F6 16                         push    ss
000039F7 1F                         pop     ds
                                    ;;;
                                    ; 04/02/2024 (PCDOS 7.1 IBMDOS.COM)
000039F8 E871FA                     call    update_fat32_fsinfo
                                    ;;;
000039FB 268A4600                   mov     al,[es:bp]
                                    ;MOV    AL,[ES:BP+DPB.DRIVE]  ; mov al,[es:bp+0]
000039FF E89630                     call    FLUSHBUF
                                    ;mov    ax,5
00003A02 B80500                     MOV     AX,error_access_denied
                                    ;call   LCritDisk
                                    ;retn
                                    ; 18/12/2022
00003A05 E907DF                     jmp     LCritDisk

                                NODEEXISTSPOPDEL:
                                    ;;;
                                    ; 04/02/2024 (PCDOS 7.1 IBMDOS.COM)
00003A08 5A                         pop     dx              ; Parent
                                    ;;;
00003A09 5A                         POP     DX              ; Parent
00003A0A 5A                         POP     DX              ; Entry sector
                                    ; MSDOS 6.0
00003A0B 8F06[0706]                 POP     word [HIGH_SECTOR] ; F.C. >32mb
00003A0F C42E[8A05]                 LES     BP,[THISDPB]
                                    ; 22/09/2023
                                    ;;mov   byte [ALLOWED],18h
                                    ;MOV    byte [ALLOWED],Allowed_FAIL+Allowed_RETRY ; *
                                    ;XOR    AL,AL ; *       ; Pre read
                                    ;call   GETBUFFR
                                    ; 22/09/2023
00003A13 E8132F                     call    GETBUFFER ; *  ; Pre read
00003A16 7217                       JC      short NODEEXISTSP
00003A18 C53E[E205]                 LDS     DI,[CURBUF]
                                    ;or     byte [di+5],4
00003A1C 804D0504                   OR      byte [DI+BUFFINFO.buf_flags],buf_isDIR
00003A20 5E                         POP     SI              ; dir_first pointer
00003A21 01FE                       ADD     SI,DI
                                    ;sub    si,1Ah ; 26
00003A23 83EE1A                     SUB     SI,dir_entry.dir_first;Point back to start of dir entry
00003A26 C604E5                     MOV     BYTE [SI],0E5H ; Free the entry
00003A29 E8BDFF                     CALL    DIRUP           ; Error doesn't matter since erroring anyway
                                NODEEXISTS:
00003A2C E9F9FE                     JMP     NODEACCERRJ ; 10/08/2018

                                NODEEXISTSP:
00003A2F 5E                         POP     SI              ; Clean stack
00003A30 EBFA                       JMP     short NODEEXISTS

                                ; 17/11/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)

                                ;BREAK <DOS_ChDir -- Verify a directory>
                                ;--------------------------------------------------------------------------
```

```
                                ;
                                ; Procedure Name : DOS_ChDir
                                ;
                                ; Inputs:
                                ;   [WFP_START] Points to WFP string ("d:/" must be first 3 chars, NUL
                                ;           terminated)
                                ;   [CURR_DIR_END] Points to end of Current dir part of string
                                ;           ( = -1 if current dir not involved, else
                                ;            Points to first char after last "/" of current dir part)
                                ;   [THISCDS] Points to CDS being used May not be NUL
                                ; Function:
                                ;   Validate the path for potential new current directory
                                ; Returns:
                                ;   NOTE:
                                ;       [SATTRIB] is modified by this call
                                ;   Carry Clear
                                ;       CX is cluster number of the DIR, LOCAL CDS ONLY
                                ;          Caller must NOT set ID fields on a NET CDS.
                                ;   Carry Set
                                ;       AX is error code
                                ;           error_path_not_found
                                ;                   Bad path
                                ;           error_access_denied
                                ;                   device or file name
                                ; DS preserved, Others destroyed
                                ;--------------------------------------------------------------------------

                                ;hkn; called from path.asm and dir2.asm. DS already set up.

                                ; 18/05/2019 - Retro DOS v4.0
                                ; DOSCODE:70DAh (MSDOS 6.21, MSDOS.SYS)

                                ; 17/11/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
                                ; DOSCODE:70C6h (MSDOS 5.0, MSDOS.SYS)

                                ; 04/02/2024 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMDOS.COM)
                                ; PCDOS 7.1 IBMDOS.COM - DOSCODE:7A23h
                                ; (Windows ME IO.SYS - BIOSCODE:7839h)
                                ; (MSDOS 6.22 MSDOS.SYS - DOSCODE:70DAh)

                                DOS_CHDIR:
00003A32 E8F4DD                     call    TestNet
00003A35 7306                       JNC     short LOCAL_CHDIR

                                ;IF NOT Installed
                                ;   transfer NET_CHDIR
                                ;ELSE
                                    ;mov    ax,1105h
00003A37 B80511                     MOV     AX,(MultNET<<8)|5
00003A3A CD2F                       int     2Fh     ; Multiplex - NETWORK REDIRECTOR - CHDIR
                                                    ; SS = DOS CS
                                                    ; SDA first filename pointer -> fully-qualified directory name
                                                    ; SDA CDS pointer -> current directory
                                                    ; Return: CF set on error, AX = DOS error code
                                                    ; CF clear if successful
00003A3C C3                         retn
                                ;ENDIF

                                LOCAL_CHDIR:
00003A3D E8A2DE                     call    ECritDisk
                                    ; MSDOS 6.0
                                    ;;test  word [es:di+43h],2000h
                                    ;TEST   word [ES:DI+curdir.flags],curdir_splice ;PTM.
                                    ; 17/12/2022
                                    ;test   byte [es:di+44h],20h
00003A40 26F6454420                 test    byte [ES:DI+curdir.flags+1],(curdir_splice>>8) ;PTM.
00003A45 740C                       JZ      short nojoin            ;PTM.
                                    ;mov    word [es:di+49h], 0FFFFh
00003A47 26C74549FFFF               MOV     word [ES:DI+curdir.ID],0FFFFH ;PTM.
                                    ;;;
                                    ; 04/02/2024 (PCDOS 7.1 IBMDOS.COM)
                                    ;mov    word [es:di+4Bh], 0FFFFh
00003A4D 26C7454BFFFF               mov     word [es:di+curdir.ID+2],0FFFFh
                                    ;;;
                                nojoin:
                                    ; MSDOS 3.3 & MSDOS 6.0
00003A53 C606[4C03]00               MOV     byte [NoSetDir],0 ; FALSE
                                    ;mov    byte [SATTRIB],16h
00003A58 C606[6D05]16               MOV     byte [SATTRIB],attr_directory+attr_system+attr_hidden
                                                            ; Dir calls can find these
                                ; DOS 3.3  6/24/86 FastOpen
00003A5D 800E[4611]01               OR      byte [FastOpenFlg],FastOpen_Set      ; set fastopen flag
00003A62 E8AE10                     call    GETPATH

                                    ; 04/02/2024
                                    ;PUSHF                                      ;AN000;
00003A65 9F                         lahf
00003A66 8026[4611]80               AND     byte [FastOpenFlg],Fast_yes ; clear it all ;AC000;
                                    ;POPF                                       ;AN000;
00003A6B 9E                         sahf

                                ; DOS 3.3  6/24/86 FastOpen

                                    ; MSDOS 3.3
                                    ;mov    byte [FastOpenFlg],0

                                    ;mov    ax,3
00003A6C B80300                     MOV     AX,error_path_not_found
00003A6F 7206                       JC      short ChDirDone
00003A71 7556                       JNZ     short NOTDIRPATH        ; Path not a DIR
00003A73 8B0E[C205]                 MOV     CX,[DIRSTART]           ; Get cluster number
                                    ; 27/06/2024
                                    ;CLC
                                ChDirDone:
                                    ;call   LCritDisk
                                    ;retn
                                    ; 18/12/2022
00003A77 E995DE                     jmp     LCritDisk

                                ;BREAK <DOS_RmDir -- Remove a directory>
                                ;--------------------------------------------------------------------------
                                ;
                                ; Procedure Name : DOS_RmDir
                                ;
                                ; Inputs:
                                ;   [WFP_START] Points to WFP string ("d:/" must be first 3 chars, NUL
                                ;           terminated)
                                ;   [CURR_DIR_END] Points to end of Current dir part of string
                                ;           ( = -1 if current dir not involved, else
                                ;            Points to first char after last "/" of current dir part)
                                ;   [THISCDS] Points to CDS being used
                                ;           (Low word = -1 if NUL CDS (Net direct request))
                                ; Function:
```

```
23794                                  ;   Remove a directory
23795                                  ;   NOTE: Attempt to remove current directory must be detected by caller
23796                                  ; Returns:
23797                                  ;   NOTE:
23798                                  ;       [SATTRIB] is modified by this call
23799                                  ;   Carry Clear
23800                                  ;           No error
23801                                  ;   Carry Set
23802                                  ;       AX is error code
23803                                  ;           error_path_not_found
23804                                  ;                   Bad path (not in curr dir part if present)
23805                                  ;           error_bad_curr_dir
23806                                  ;                   Bad path in current directory part of path
23807                                  ;           error_access_denied
23808                                  ;                   device or file name, root directory
23809                                  ;                   Bad directory ('.' '..' messed up)
23810                                  ; DS preserved, Others destroyed
23811                                  ;--------------------------------------------------------------------------
23812
23813                                  ;hkn; called from path.asm. DS already set up.
23814
23815                                  ; 18/05/2019 - Retro DOS v4.0
23816                                  ; DOSCODE:711Fh (MSDOS 6.21, MSDOS.SYS)
23817
23818                                  ; 17/11/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
23819                                  ; DOSCODE:710Bh (MSDOS 5.0, MSDOS.SYS)
23820
23821                                  ; 06/02/2024 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMDOS.COM)
23822                                  ; PCDOS 7.1 IBMDOS.COM - DOSCODE:7A6Bh
23823
23824                                  DOS_RMDIR:
23825 00003A7A E8ACDD                      call    TestNet
23826 00003A7D 7306                        JNC     short LOCAL_RMDIR
23827
23828                                  ;IF NOT Installed
23829                                  ;   transfer NET_RMDIR
23830                                  ;ELSE
23831                                      ;mov    ax,1101h
23832 00003A7F B80111                      MOV     AX,(MultNET<<8)|1
23833 00003A82 CD2F                        int     2Fh     ; Multiplex - NETWORK REDIRECTOR - REMOVE REMOTE DIRECTORY
23834                                                      ; SS = DOS CS
23835                                                      ; SDA first filename pointer -> fully-qualified directory name
23836                                                      ; SDA CDS pointer -> current directory
23837                                                      ; Return: CF set on error, AX = DOS error code
23838                                                      ; CF clear if successful
23839 00003A84 C3                          retn
23840                                  ;ENDIF
23841
23842                                  LOCAL_RMDIR:
23843 00003A85 E85ADE                      call    ECritDisk
23844 00003A88 C606[4C03]00                MOV     byte [NoSetDir],0
23845                                      ;mov    byte [SATTRIB],16h
23846 00003A8D C606[6D05]16                MOV     byte [SATTRIB],attr_directory+attr_system+attr_hidden
23847                                                                      ; Dir calls can find these
23848 00003A92 E87E10                      call    GETPATH
23849 00003A95 7229                        JC      short NOPATH            ; Path not found
23850 00003A97 7530                        JNZ     short NOTDIRPATH        ; Path not a DIR
23851
23852                                  ; 06/04/2024
23853                                  %if 0
23854                                      MOV     DI,[DIRSTART]
23855                                      OR      DI,DI                   ; Root ?
23856                                      JNZ     short rmdir_get_buf     ; No
23857                                      JMP     SHORT NOTDIRPATH
23858                                  %else
23859                                      ; 06/02/2024 - Retro DOS v5.0
23860                                      ; PCDOS 7.1 IBMDOS.COM
23861 00003A99 8B3E[C205]                  mov     di,[DIRSTART]
23862 00003A9D 09FF                        or      di,di                   ; Root ?
23863 00003A9F 7506                        jnz     short LOCAL_RMDIR_cont; No
23864
23865                                      ;cmp    word [DIRSTART_HW],0
23866 00003AA1 393E[DC0A]                  cmp     [DIRSTART_HW],di ; 0
23867 00003AA5 7422                        jz       short NOTDIRPATH ; root directory
23868
23869                                  LOCAL_RMDIR_cont:
23870                                      ;cmp    word [es:bp+0Fh],0
23871 00003AA7 26837E0F00                  cmp     word [es:bp+DPB.FAT_SIZE],0
23872 00003AAC 751E                        jnz     short rmdir_get_buf     ; not FAT32
23873                                      ;cmp    di,[es:bp+35h]
23874 00003AAE 263B7E35                    cmp     di,[es:bp+DPB.ROOT_CLUSTER]
23875 00003AB2 7518                        jne     short rmdir_get_buf
23876 00003AB4 8B3E[DC0A]                  mov     di,[DIRSTART_HW]
23877                                      ;cmp    di,[es:bp+37h]
23878 00003AB8 263B7E37                    cmp     di,[es:bp+DPB.ROOT_CLUSTER+2]
23879 00003ABC 750E                        jne     short rmdir_get_buf
23880 00003ABE EB09                        jmp     short NOTDIRPATH
23881                                  %endif
23882
23883                                  NOPATH:
23884                                      ;mov    ax,3
23885 00003AC0 B80300                      MOV     AX,error_path_not_found
23886 00003AC3 E965FE                      JMP     _BadRet
23887
23888                                  NOTDIRPATHPOP:
23889 00003AC6 58                          POP     AX  ; MSDOS 6.0              ;F.C. >32mb
23890 00003AC7 58                          POP     AX
23891                                  NOTDIRPATHPOP2:
23892 00003AC8 58                          POP     AX
23893                                  NOTDIRPATH:
23894 00003AC9 E95CFE                      JMP     NODEACCERRJ
23895
23896                                  rmdir_get_buf:
23897 00003ACC C53E[E205]                  LDS     DI,[CURBUF]
23898 00003AD0 29FB                        SUB     BX,DI           ; Compute true offset
23899 00003AD2 53                          PUSH    BX              ; Save entry pointer
23900
23901                                  ; 06/04/2024
23902                                  %if 0
23903                                      ; MSDOS 6.0
23904                                      ;push   word [di+8]
23905                                      PUSH    WORD [DI+BUFFINFO.buf_sector+2] ;F.C. >32mb
23906
23907                                      ; MSDOS 3.3 (& MSDOS 6.0)
23908                                      ;push   word [di+6]
23909                                      PUSH    WORD [DI+BUFFINFO.buf_sector] ; Save sector number
23910                                  %else
23911                                      ; 06/02/2024 - Retro DOS v5.0
23912                                      ; PCDOS 7.1 IBMDOS.COM
23913
23914                                      ;les    di,[di+6]
23915 00003AD3 C47D06                      les     di,[di+BUFFINFO.buf_sector]
23916 00003AD6 06                          push    es
23917 00003AD7 57                          push    di
```

```
                                %endif

                                ;hkn; SS is DOSDATA
                                    ;context DS
00003AD8 16                         push    ss
00003AD9 1F                         pop     ds
                                    ;context ES
00003ADA 16                         push    ss
00003ADB 07                         pop     es

                                ;hkn; NAME1 is in DOSDATA
00003ADC BF[4B05]                   MOV     DI,NAME1
                                    ;MOV    AL,'?' ; 3Fh
                                    ;MOV    CX,11
                                    ;REP    STOSB
                                    ;XOR    AL,AL
                                    ;STOSB                          ; Nul terminate it
                                    ; 27/06/2024
00003ADF B83F00                     mov     ax,3Fh
00003AE2 B90A00                     mov     cx,10
00003AE5 F3AA                       rep     stosb   ; al = "?"
00003AE7 AB                         stosw           ; ah = 0
                                    ;
00003AE8 E8C812                     call    STARTSRCH               ; Set search
00003AEB E8B10D                     call    GETENTRY                ; Get start of directory
00003AEE 72D6                       JC      short NOTDIRPATHPOP     ; Screw up
00003AF0 8E1E[E405]                 MOV     DS,[CURBUF+2]
00003AF4 89DE                       MOV     SI,BX
00003AF6 AD                         LODSW
                                    ;CMP    AX,(' ' SHL 8) OR '.'   ; First entry '.'?
00003AF7 3D2E20                     cmp     ax,202Eh ; ". "
00003AFA 75CA                       JNZ     short NOTDIRPATHPOP     ; Nope
                                    ;add    si,30
00003AFC 83C61E                     ADD     SI,dir_entry.size-2 ; Next entry
00003AFF AD                         LODSW
                                    ;CMP    AX,('.' SHL 8) OR '.'   ; Second entry '..'?
                                    ;cmp    ax, '..'
00003B00 3D2E2E                     cmp     ax,2E2Eh
00003B03 75C1                       JNZ     short NOTDIRPATHPOP     ; Nope

                                ;hkn; SS is DOSDATA
                                    ;context DS
00003B05 16                         push    ss
00003B06 1F                         pop     ds
00003B07 C706[4803]0200             MOV     word [LASTENT],2        ; Skip . and ..
00003B0D E88F0D                     call    GETENTRY                ; Get next entry
00003B10 72B4                       JC      short NOTDIRPATHPOP     ; Screw up
                                    ;mov    byte [ATTRIB],16h
00003B12 C606[6B05]16               MOV     byte [ATTRIB],attr_directory+attr_hidden+attr_system
00003B17 E83B0C                     call    SRCH                    ; Do a search
00003B1A 73AA                       JNC     short NOTDIRPATHPOP     ; Found another entry!
00003B1C 803E[4A03]00               CMP     byte [FAILERR],0
00003B21 75A3                       JNZ     short NOTDIRPATHPOP     ; Failure of search due to I 24 FAIL
00003B23 C42E[8A05]                 LES     BP,[THISDPB]
                                    ;;;
                                    ; 06/02/2024 - Retro DOS v5.0
                                    ; PCDOS 7.1 IBMDOS.COM
00003B27 8B1E[DC0A]                 mov     bx,[DIRSTART_HW]
00003B2B 891E[E80A]                 mov     [CLUSTNUM_HW],bx
                                    ;;;
00003B2F 8B1E[C205]                 MOV     BX,[DIRSTART]
00003B33 E89A20                     call    RELEASE                 ; Release data in sub dir
00003B36 728E                       JC      short NOTDIRPATHPOP     ; Screw up
                                    ;;;
                                    ; 06/02/2024 - Retro DOS v5.0 (PCDOS 7.1 IBMDOS.COM)
00003B38 E831F9                     call    update_fat32_fsinfo
                                    ;;;
00003B3B 5A                         POP     DX                      ; Sector # of entry
00003B3C 8F06[0706]                 POP     word [HIGH_SECTOR] ; MSDOS 6.0       ; F.C. >32mb
                                    ; 22/09/2023
                                    ;;mov   byte [ALLOWED],18h
                                    ;MOV    byte [ALLOWED],Allowed_FAIL+Allowed_RETRY ; *
                                    ;XOR    AL,AL ; *               ; Pre read
                                    ;call   GETBUFFR                ; Get sector back
00003B40 E8E62D                     call    GETBUFFER ; *           ; Pre Read
00003B43 7283                       JC      short NOTDIRPATHPOP2    ; Screw up

                                rmdir_fde:  ; 06/02/2024
00003B45 C53E[E205]                 LDS     DI,[CURBUF]
                                    ;or     byte [di+5],4
00003B49 804D0504                   OR      byte [DI+BUFFINFO.buf_flags],buf_isDIR
00003B4D 5B                         POP     BX                      ; Pointer to start of entry
00003B4E 01FB                       ADD     BX,DI                   ; Corrected
00003B50 C607E5                     MOV     BYTE [BX],0E5H          ; Free the entry

                                ;DOS 3.3 FastOpen  6/16/86  F.C.
00003B53 1E                         PUSH    DS

                                ;hkn; SS is DOSDATA
                                    ;context DS
00003B54 16                         push    ss
00003B55 1F                         pop     ds

                                    ; MSDOS 6.0
00003B56 E8E6F1                     call    FastOpen_Delete         ; call fastopen to delete an entry

                                ;   ; MSDOS 3.3
                                ;_FastOpen_Delete:
                                ;   push    ax
                                ;   mov     si,[WFP_START]
                                ;   mov     bx,FastTable
                                ;   ;mov    al,3  ; FONC_delete
                                ;   mov     al,FONC_delete
                                ;   call    far [BX+2]  ; FastTable+2
                                ;   pop     ax

00003B59 1F                         POP     DS
                                ;DOS 3.3 FastOpen  6/16/86  F.C.

00003B5A E98CFE                     JMP     DIRUP                   ; In MKDIR, dirty buffer and flush

                                ;==========================================================================
                                ; DISK.ASM, MSDOS 6.0, 1991
                                ;==========================================================================
                                ; 23/07/2018 - Retro DOS v3.0
                                ; 04/05/2019 - Retro DOS v4.0
                                ; 06/02/2024 - Retro DOS v5.0

                                ;   TITLE   DISK - Disk utility routines
                                ;   NAME    Disk

                                ;** Low level Read and write routines for local SFT I/O on files and devs
                                ;
                                ;   SWAPCON
```

```
24042                                   ;   SWAPBACK
24043                                   ;   DOS_READ
24044                                   ;   DOS_WRITE
24045                                   ;   get_io_sft
24046                                   ;   DirRead
24047                                   ;   FIRSTCLUSTER
24048                                   ;   SET_BUF_AS_DIR
24049                                   ;   FATSecRd
24050                                   ;   DREAD
24051                                   ;   CHECK_WRITE_LOCK
24052                                   ;   CHECK_READ_LOCK
24053                                   ;
24054                                   ;   Revision history:
24055                                   ;
24056                                   ;           A000   version 4.00  Jan. 1988
24057                                   ;
24058                                   ;--------------------------------------------------------------------------
24059                                   ;
24060                                   ; M065 : B#5276. On raw read/write of a block of characters if a critical
24061                                   ;           error happens, DOS retries the entire block assuming that
24062                                   ;           zero characters were transferred. Modified the code to take
24063                                   ;           into account the number of characters transfered before
24064                                   ;           retrying the operation.
24065                                   ;
24066                                   ;--------------------------------------------------------------------------
24067                                   ;
24068
24069                                   ;Installed = TRUE
24070
24071                                   ;Break      <SwapCon, Swap Back - Old-style I/O to files>
24072
24073                                   ; **** Drivers for file input from devices ****
24074                                   ;--------------------------------------------------------------------------
24075                                   ;   Indicate that there is no more I/O occurring through another SFT outside
24076                                   ;   of handles 0 and 1
24077                                   ;
24078                                   ;   Inputs: DS is DOSDATA
24079                                   ;   Outputs:       CONSWAP is set to false.
24080                                   ;   Registers modified: none
24081                                   ;--------------------------------------------------------------------------
24082
24083                                   ; IBMDOS.COM (MSDOS 3.3) - Offset 3CF8h
24084
24085                                   ; DOSCODE:71E3h (MSDOS 6.21, MSDOS.SYS)
24086                                   ; 04/05/2019 - Retro DOS v4.0
24087
24088                                   ; DOSCODE:71CFh (MSDOS 5.0, MSDOS.SYS)
24089                                   ; 17/11/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
24090
24091                                   SWAPBACK:
24092 00003B5D C606[5703]00               MOV     BYTE [CONSWAP],0      ; signal no conswaps
24093 00003B62 C3                         retn
24094
24095                                   ;--------------------------------------------------------------------------
24096                                   ;
24097                                   ; Procedure Name : SWAPCON
24098                                   ;
24099                                   ;   Copy ThisSFT to CONSFT for use by the 1-12 primitives.
24100                                   ;
24101                                   ;   Inputs: ThisSFT as the sft of the desired file
24102                                   ;           DS is DOSDATA
24103                                   ;   Outputs:       CONSWAP is set.  CONSFT = ThisSFT.
24104                                   ;   Registers modified: none
24105                                   ;--------------------------------------------------------------------------
24106
24107                                   SWAPCON:
24108                                       ; MSDOS 3.3
24109                                       ;push   es
24110                                       ;push   di
24111                                       ;mov    byte [CONSWAP],1
24112                                       ;les    di,[THISSFT]
24113                                       ;mov    word [CONSFT],di
24114                                       ;mov    word [CONSFT+2],es
24115                                       ;pop    di
24116                                       ;pop    es
24117                                       ;retn
24118
24119                                       ; MSDOS 6.0
24120 00003B63 C606[5703]01               mov     byte [CONSWAP],1      ; ConSwap = TRUE
24121 00003B68 50                         push    ax
24122 00003B69 A1[9E05]                   mov     ax,[THISSFT]
24123 00003B6C A3[E605]                   mov     [CONSFT],ax
24124 00003B6F A1[A005]                   mov     ax,[THISSFT+2]
24125 00003B72 A3[E805]                   mov     [CONSFT+2],ax
24126 00003B75 58                         pop     ax
24127 00003B76 C3                         retn
24128
24129                                   ; DOSCODE:71FDh (MSDOS 6.21, MSDOS.SYS)
24130                                   ; 04/05/2019 - Retro DOS v4.0
24131
24132                                   ;Break      <DOS_READ -- MAIN READ ROUTINE AND DEVICE IN ROUTINES>
24133                                   ;--------------------------------------------------------------------------
24134                                   ;
24135                                   ; Inputs:
24136                                   ;   ThisSFT set to the SFT for the file being used
24137                                   ;   [DMAADD] contains transfer address
24138                                   ;   CX = No. of bytes to read
24139                                   ;   DS = DOSDATA
24140                                   ; Function:
24141                                   ;   Perform read operation
24142                                   ; Outputs:
24143                                   ;    Carry clear
24144                                   ;   SFT Position and cluster pointers updated
24145                                   ;   CX = No. of bytes read
24146                                   ;   ES:DI point to SFT
24147                                   ;    Carry set
24148                                   ;   AX is error code
24149                                   ;   CX = 0
24150                                   ;   ES:DI point to SFT
24151                                   ; DS preserved, all other registers destroyed
24152                                   ;
24153                                   ;--------------------------------------------------------------------------
24154
24155                                   ;hkn; called from fcbio.asm, handle.asm and dev.asm. DS is be set up.
24156
24157                                   ; DOSCODE:71E9h (MSDOS 5.0, MSDOS.SYS)
24158                                   ; 17/11/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
24159
24160                                   ; 07/02/2024
24161                                   ; 06/02/2024 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMDOS.COM)
24162                                   ; PCDOS 7.1 IBMDOS.COM - DOSCODE:7B76h
24163
24164                                   ; (Windows ME IO.SYS - BIOSCODE:7997h)
24165                                   ; (MSDOS 6.22 MSDOS.SYS - DOSCODE:71FDh)
```

```
                                DOS_READ:
00003B77 C43E[9E05]                 LES     DI,[THISSFT]

                                ; Verify that the sft has been opened in a mode that allows reading.

                                    ;mov    al,[es:di+2]
00003B7B 268A4502                   MOV     AL,[ES:DI+SF_ENTRY.sf_mode]
                                    ;;and   al,0Fh  ; (MSDOS 5.0 & 6.22)
                                    ;AND    AL,access_mask
                                    ; 06/02/2024 - Retro DOS v5.0
                                    ; (PCDOS 7.1 & Win Me)
00003B7F 2403                       and     al,3    ; open_mode_mask ?

                                    ;cmp    al,1
00003B81 3C01                       CMP     AL,open_for_write
00003B83 7503                       JNE     short READ_NO_MODE    ; Is read or both
00003B85 E96406                     jmp     SET_ACC_ERR

                                READ_NO_MODE:
00003B88 E82405                     call    SETUP
00003B8B E30B                       JCXZ    NoIORet               ; no bytes to read - fast return
00003B8D E8B2DC                     call    IsSFTNet
00003B90 7408                       JZ      short LOCAL_READ

                                ;IF NOT Installed
                                ;   transfer NET_READ
                                ;ELSE
                                    ;mov    ax,1108h
00003B92 B80811                     MOV     AX,(MultNET<<8)|8
00003B95 CD2F                       int     2Fh     ; Multiplex - NETWORK REDIRECTOR - READ FROM REMOTE FILE
                                                    ; ES:DI -> SFT
                                                    ; SFT DPB field -> DPB of drive containing file
                                                    ; CX = number of bytes, SS = DOS CS, SDA DTA field -> user buffer
                                                    ; Return: CF set on error, CX = bytes read
00003B97 C3                         retn
                                ;ENDIF

                                ; The user ended up requesting 0 bytes of input. We do nothing for this case
                                ; except return immediately.

                                NoIORet:
00003B98 F8                         CLC
00003B99 C3                         retn

                                LOCAL_READ:
                                    ;test   word [es:di+5],80h
                                    ;TEST   word [ES:DI+SF_ENTRY.sf_flags],devid_device  ; Check for named device I/O
00003B9A 26F6450580                 test    byte [ES:DI+SF_ENTRY.sf_flags],devid_device ; 02/06/2019
00003B9F 750E                       JNZ     short READDEV

                                    ;mov    byte [EXTERR_LOCUS],2
00003BA1 C606[2303]02               MOV     byte [EXTERR_LOCUS],errLOC_Disk
00003BA6 E839DD                     call    ECritDisk
00003BA9 E8F705                     call    DISKREAD

                                critexit:
                                    ;call   LCritDisk
                                    ;retn
                                    ; 16/12/2022
00003BAC E960DD                     jmp     LCritDisk

                                ; We are reading from a device. Examine the status of the device to see if we
                                ; can short-circuit the I/O. If the device in the EOF state or if it is the
                                ; null device, we can safely indicate no transfer.

                                READDEV:
                                    ;mov    byte [EXTERR_LOCUS],4
00003BAF C606[2303]04               MOV     byte [EXTERR_LOCUS],errLOC_SerDev
                                    ;mov    bl,[es:di+5]
00003BB4 268A5D05                   MOV     BL,[ES:DI+SF_ENTRY.sf_flags]
00003BB8 C43E[2C03]                 LES     DI,[DMAADD]
                                    ;test   bl,40h
00003BBC F6C340                     test    BL,devid_device_EOF   ; End of file?
00003BBF 7407                       JZ      short ENDRDDEVJ3
                                    ;test   bl,4
00003BC1 F6C304                     test    BL,devid_device_null  ; NUL device?
00003BC4 7405                       JZ      short TESTRAW         ; NO
00003BC6 30C0                       XOR     AL,AL                 ; Indicate EOF by setting zero
                                ENDRDDEVJ3:
                                    ; 17/11/2022 (MSDOS 5.0 MSDOS.SYS compatibility!)
                                    ;JMP    short ENDRDDEVJ2
                                    ; 16/12/2022
00003BC8 E93F01                     jmp     ENDRDDEV ; 04/05/2019

                                ; We need to hit the device. Figure out if we do a raw read or we do the
                                ; bizarre std_con_string_input.

                                TESTRAW:
                                    ;test   bl,20h
00003BCB F6C320                     test    BL,devid_device_raw   ; Raw mode?
00003BCE 7508                       JNZ     short DVRDRAW         ; Yes, let the device do all local editing
                                    ;test   bl,1
00003BD0 F6C301                     test    BL,devid_device_con_in; Is it console device?
00003BD3 7458                       JZ      short NOTRDCON
00003BD5 E96701                     JMP     READCON

                                DVRDRAW:
00003BD8 06                         PUSH    ES
00003BD9 1F                         POP     DS                    ; Xaddr to DS:DI

                                    ; 04/05/2019 - Retro DOS v4.0

                                    ; MSDOS 6.0
                                ;SR;
                                ;Check for win386 presence -- if present, do polled read of characters

00003BDA 36F606[5B0F]01             test    byte [ss:IsWin386],1 ; 19/05/2019
00003BE0 7408                       jz      short ReadRawRetry    ;not present
00003BE2 F6C301                     test    bl,devid_device_con_in;is it console device?
00003BE5 7403                       jz      short ReadRawRetry    ;no, do normal read
00003BE7 E9A800                     jmp     do_polling            ;yes, do win386 polling loop

                                ReadRawRetry:

                                ; 07/02/2024
                                %if 0
                                    MOV     BX,DI                 ; DS:BX transfer addr
                                    ; 06/02/2024 ; *
                                    ;XOR    AX,AX                 ; Media Byte, unit = 0
                                    ;;MOV   DX,AX                 ; Start at 0
                                    ;; 06/02/2024
                                    ;cwd
                                    ;call   SETREAD
```

```
                                    ; 06/02/2024 ; *
                                    call    SETREAD_X
                                %else
00003BEA E81B17                     call    SETREAD_XJ
                                %endif

00003BED 1E                         PUSH    DS                      ; Save Seg part of Xaddr

                                ;hkn; SS override
00003BEE 36C536[9E05]               LDS     SI,[SS:THISSFT]
00003BF3 E89C16                     call    DEVIOCALL
00003BF6 89FA                       MOV     DX,DI                   ; DS:DX is preserved by INT 24
00003BF8 B486                       MOV     AH,86H                  ; Read error

                                ;hkn; SS override
00003BFA 368B3E[5D03]               MOV     DI,[SS:DEVCALL_REQSTAT]
                                    ; MSDOS 3.3
                                    ;test  di,8000h
                                    ;jz    short CRDROK
                                    ; MSDOS 6.0
00003BFF 09FF                       or      di,di
00003C01 7920                       jns     short CRDROK            ; no errors
                                    ; MSDOS 3.3 (& MSDOS 6.0)
00003C03 E81D24                     call    CHARHARD

                                ; 06/02/2024 - Retro DOS v5.0
                                %if 0
                                    MOV     DI,DX                   ; DS:DI is Xaddr
                                    ; 04/05/2019
                                    ; MSDOS 6.0
                                    add     di,[ss:CALLSCNT]        ; update ptr and count to reflect the       M065
                                    sub     cx,[ss:CALLSCNT]        ; number of chars xferred           M065
                                %else
00003C06 368B3E[6C03]               mov     di,[ss:CALLSCNT]
00003C0B 29F9                       sub     cx,di                   ; update transfer count
00003C0D 01D7                       add     di,dx                   ; update pointer
                                %endif
                                    ; MSDOS 3.3 (& MSDOS 6.0)
00003C0F 08C0                       OR      AL,AL
00003C11 7410                       JZ      short CRDROK            ; Ignore
00003C13 3C03                       CMP     AL,3
00003C15 7403                       JZ      short CRDFERR           ; fail.
00003C17 1F                         POP     DS                      ; Recover saved seg part of Xaddr
00003C18 EBD0                       JMP     short ReadRawRetry      ; Retry

                                ; We have encountered a device-driver error. We have informed the user of it
                                ; and he has said for us to fail the system call.

                                CRDFERR:
00003C1A 5F                         POP     DI                      ; Clean stack
                                DEVIOFERR:

                                ;hkn; SS override
00003C1B 36C43E[9E05]               LES     DI,[SS:THISSFT]
00003C20 E9C705                     jmp     SET_ACC_ERR_DS

                                CRDROK:
00003C23 5F                         POP     DI                      ; Chuck saved seg of Xaddr
00003C24 89D7                       MOV     DI,DX

                                ;hkn; SS override
00003C26 36033E[6C03]               ADD     DI,[ss:CALLSCNT]        ; Amount transferred
                                    ;JMP    SHORT ENDRDDEVJ3
                                    ; 16/12/2022
00003C2B EB63                       jmp     short ENDRDDEVJ2

                                ; We are going to do a cooked read on some character device. There is a
                                ; problem here, what does the data look like? Is it a terminal device, line
                                ; CR line CR line CR, or is it file data, line CR LF line CR LF? Does it have
                                ; a ^Z at the end which is data, or is the ^Z not data?  In any event we're
                                ; going to do this: Read in pieces up to CR (CRs included in data) or ^z (^z
                                ; included in data). this "simulates" the way con works in cooked mode
                                ; reading one line at a time. With file data, however, the lines will look
                                ; like, LF line CR. This is a little weird.

                                NOTRDCON:
                                    ;MOV    AX,ES
                                    ;MOV    DS,AX
                                    ; 07/02/2024
00003C2D 06                         push    es
00003C2E 1F                         pop     ds

                                ; 07/02/2024
                                %if 0
                                    MOV     BX,DI
                                    ; 06/02/2024 ; *
                                    ;;XOR   DX,DX
                                    ;;MOV   AX,DX
                                    ;; 06/02/2024
                                    ;xor    ax,ax
                                    ;cwd
                                    PUSH    CX
                                    MOV     CX,1
                                    ;call   SETREAD
                                    ; 06/02/2024 ; *
                                    call    SETREAD_X
                                    POP     CX
                                %else
00003C2F 51                         push    cx
00003C30 B90100                     mov     cx,1
00003C33 E8D216                     call    SETREAD_XJ
00003C36 59                         pop     cx
                                %endif

                                ;hkn; SS override
00003C37 36C536[9E05]               LDS     SI,[SS:THISSFT]
                                    ;lds    si,[si+7]
00003C3C C57407                     LDS     SI,[SI+SF_ENTRY.sf_devptr]
                                DVRDLP:
00003C3F E8C121                     call    DSKSTATCHK
00003C42 E85016                     call    DEVIOCALL2
00003C45 57                         PUSH    DI                      ; Save "count" done
00003C46 B486                       MOV     AH,86H

                                ;hkn; SS override
00003C48 368B3E[5D03]               MOV     DI,[SS:DEVCALL_REQSTAT]

                                    ; MSDOS 3.3
                                    ;test  di,8000h
                                    ;jz    short CRDOK
                                    ; MSDOS 6.0
00003C4D 09FF                       or      di,di
00003C4F 7917                       jns     short CRDOK
```

```
24414 00003C51 E8CF23                  call    CHARHARD
24415 00003C54 5F                      POP     DI
24416
24417                                 ;hkn; SS override
24418 00003C55 36C706[6C03]0100        MOV     word [SS:CALLSCNT],1
24419 00003C5C 3C01                    CMP     AL,1
24420 00003C5E 74DF                    JZ      short DVRDLP          ; Retry
24421 00003C60 3C03                    CMP     AL,3
24422 00003C62 74B7                    JZ      short DEVIOFERR               ; FAIL
24423 00003C64 30C0                    XOR     AL,AL                 ; Ignore, Pick some random character
24424 00003C66 EB12                    JMP     SHORT DVRDIGN
24425
24426                                 CRDOK:
24427 00003C68 5F                      POP     DI
24428
24429                                 ;hkn; SS override
24430 00003C69 36833E[6C03]01          CMP     word [SS:CALLSCNT],1
24431                                  ; 17/11/2022 (MSDOS 5.0 MSDOS.SYS compatibility!)
24432 00003C6F 751F                    JNZ     short ENDRDDEVJ2
24433                                  ; 16/12/2022
24434                                  ;jnz    short ENDRDDEV ; 24/07/2019
24435
24436 00003C71 1E                      PUSH    DS
24437
24438                                 ;hkn; SS override
24439 00003C72 368E1E[6A03]            MOV     DS,[SS:CALLXAD+2]
24440 00003C77 8A05                    MOV     AL,[DI]               ; Get the character we just read
24441 00003C79 1F                      POP     DS
24442                                 DVRDIGN:
24443
24444                                 ;hkn; SS override
24445 00003C7A 36FF06[6803]            INC     WORD [SS:CALLXAD]     ; Next character
24446 00003C7F 36C706[5D03]0000        MOV     word [SS:DEVCALL_REQSTAT],0
24447 00003C86 47                      INC     DI                    ; Next character
24448 00003C87 3C1A                    CMP     AL,1Ah                ; ^Z?
24449                                  ; 17/11/2022 (MSDOS 5.0 MSDOS.SYS compatibility!)
24450 00003C89 7405                    JZ      short ENDRDDEVJ2      ; Yes, done zero set (EOF)
24451                                  ; 16/12/2022
24452                                  ;jz     short ENDRDDEV ; 24/07/2019
24453 00003C8B 3C0D                    CMP     AL,c_CR  ; 0Dh        ; CR?
24454 00003C8D E0B0                    LOOPNZ  DVRDLP                ; Loop if no, else done
24455 00003C8F 40                      INC     AX                    ; Resets zero flag so NOT EOF, unless
24456                                                                ;  AX=FFFF which is not likely
24457                                 ENDRDDEVJ2:
24458                                  ; 16/12/2022
24459                                  ;JMP    short ENDRDDEV        ; changed short to long for win386
24460                                  ; 17/11/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
24461 00003C90 EB78                    jmp     ENDRDDEV
24462
24463                                  ; 04/05/2019
24464
24465                                  ; MSDOS 6.0
24466                                 ;SR;
24467                                 ;Polling code for raw read on CON when WIN386 is present
24468                                 ;
24469                                 ;At this point -- ds:di is transfer address
24470                                 ;             cx is count
24471
24472                                 do_polling:
24473
24474                                 ; 07/02/2024
24475                                 %if 0
24476                                     mov     bx,di                ;ds:bx is Xfer address
24477                                     ; 06/02/2024 ; *
24478                                     ;xor    ax,ax
24479                                     ;;mov   dx,ax
24480                                     ;; 06/02/2024
24481                                     ;cwd
24482                                     ;call   SETREAD              ;prepare device packet
24483                                     ; 06/02/2024 ; *
24484                                     call    SETREAD_X
24485                                 %else
24486 00003C92 E87316                   call    SETREAD_XJ
24487                                 %endif
24488
24489                                 do_io:
24490                                 ;Change read to a NON-DESTRUCTIVE READ, NO WAIT
24491
24492 00003C95 26C6470205               mov     byte [es:bx+2],DEVRDND ; 5 ;Change command code
24493 00003C9A 1E                       push    ds
24494 00003C9B 36C536[9E05]             lds     si,[ss:THISSFT]              ;get device header
24495 00003CA0 E8EF15                   call    DEVIOCALL            ;call device driver
24496 00003CA3 1F                       pop     ds
24497
24498                                    ;test   word [es:bx+3],8000h
24499                                    ; 16/12/2022
24500                                    ;test   byte [es:bx+4],80h
24501 00003CA4 26F6470480               test    byte [es:bx+SRHEAD.REQSTAT+1],STERR>>8
24502                                    ;test   word [es:bx+SRHEAD.REQSTAT],STERR ;check if error
24503 00003CA9 7413                     jz      short check_busy      ;no
24504
24505 00003CAB 1E                       push    ds
24506 00003CAC 89FA                     mov     dx,di
24507
24508                                 invoke_charhard:   ; 07/02/2024
24509                                    ;invoke charhard             ;invoke int 24h handler
24510 00003CAE E87223                   call    CHARHARD
24511 00003CB1 89D7                     mov     di,dx
24512 00003CB3 08C0                     or      al,al
24513 00003CB5 744D                     jz      short pop_done_read   ;ignore by user, assume read done
24514 00003CB7 3C03                     cmp     al,3
24515 00003CB9 7438                     jz      short devrderr        ;user asked to fail
24516 00003CBB 1F                       pop     ds
24517 00003CBC EBD7                     jmp     short do_io           ;user asked to retry
24518
24519                                 check_busy:
24520                                    ;test   word [es:bx+3],200h
24521                                    ; 16/12/2022
24522 00003CBE 26F6470402               test    byte [es:bx+SRHEAD.REQSTAT+1],02h
24523                                    ;test   word [es:bx+SRHEAD.REQSTAT],0200h ;see if busy bit set
24524 00003CC3 7537                     jnz     short no_char         ;yes, no character available
24525
24526                                 ;Character is available. Read in 1 character at a time until all characters
24527                                 ;are read in or no character is available
24528
24529 00003CC5 26C6470204               mov     byte [es:bx+2],DEVRD ; 4 ;command code is READ now
24530 00003CCA 26C747120100             mov     word [es:bx+18],1     ;change count to 1 character
24531 00003CD0 1E                       push    ds
24532 00003CD1 36C536[9E05]             lds     si,[ss:THISSFT]
24533 00003CD6 E8B915                   call    DEVIOCALL
24534
24535 00003CD9 89FA                     mov     dx,di
24536 00003CDB B486                     mov     ah,86h
24537                                    ;mov    di,[es:bx+3]
```

```
24538 00003CDD 268B7F03                    mov     di,[es:bx+SRHEAD.REQSTAT] ;get returned status
24539 00003CE1 F7C70080                    test    di,STERR ; 8000h        ;was there an error during read?
24540                                      ;jz     short next_char                 ;no,read next character
24541                                      ; 07/02/2024
24542 00003CE5 75C7                        jnz     short invoke_charhard
24543
24544                                  ; 07/02/2024
24545                                  %if 0
24546                                      ;invoke charhard                ;invoke int 24h handler
24547                                      call    CHARHARD
24548                                      mov     di,dx                   ;restore di
24549                                      or      al,al                   ;
24550                                      jz      short pop_done_read     ;ignore by user,assume read is done
24551                                      cmp     al,3
24552                                      jz      short devrderr          ;user issued a 'fail',indicate error
24553                                      pop     ds
24554                                      jmp     short do_io             ;user issued a retry
24555                                  %endif
24556
24557                                  next_char:
24558 00003CE7 1F                          pop     ds
24559 00003CE8 89D7                        mov     di,dx
24560 00003CEA 49                          dec     cx                      ;decrement count
24561                                      ;jcxz   done_read               ;all characters read in
24562                                      ; 07/02/2024
24563 00003CEB 7418                        jz      short done_read
24564 00003CED 26FF470E                    inc     word [es:bx+14]                 ;update transfer address
24565 00003CF1 EBA2                        jmp     short do_io             ;read next character in
24566
24567                                  devrderr:
24568 00003CF3 5F                          pop     di                      ;discard segment address
24569 00003CF4 36C43E[9E05]                les     di,[ss:THISSFT]
24570                                      ;transfer SET_ACC_ERR_DS        ;indicate error
24571 00003CF9 E9EE04                      jmp     SET_ACC_ERR_DS
24572
24573                                  no_char:
24574                                  ;Since no character is available, we let win386 switch the VM out
24575
24576 00003CFC 50                           push    ax
24577 00003CFD B484                        mov     ah,84h  ; Microsoft Networks - KEYBOARD BUSY LOOP
24578 00003CFF CD2A                        int     2Ah                     ;indicate idle to WIN386
24579
24580                                  ;When control returns from WIN386, we continue the raw read
24581
24582 00003D01 58                           pop     ax
24583 00003D02 EB91                        jmp     short do_io    ; 27/06/2024
24584
24585                                  pop_done_read:
24586 00003D04 1F                          pop     ds
24587                                  done_read:
24588 00003D05 36033E[6C03]                add     di,[ss:CALLSCNT] ; 19/05/2019
24589
24590                                      ; 16/12/2022
24591
24592                                      ;jmp    ENDRDDEVJ3      ;jump back to normal DOS raw read exit
24593                                      ;jmp    ENDRDDEV ; 04/05/2019
24594
24595                                      ; 04/05/2019 - Retro DOS v4.0
24596                                  ENDRDDEV:
24597 00003D0A 16                          push    ss
24598 00003D0B 1F                          pop     ds
24599 00003D0C EB1F                        jmp     short endrddev1
24600
24601                                      ; 17/11/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
24602                                      ;jmp    ENDRDDEVJ3      ;jump back to normal DOS raw read exit
24603
24604                                  TRANBUF:
24605 00003D0E AC                          LODSB
24606 00003D0F AA                          STOSB
24607 00003D10 3C0D                        CMP     AL,c_CR ; 0Dh  ; Check for carriage return
24608 00003D12 7503                        JNZ     short NORMCH
24609 00003D14 C6040A                      MOV     BYTE [SI],c_LF ; 0Ah
24610                                  NORMCH:
24611 00003D17 3C0A                        CMP     AL,c_LF ; 0Ah
24612 00003D19 E0F3                        LOOPNZ  TRANBUF
24613 00003D1B 7507                        JNZ     short ENDRDCON
24614 00003D1D 31F6                        XOR     SI,SI           ; Cause a new buffer to be read
24615 00003D1F E80FDF                      call    OUTT            ; Transmit linefeed
24616 00003D22 0C01                        OR      AL,1            ; Clear zero flag--not end of file
24617                                  ENDRDCON:
24618                                  ;hkn; SS is DOSDATA
24619 00003D24 16                          push    ss
24620 00003D25 1F                          pop     ds
24621 00003D26 E834FE                      CALL    SWAPBACK
24622 00003D29 8936[2200]                  MOV     [CONTPOS],SI
24623
24624                                      ; 16/12/2022
24625                                  ;ENDRDDEV:
24626                                  ;;hkn; SS is DOSDATA
24627                                  ;   push    ss
24628                                  ;   pop     ds
24629                                  endrddev1:  ; 04/05/2019
24630 00003D2D 893E[B805]                  MOV     [NEXTADD],DI
24631 00003D31 7509                        JNZ     short SETSFTC  ; Zero set if Ctrl-Z found in input
24632 00003D33 C43E[9E05]                  LES     DI,[THISSFT]
24633                                      ;and    byte [es:di+5],0BFh
24634 00003D37 26806505BF                  AND     BYTE [ES:DI+SF_ENTRY.sf_flags],~devid_device_EOF
24635                                                                      ; Mark as no more data available
24636                                  SETSFTC:
24637                                      ; 31/07/2019
24638                                      ;call   SETSFT
24639                                      ;retn
24640 00003D3C E95A05                      jmp     SETSFT
24641
24642                                  ; 16/12/2022
24643                                  %if 0
24644                                      ; 17/11/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
24645                                  ENDRDDEV:
24646                                  ;hkn; SS is DOSDATA
24647                                      push    ss
24648                                      pop     ds
24649                                      MOV     [NEXTADD],DI
24650                                      JNZ     short SETSFTC  ; Zero set if Ctrl-Z found in input
24651                                      LES     DI,[THISSFT]
24652                                      ;and    byte [es:di+5],0BFh
24653                                      AND     BYTE [ES:DI+SF_ENTRY.sf_flags],~devid_device_EOF
24654                                                                      ; Mark as no more data available
24655                                  SETSFTC:
24656                                      ;call   SETSFT
24657                                      ;retn
24658                                      jmp     SETSFT
24659                                  %endif
24660
24661                                  READCON:
```

```
00003D3F E821FE                  CALL    SWAPCON
00003D42 8B36[2200]              MOV     SI,[CONTPOS]
00003D46 09F6                    OR      SI,SI
00003D48 75C4                    JNZ     short TRANBUF
00003D4A 803E[7B02]80            CMP     BYTE [CONBUF],128 ; 80h
00003D4F 7406                    JZ      short GETBUF
00003D51 C706[7B02]80FF          MOV     WORD [CONBUF],0FF80H ; Set up 128-byte buffer with no template
                               GETBUF:
00003D57 51                      PUSH    CX
00003D58 06                      PUSH    ES
00003D59 57                      PUSH    DI

                               ;hkn; CONBUF is in DOSDATA
00003D5A BA[7B02]                MOV     DX,CONBUF

00003D5D E846DC                  call    _$STD_CON_STRING_INPUT; Get input buffer
00003D60 5F                      POP     DI
00003D61 07                      POP     ES
00003D62 59                      POP     CX

                               ;hkn; CONBUF is in DOSDATA
00003D63 BE[7D02]                MOV     SI,CONBUF+2

00003D66 803C1A                  CMP     BYTE [SI],1AH  ; Check for Ctrl-Z in first character
00003D69 75A3                    JNZ     short TRANBUF
00003D6B B01A                    MOV     AL,1AH
00003D6D AA                      STOSB
00003D6E 4F                      DEC     DI
00003D6F B00A                    MOV     AL,c_LF
00003D71 E8BDDE                  call    OUTT            ; Send linefeed
00003D74 31F6                    XOR     SI,SI
00003D76 EBAC                    JMP     short ENDRDCON ; 04/05/2019

                               ; 24/07/2018 - Retro DOS v3.0

                               ;Break      <DOS_WRITE -- MAIN WRITE ROUTINE AND DEVICE OUT ROUTINES>
                               ;-----------------------------------------------------------------------------
                               ;
                               ; Procedure Name : DOS_WRITE
                               ;
                               ; Inputs:
                               ;   ThisSFT set to the SFT for the file being used
                               ;   [DMAADD] contains transfer address
                               ;   CX = No. of bytes to write
                               ; Function:
                               ;   Perform write operation
                               ;   NOTE: If CX = 0 on input, file is truncated or grown
                               ;           to current sf_position
                               ; Outputs:
                               ;    Carry clear
                               ;   SFT Position and cluster pointers updated
                               ;   CX = No. of bytes written
                               ;   ES:DI point to SFT
                               ;    Carry set
                               ;   AX is error code
                               ;   CX = 0
                               ;   ES:DI point to SFT
                               ; DS preserved, all other registers destroyed
                               ;-----------------------------------------------------------------------------

                               ; 04/05/2019 - Retro DOS v4.0
                               ; DOSCODE:742Ch (MSDOS 6.21, MSDOS.SYS)

                               ; 17/11/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
                               ; DOSCODE:7418h (MSDOS 5.0, MSDOS.SYS)

                               ; 08/02/2024 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMDOS.COM)
                               ; PCDOS 7.1 IBMDOS.COM - DOSCODE:7D8Fh

                               DOS_WRITE:
00003D78 C43E[9E05]              LES     DI,[THISSFT]
                                 ;mov    al,[ES:DI+2]
00003D7C 268A4502                MOV     AL,[ES:DI+SF_ENTRY.sf_mode]
                                 ;;and   al,0Fh
                                 ;AND    AL,access_mask
                                 ; 07/02/2024 - Retro DOS v5.0
                                 ; (PCDOS 7.1 & Win Me)
00003D80 2403                    and     al,3    ; open_mode_mask ?
                                 ;cmp    al,0
00003D82 3C00                    CMP     AL,open_for_read
00003D84 7503                    JNE     short Check_FCB_RO              ;Is write or both
                               BadMode:
00003D86 E96304                  jmp     SET_ACC_ERR

                               ; NOTE: The following check for writting to a Read Only File is performed
                               ;       ONLY on FCBs!!!!
                               ;   We ALLOW writes to Read Only files via handles to allow a CREATE
                               ;       of a read only file which can then be written to.
                               ;   This is OK because we are NOT ALLOWED to OPEN a RO file via handles
                               ;       for writting, or RE-CREATE an EXISTING RO file via handles. Thus,
                               ;       CREATing a NEW RO file, or RE-CREATing an existing file which
                               ;       is NOT RO to be RO, via handles are the only times we can write
                               ;       to a read-only file.

                               Check_FCB_RO:
                                 ;;test  word [es:di+2],8000h
                                 ;TEST   word [ES:DI+SF_ENTRY.sf_mode],sf_isFCB
                                 ;JZ     short WRITE_NO_MODE    ; Not an FCB

                                 ;test   byte [es:di+3],80h
00003D89 26F6450380              TEST    byte [ES:DI+SF_ENTRY.sf_mode+1],(sf_isFCB>>8)
00003D8E 7407                    JZ      short WRITE_NO_MODE    ; Not an FCB

                                 ;test   byte [es:di+4],1
00003D90 26F6450401              TEST    byte [ES:DI+SF_ENTRY.sf_attr],attr_read_only
00003D95 75EF                    JNZ     short BadMode          ; Can't write to Read_Only files via FCB
                               WRITE_NO_MODE:
00003D97 E81503                  call    SETUP
00003D9A E8A5DA                  call    IsSFTNet
00003D9D 7406                    JZ      short LOCAL_WRITE

                               ;IF NOT Installed
                               ;   transfer NET_WRITE
                               ;ELSE
                                 ;mov    ax,1109h
00003D9F B80911                  MOV     AX,(MultNET<<8)|9
00003DA2 CD2F                    int     2Fh     ; Multiplex - NETWORK REDIRECTOR - WRITE TO REMOTE FILE
                                                 ; ES:DI -> SFT
                                                 ; SFT DPB field -> DPB of drive containing file
                                                 ; CX = number of bytes, SS = DOS CS, SDA DTA field -> user buffer
                                                 ; Return: CF set on error, CX = bytes written
00003DA4 C3                      retn
                               ;ENDIF
```

```
                                LOCAL_WRITE:
                                    ;;test word [es:di+5],80h
                                    ;TEST  word [ES:DI+SF_ENTRY.sf_flags],devid_device
                                    ;jnz   short WRTDEV
                                
                                    ;test  byte [es:di+5],80h
00003DA5 26F6450580             TEST   byte [ES:DI+SF_ENTRY.sf_flags],devid_device ; Check for named device I/O
00003DAA 756D                   jnz    short WRTDEV
                                
                                    ;mov   byte [EXTERR_LOCUS],2
00003DAC C606[2303]02           MOV    byte [EXTERR_LOCUS],errLOC_Disk
00003DB1 E82EDB                 call   ECritDisk
                                
00003DB4 E8A805                 call   DISKWRITE
                                
                                    ; 04/05/2019 - Retro DOS v4.0
                                
                                    ; MSDOS 6.0
                                ; Extended Open
00003DB7 7210                   JC     short nocommit
                                
00003DB9 C43E[9E05]             LES    DI,[THISSFT]
                                
                                    ;;test word [ES:DI+2],4000h
                                    ;TEST  word [ES:DI+SF_ENTRY.sf_mode],AUTO_COMMIT_WRITE
                                    ;JZ    short nocommit
                                
                                    ;test  byte [ES:DI+3],40h
00003DBD 26F6450340             TEST   byte [ES:DI+SF_ENTRY.sf_mode+1],(AUTO_COMMIT_WRITE>>8)
00003DC2 7405                   JZ     short nocommit
                                
00003DC4 51                     PUSH   CX
00003DC5 E80FFB                 call   DOS_COMMIT
00003DC8 59                     POP    CX
                                nocommit:
                                ; Extended Open
                                    ;call  LCritDisk
                                    ;retn
                                    ; 18/12/2022
00003DC9 E943DB                 jmp    LCritDisk
                                
                                DVWRTRAW:
00003DCC 31C0                   XOR    AX,AX                   ; Media Byte, unit = 0
00003DCE E87815                 call   SETWRITE
00003DD1 1E                     PUSH   DS                      ; Save seg of transfer
                                
                                ;hkn; SS override
00003DD2 36C536[9E05]           LDS    SI,[SS:THISSFT]
00003DD7 E8B814                 call   DEVIOCALL               ; DS:SI -> DEVICE
                                
00003DDA 89FA                   MOV    DX,DI                   ; Offset part of Xaddr saved in DX
00003DDC B487                   MOV    AH,87H
                                
                                ;hkn; SS override
00003DDE 368B3E[5D03]           MOV    DI,[SS:DEVCALL_REQSTAT]
                                
                                    ; MSDOS 3.3
                                    ;test  di,8000h
                                    ;jz    short CWRTROK
                                
                                    ; MSDOS 6.0
00003DE3 09FF                   or     di,di
00003DE5 791F                   jns    short CWRTROK
                                
                                    ; MSDOS 3.3 (& MSDOS 6.0)
00003DE7 E83922                 call   CHARHARD
                                
                                    ; 04/05/2019  - Retro DOS v4.0
                                
                                    ; 08/02/2024
00003DEA 368B3E[6C03]           mov    di,[ss:CALLSCNT]
                                    ; MSDOS 6.0
                                    ;sub   cx,[ss:CALLSCNT]      ; update ptr & count to reflect      M065
00003DEF 29F9                   sub    cx,di
00003DF1 89D3                   mov    bx,dx                   ; number of chars xferred     M065
                                    ;add   bx,[ss:CALLSCNT]      ;                             M065
00003DF3 01FB                   add    bx,di
00003DF5 89DF                   mov    di,bx                   ;                             M065
                                
                                    ; MSDOS 3.3
                                    ;MOV   BX,DX                 ; Recall transfer addr        M065
                                
                                    ; MSDOS 3.3 (& MSDOS 6.0)
00003DF7 08C0                   OR     AL,AL
00003DF9 740B                   JZ     short CWRTROK           ; Ignore
00003DFB 3C03                   CMP    AL,3
00003DFD 7403                   JZ     short CWRFERR
00003DFF 1F                     POP    DS                      ; Recover saved seg of transfer
00003E00 EBCA                   JMP    short DVWRTRAW          ; Try again
                                CWRFERR:
00003E02 58                     POP    AX                      ; Chuck saved seg of transfer
00003E03 E914FE                 JMP    CRDFERR                 ; Will pop one more stack element
                                CWRTROK:
00003E06 58                     POP    AX                      ; Chuck saved seg of transfer
00003E07 1F                     POP    DS
00003E08 A1[6C03]               MOV    AX,[CALLSCNT]           ; Get actual number of bytes transferred
                                ENDWRDEV:
00003E0B C43E[9E05]             LES    DI,[THISSFT]
00003E0F 89C1                   MOV    CX,AX
                                    ;call  ADDREC
                                    ;retn
                                    ; 16/12/2022
                                    ; 10/06/2019
00003E11 E9B404                 jmp    ADDREC
                                    ; 17/11/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
                                    ;call  ADDREC
                                    ;retn
                                
                                WRTNUL:
00003E14 89CA                   MOV    DX,CX                   ; Entire transfer done
                                WRTCOOKJ:
00003E16 E98C00                 JMP    WRTCOOKDONE
                                WRTDEV:
                                    ;mov   byte [EXTERR_LOCUS],4
00003E19 C606[2303]04           MOV    byte [EXTERR_LOCUS],errLOC_SerDev
                                    ;or    byte [es:di+5],40h
00003E1E 26804D0540             OR     BYTE [ES:DI+SF_ENTRY.sf_flags],devid_device_EOF
                                                               ; Reset EOF for input
                                    ;mov   bl,[es:di+5]
00003E23 268A5D05               MOV    BL,[ES:DI+SF_ENTRY.sf_flags]
00003E27 31C0                   XOR    AX,AX
00003E29 E3E0                   JCXZ   ENDWRDEV                ; problem of creating on a device.
00003E2B 1E                     PUSH   DS
00003E2C 88D8                   MOV    AL,BL
```

```
24910 00003E2E C51E[2C03]               LDS     BX,[DMAADD]             ; Xaddr to DS:BX
24911 00003E32 89DF                     MOV     DI,BX                   ; Xaddr to DS:DI
24912                                   ; 27/06/2024 (PCDOS 7.1 IBMDOS.COM)
24913                                   ; 08/02/2024
24914 00003E34 99                       cwd
24915                                   ;XOR    DX,DX                   ; Set starting point
24916                                   ;test   al,20h
24917 00003E35 A820                     test    AL,devid_device_raw     ; Raw?
24918                                   ;JZ     short TEST_DEV_CON
24919                                   ;JMP    DVWRTRAW
24920                                   ; 16/12/2022
24921 00003E37 7593                     jnz     short DVWRTRAW
24922                                   ; 17/11/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
24923                                   ;JZ     short TEST_DEV_CON
24924                                   ;JMP    short DVWRTRAW
24925
24926                                 TEST_DEV_CON:
24927                                   ;test   al,2
24928 00003E39 A802                     test    AL,devid_device_con_out ; Console output device?
24929 00003E3B 756E                     jnz     short WRITECON
24930                                   ;test   al,4
24931 00003E3D A804                     test    AL,devid_device_null
24932 00003E3F 75D3                     JNZ     short WRTNUL
24933 00003E41 89D0                     MOV     AX,DX
24934 00003E43 803F1A                   CMP     BYTE [BX],1Ah           ; ^Z?
24935 00003E46 74CE                     JZ      short WRTCOOKJ          ; Yes, transfer nothing
24936 00003E48 51                       PUSH    CX
24937 00003E49 B90100                   MOV     CX,1
24938 00003E4C E8FA14                   call    SETWRITE
24939 00003E4F 59                       POP     CX
24940
24941                                 ;hkn; SS override
24942 00003E50 36C536[9E05]             LDS     SI,[SS:THISSFT]
24943                                 ;
24944                                 ;SR; Removed X25 support from here
24945                                 ;
24946                                   ;lds    si,[si+7]
24947 00003E55 C57407                   LDS     SI,[SI+SF_ENTRY.sf_devptr]
24948                                 DVWRTLP:
24949 00003E58 E8A81F                   call    DSKSTATCHK
24950 00003E5B E83714                   call    DEVIOCALL2
24951 00003E5E 57                       PUSH    DI
24952 00003E5F B487                     MOV     AH,87H
24953
24954                                 ;hkn; SS override
24955 00003E61 368B3E[5D03]             MOV     DI,[SS:DEVCALL_REQSTAT]
24956
24957                                   ; MSDOS 3.3
24958                                   ;test   di,8000h
24959                                   ;jz     short CWROK
24960
24961                                   ; MSDOS 6.0
24962 00003E66 09FF                     or      di,di
24963 00003E68 7916                     jns     short CWROK
24964
24965                                   ; MSDOS 3.3 (& MSDOS 6.0)
24966 00003E6A E8B621                   call    CHARHARD
24967 00003E6D 5F                       POP     DI
24968
24969                                 ;hkn; SS override
24970 00003E6E 36C706[6C03]0100         MOV     word [SS:CALLSCNT],1
24971 00003E75 3C01                     CMP     AL,1
24972 00003E77 74DF                     JZ      short DVWRTLP  ; Retry
24973 00003E79 08C0                     OR      AL,AL
24974 00003E7B 740C                     JZ      short DVWRTIGN ; Ignore
24975                                   ; 10/08/2018
24976 00003E7D E99AFD                   JMP     CRDFERR        ; Fail, pops one stack element
24977                                 CWROK:
24978 00003E80 5F                       POP     DI
24979
24980                                 ;hkn; SS override
24981 00003E81 36833E[6C03]00           CMP     word [SS:CALLSCNT],0
24982 00003E87 741C                     JZ      short WRTCOOKDONE
24983                                 DVWRTIGN:
24984 00003E89 42                       INC     DX
24985
24986                                 ;hkn; SS override for CALLXAD
24987 00003E8A 36FF06[6803]             INC     WORD [SS:CALLXAD]
24988 00003E8F 47                       INC     DI
24989 00003E90 1E                       PUSH    DS
24990 00003E91 368E1E[6A03]             MOV     DS,[SS:CALLXAD+2]
24991 00003E96 803D1A                   CMP     BYTE [DI],1Ah  ; ^Z?
24992 00003E99 1F                       POP     DS
24993 00003E9A 7409                     JZ      short WRTCOOKDONE
24994
24995                                 ;hkn; SS override
24996 00003E9C 36C706[5D03]0000         MOV     word [SS:DEVCALL_REQSTAT],0
24997 00003EA3 E2B3                     LOOP    DVWRTLP
24998                                 WRTCOOKDONE:
24999 00003EA5 89D0                     MOV     AX,DX
25000 00003EA7 1F                       POP     DS
25001 00003EA8 E960FF                   JMP     ENDWRDEV ; 10/08/2018
25002
25003                                 WRITECON:
25004 00003EAB 1E                       PUSH    DS
25005
25006                                 ;hkn; SS is DOSDATA
25007 00003EAC 16                       push    ss
25008 00003EAD 1F                       pop     ds
25009 00003EAE E8B2FC                   CALL    SWAPCON
25010 00003EB1 1F                       POP     DS
25011 00003EB2 89DE                     MOV     SI,BX
25012 00003EB4 51                       PUSH    CX
25013                                 WRCONLP:
25014 00003EB5 AC                       LODSB
25015 00003EB6 3C1A                     CMP     AL,1Ah         ; ^Z?
25016 00003EB8 7405                     JZ      short CONEOF
25017 00003EBA E874DD                   call    OUTT
25018 00003EBD E2F6                     LOOP    WRCONLP
25019                                 CONEOF:
25020 00003EBF 58                       POP     AX                      ; Count
25021 00003EC0 1F                       POP     DS
25022 00003EC1 29C8                     SUB     AX,CX                   ; Amount actually written
25023 00003EC3 E897FC                   CALL    SWAPBACK
25024 00003EC6 E942FF                   JMP     ENDWRDEV
25025
25026                                 ;--------------------------------------------------------------------------
25027                                 ;
25028                                 ; Procedure Name : get_io_sft
25029                                 ;
25030                                 ;   Convert JFN number in BX to sf_entry in DS:SI We get the normal SFT if
25031                                 ;   CONSWAP is FALSE or if the handle desired is 2 or more. Otherwise, we
25032                                 ;   retrieve the sft from ConSFT which is set by SwapCon.
25033                                 ;
```

```
                                ;--------------------------------------------------------------------------
                                
                                ; 04/05/2019 - Retro DOS v4.0
                                ; DOSCODE:7583h (MSDOS 6.21, MSDOS.SYS)
                                ; 17/11/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
                                ; DOSCODE:756Fh (MSDOS 5.0, MSDOS.SYS)
                                
                                GET_IO_SFT:
                                    ;test  byte [SS:CONSWAP],0FFh
00003EC9 36803E[5703]00             cmp     byte [SS:CONSWAP],0                 ;smr;SS Override
00003ECF 7512                       JNZ     short GetRedir
                                GetNormal:
00003ED1 16                         push    ss
00003ED2 1F                         pop     ds
00003ED3 06                         PUSH    ES
00003ED4 57                         PUSH    DI
00003ED5 E8FB37                     call    SFFromHandle
00003ED8 7206                       JC      short RET44P
00003EDA 8CC6                       MOV     SI,ES
00003EDC 8EDE                       MOV     DS,SI
00003EDE 89FE                       MOV     SI,DI
                                RET44P:
00003EE0 5F                         POP     DI
00003EE1 07                         POP     ES
00003EE2 C3                         retn
                                GetRedir:
00003EE3 83FB01                     CMP     BX,1
00003EE6 77E9                       JA      short GetNormal
00003EE8 36C536[E605]               LDS     SI,[SS:CONSFT]
00003EED F8                         CLC
                                get_io_sft_retn:
00003EEE C3                         retn
                                
                                ;Break      <DIRREAD -- READ A DIRECTORY SECTOR>
                                ;--------------------------------------------------------------------------
                                ;
                                ; Procedure Name : DIRREAD
                                ;
                                ; Inputs:
                                ;   AX = Directory block number (relative to first block of directory)
                                ;   ES:BP = Base of drive parameters
                                ;   [DIRSEC] = First sector of first cluster of directory
                                ;   [CLUSNUM] = Next cluster
                                ;   [CLUSFAC] = Sectors/Cluster
                                ; Function:
                                ;   Read the directory block into [CURBUF].
                                ; Outputs:
                                ;   [NXTCLUSNUM] = Next cluster (after the one skipped to)
                                ;   [SECCLUSPOS] Set
                                ;   ES:BP unchanged
                                ;   [CURBUF] Points to Buffer with dir sector
                                ;   Carry set if error (user said FAIL to I 24)
                                ; DS preserved, all other registers destroyed.
                                ;--------------------------------------------------------------------------
                                
                                ;hkn; called from dir.asm. DS already set up to DOSDATA.
                                
                                ; 08/02/2024 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMDOS.COM)
                                
                                DIRREAD:
                                
                                ; Note that ClusFac is the sectors per cluster. This is NOT necessarily
                                ; the same as what is in the DPB! In the case of the root directory, we have
                                ; ClusFac = # sectors in the root directory. The root directory is detected
                                ; by DIRStart = 0.
                                
00003EEF 31D2                       XOR     DX,DX
                                    ; 08/02/2024 (PCDOS 7.1)
00003EF1 3916[DC0A]                 cmp     [DIRSTART_HW],dx
00003EF5 7509                       jnz     short SubDir
                                    ;CMP    word [DIRSTART],0
                                    ; 21/09/2023
00003EF7 3916[C205]                 cmp     [DIRSTART],dx ; 0
00003EFB 7503                       jnz     short SubDir
00003EFD 92                         XCHG    AX,DX
00003EFE EB0C                       JMP     short DoRead
                                
                                ; Convert the sector number in AX into cluster and sector-within-cluster pair
                                
                                SubDir:
00003F00 88C2                       MOV     DL,AL
                                    ;and    dl,[es:bp+4]
00003F02 26225604                   AND     DL,[ES:BP+DPB.CLUSTER_MASK]
                                
                                ;   (DX) = sector-in-cluster
                                
                                    ;mov    cl,[es:bp+5]
00003F06 268A4E05                   MOV     CL,[ES:BP+DPB.CLUSTER_SHIFT]
00003F0A D3E8                       SHR     AX,CL
                                
                                ;   (DX) = position in cluster
                                ;   (AX) = number of clusters to skip
                                
                                DoRead:
00003F0C 8816[7305]                 MOV     [SECCLUSPOS],DL
00003F10 89C1                       MOV     CX,AX
00003F12 88D4                       MOV     AH,DL
                                
                                ;   (CX) = number of clusters to skip.
                                ;   (AH) = remainder
                                
                                    ; 04/05/2019 - Retro DOS v4.0
                                
                                ; 08/02/2024
                                %if 0
                                    ; MSDOS 6.0
                                    ;MOV    DX,[DIRSEC+2]           ;>32mb
                                    ;MOV    [HIGH_SECTOR],DX        ;>32mb
                                    ;MOV    DX,[DIRSEC]
                                    ;ADD    DL,AH
                                    ;ADC    DH,0
                                    ;ADC    word [HIGH_SECTOR],0    ;>32mb
                                    ; 21/09/2023
                                    xor     bx,bx ; 0
                                    mov     dx,[DIRSEC]
                                    add     dl,ah
                                    adc     dh,bl ; 0
                                    adc     bx,[DIRSEC+2]
                                    mov     [HIGH_SECTOR],bx
                                
                                    MOV     BX,[CLUSNUM]
                                    MOV     [NXTCLUSNUM],BX
                                    JCXZ    FIRSTCLUSTER
                                %else
```

```
                                    ; 08/02/2024 (PCDOS 7.1)
00003F14 8B1E[DE0A]                 mov     bx,[CLUSNUM_HW]
00003F18 891E[E00A]                 mov     [NXTCLUSNUM_HW],bx
00003F1C 891E[E80A]                 mov     [CLUSTNUM_HW],bx
00003F20 8B16[C005]                 mov     dx,[DIRSEC+2]
00003F24 8916[0706]                 mov     [HIGH_SECTOR],dx
00003F28 8B16[BE05]                 mov     dx,[DIRSEC]
00003F2C 00E2                       add     dl,ah
00003F2E 80D600                     adc     dh,0
00003F31 8316[0706]00               adc     word [HIGH_SECTOR],0
00003F36 8B1E[BC05]                 mov     bx,[CLUSNUM]
00003F3A 891E[DC05]                 mov     [NXTCLUSNUM],bx
00003F3E E339                       jcxz    FIRSTCLUSTER
                                %endif

                                SKPCLLP:
00003F40 E89E23                     call    UNPACK
00003F43 72A9                       jc      short get_io_sft_retn
                                    ;;;
                                    ; 08/02/2024 (PCDOS 7.1)
00003F45 FF36[E80A]                 push    word [CLUSTNUM_HW]
00003F49 8F06[F20A]                 pop     word [CLUSTERS_HW]
00003F4D FF36[EC0A]                 push    word [CCONTENT_HW]
00003F51 8F06[E80A]                 pop     word [CLUSTNUM_HW]
                                    ;;;
00003F55 87DF                       XCHG    BX,DI
00003F57 E85E23                     call    IsEOF                   ; test for eof based on fat size
00003F5A 7302                       JAE     short HAVESKIPPED
00003F5C E2E2                       LOOP    SKPCLLP
                                HAVESKIPPED:
00003F5E 891E[DC05]                 MOV     [NXTCLUSNUM],BX
                                    ;;;
                                    ; 08/02/2024 (PCDOS 7.1)
00003F62 8B1E[E80A]                 mov     bx,[CLUSTNUM_HW]
00003F66 891E[E00A]                 mov     [NXTCLUSNUM_HW],bx
                                    ;
00003F6A 8B1E[F20A]                 mov     bx,[CLUSTERS_HW]
00003F6E 891E[E80A]                 mov     [CLUSTNUM_HW],bx
                                    ;;;
00003F72 89FA                       MOV     DX,DI
00003F74 88E3                       MOV     BL,AH
00003F76 E81E1A                     call    FIGREC

                                    ;entry  FIRSTCLUSTER

                                FIRSTCLUSTER:
                                    ; 22/09/2023
                                    ;;mov   byte [ALLOWED],18h
                                    ;MOV    byte [ALLOWED],Allowed_RETRY+Allowed_FAIL ; *
                                    ;XOR    AL,AL ; *       ; Indicate pre-read
                                    ;call   GETBUFFR
00003F79 E8AD29                     call    GETBUFFER ; *  ; pre-read
                                    ;jc     short get_io_sft_retn
                                    ; 08/02/2024
00003F7C 720C                       jc      short dirread_retn

                                    ;entry  SET_BUF_AS_DIR

                                SET_BUF_AS_DIR:

                                ;   Set the type of CURBUF to be a directory sector.
                                ;   Only flags are modified.

00003F7E 1E                         PUSH    DS
00003F7F 56                         PUSH    SI
00003F80 C536[E205]                 LDS     SI,[CURBUF]
                                    ;or     byte [si+5],4
00003F84 804C0504                   OR      byte [SI+BUFFINFO.buf_flags],buf_isDIR      ; Clears carry
00003F88 5E                         POP     SI
00003F89 1F                         POP     DS
                                dirread_retn:
00003F8A C3                         retn

                                ;Break      <FATSECRD -- READ A FAT SECTOR>
                                ;--------------------------------------------------------------------------
                                ;
                                ; Procedure Name : FATSECRD
                                ; Inputs:
                                ;   Same as DREAD
                                ;   DS:BX = Transfer address
                                ;   CX = Number of sectors
                                ;   DX = Absolute record number
                                ;   ES:BP = Base of drive parameters
                                ; Function:
                                ;   Calls BIOS to perform FAT read.
                                ; Outputs:
                                ;   Same as DREAD
                                ;--------------------------------------------------------------------------

                                    ; 04/05/2019 - Retro DOS v4.0
                                    ; 18/11/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)

                                    ; 09/02/2024 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMDOS.COM)

                                FATSECRD:
                                ;hkn; SS override
                                    ;mov    byte [ss:ALLOWED],18h
00003F8B 36C606[4B03]18             MOV     byte [SS:ALLOWED],Allowed_RETRY+Allowed_FAIL
00003F91 89CF                       MOV     DI,CX
                                    ;mov    cl,[es:bp+8]
00003F93 268A4E08                   MOV     CL,[ES:BP+DPB.FAT_COUNT]
                                    ; MSDOS 3.3
                                    ;;mov   al,[es:bp+0Fh]
                                    ;MOV    AL,[ES:BP+DPB.FAT_SIZE]
                                    ;XOR    AH,AH
                                    ; MSDOS 6.0
                                    ;mov    ax,[es:bp+0Fh]
00003F97 268B460F                   MOV     AX,[ES:BP+DPB.FAT_SIZE] ;>32mb
00003F9B 30ED                       XOR     CH,CH
                                    ;;;
                                    ; 09/02/2024 (PCDOS 7.1 IBMDOS.COM)
00003F9D 09C0                       or      ax,ax
00003F9F 7509                       jnz     short FATSECRD_cont   ; not FAT32
                                    ;test   byte [es:bp+23h],80h
00003FA1 26F6462380                 test    byte [es:bp+DPB.EXT_FLAGS],80h ; (FAT32 bs extd flags bit 7)
00003FA6 7402                       jz       short FATSECRD_cont
                                    ;mov    cx,1                        ; only one FAT is active
00003FA8 B101                       mov     cl,1
                                FATSECRD_cont:
00003FAA 36FF36[0706]               push    word [ss:HIGH_SECTOR]
                                    ;;;
00003FAF 52                         PUSH    DX
                                NXTFAT:
                                    ; MSDOS 6.0
```

```
                                ;hkn; SS override
                                    ; 09/02/2024 (PCDOS 7.1)
                                    ;MOV    word [SS:HIGH_SECTOR],0       ;>32mb FAT sectors cannot exceed
00003FB0 51                         PUSH    CX                      ;32mb
00003FB1 50                         PUSH    AX
                                    ;;;
                                    ; 08/02/2024 (PCDOS 7.1 IBMDOS.COM)
00003FB2 36FF36[0706]               push    word [ss:HIGH_SECTOR]
00003FB7 52                         push    dx
                                    ;;;
00003FB8 89F9                       MOV     CX,DI
00003FBA E88600                     call    DSKREAD
                                    ;;;
                                    ; 09/02/2024 (PCDOS 7.1 IBMDOS.COM)
00003FBD 5A                         pop     dx
00003FBE 368F06[0706]               pop     word [ss:HIGH_SECTOR]
                                    ;;;
00003FC3 58                         POP     AX
00003FC4 59                         POP     CX
00003FC5 7440                       JZ      short RET41P          ; Carry clear
                                    ;;;
                                    ; 09/02/2024 (PCDOS 7.1 IBMDOS.COM)
00003FC7 09C0                       or      ax,ax
00003FC9 7511                       jnz     short NXTFAT2  ; not FAT32
                                    ;add    dx,[es:bp+31h]
00003FCB 26035631                   add     dx,[es:bp+DPB.FAT32_SIZE]
00003FCF 50                         push    ax
                                    ;mov    ax,[es:bp+33h]
00003FD0 268B4633                   mov     ax,[es:bp+DPB.FAT32_SIZE+2]
00003FD4 361106[0706]               adc     [ss:HIGH_SECTOR],ax
00003FD9 58                         pop     ax
00003FDA EB08                       jmp     short NXTFAT3
                                NXTFAT2:
                                    ;;;
00003FDC 01C2                       ADD     DX,AX
                                    ;;;
                                    ; 09/02/2024 (PCDOS 7.1)
00003FDE 368316[0706]00             adc     word [ss:HIGH_SECTOR],0
                                NXTFAT3:
                                    ;;;
00003FE4 E2CA                       LOOP    NXTFAT
00003FE6 5A                         POP     DX
                                    ;;;
                                    ; 09/02/2024 (PCDOS 7.1)
00003FE7 368F06[0706]               pop     word [ss:HIGH_SECTOR]
                                    ;;;
00003FEC 89F9                       MOV     CX,DI

                                ; NOTE FALL THROUGH

                                ;Break      <DREAD -- DO A DISK READ>
                                ;--------------------------------------------------------------------------
                                ;
                                ; Procedure Name : DREAD
                                ;
                                ; Inputs:
                                ;   DS:BX = Transfer address
                                ;   CX = Number of sectors
                                ;   DX = Absolute record number        (LOW)
                                ;   [HIGH_SECTOR] = Absolute record number (HIGH)
                                ;   ES:BP = Base of drive parameters
                                ;   [ALLOWED] must be set in case call to HARDERR needed
                                ; Function:
                                ;   Calls BIOS to perform disk read. If BIOS reports
                                ;   errors, will call HARDERRRW for further action.
                                ; Outputs:
                                ;   Carry set if error (currently user FAILED to INT 24)
                                ; DS,ES:BP preserved. All other registers destroyed.
                                ;--------------------------------------------------------------------------

                                    ;entry  DREAD
                                DREAD:
00003FEE E85200                     call    DSKREAD
00003FF1 7497                       jz      short dirread_retn    ; Carry clear
                                ;hkn; SS override
00003FF3 36C606[7505]00             MOV     BYTE [SS:READOP],0    ; Read
00003FF9 E89A00                     call    HARDERRRW
00003FFC 3C01                       CMP     AL,1                  ; Check for retry
00003FFE 74EE                       JZ      short DREAD

                                fail_ignore:       ; 09/02/2024
00004000 3C03                       CMP     AL,3                  ; Check for FAIL
00004002 F8                         CLC
00004003 7501                       JNZ     short NO_CAR          ; Ignore
00004005 F9                         STC
                                NO_CAR:
00004006 C3                         retn

                                    ; 09/02/2024 - Retro DOS v5.0
                                RET41P:
00004007 5A                         POP     DX
                                    ;;;
                                    ; 09/02/2024 (PCDOS 7.1 IBMDOS.COM)
00004008 368F06[0706]               pop     word [ss:HIGH_SECTOR]
                                    ;;;
0000400D C3                         retn

                                ; 24/07/2018 - Retro DOS v3.0

                                ;Break      <CHECK_WRITE_LOCK>
                                ;--------------------------------------------------------------------------
                                ;
                                ; Procedure Name : CHECK_WRITE_LOCK
                                ;
                                ; Inputs:
                                ;   output of SETUP
                                ;   ES:DI -> SFT
                                ; Function:
                                ;   check write lock
                                ; Outputs:
                                ;   Carry set if error
                                ;   Carry clear if ok
                                ;
                                ;--------------------------------------------------------------------------

                                    ; 04/05/2019 - Retro DOS v4.0
                                    ; 18/11/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)

                                CHECK_WRITE_LOCK:
                                    ; MSDOS 6.0
                                    ;test   byte [es:di+4],8
0000400E 26F6450408                 TEST    byte [ES:DI+SF_ENTRY.sf_attr],attr_volume_id ;volume id
                                    ;JZ     short write_cont                        ;no
                                    ;;call  SET_ACC_ERR_DS
```

```
                                    ;;retn
                                    ;;jnz   SET_ACC_ERR_DS
                                    ; 19/08/2018
                                    ;jz     short write_cont
                                    ;jmp    SET_ACC_ERR_DS
                                    ; 18/11/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
00004013 7403                       JZ      short write_cont
                                    ;call   SET_ACC_ERR_DS
                                    ;retn
                                    ; 16/12/2022
00004015 E9D201                     jmp     SET_ACC_ERR_DS

                                write_cont:                     ;
00004018 51                         PUSH    CX                  ;save reg
00004019 09C9                       OR      CX,CX               ;
0000401B 7501                       JNZ     short Not_Truncate  ;
0000401D 49                         dec     cx                  ;(cx) = -1; check for lock on whole file
                                Not_Truncate:                       ;
0000401E B080                       MOV     AL,80H              ;check write access
00004020 E8C143                     call    LOCK_CHECK          ;check lock
00004023 59                         POP     CX                  ;restore reg
00004024 7305                       JNC     short WRITE_OK      ;lock ok
00004026 E87301                     call    WRITE_LOCK_VIOLATION ;issue I24
00004029 73ED                       JNC     short write_cont    ;retry
                                WRITE_OK:                       ;
0000402B C3                         retn                        ;

                                ;Break      <CHECK_READ_LOCK>
                                ;-----------------------------------------------------------------------------
                                ;
                                ; Procedure Name : CHECK_READ_LOC
                                ;
                                ; Inputs:
                                ;   ES:DI -> SFT
                                ;   output of SETUP
                                ; Function:
                                ;   check read lock
                                ; Outputs:
                                ;   Carry set if error
                                ;   Carry clear if ok
                                ;-----------------------------------------------------------------------------

                                CHECK_READ_LOCK:
                                    ; MSDOS 6.0
                                    ;test   byte [es:di+4],8
0000402C 26F6450408                 TEST    byte [ES:DI+SF_ENTRY.sf_attr],attr_volume_id ;volume id
                                    ;JZ     short do_retry                      ; no
                                    ;;call  SET_ACC_ERR
                                    ;;retn
                                    ;;jnz   SET_ACC_ERR
                                    ; 16/12/2022
                                    ; 28/07/2019
00004031 7403                       jz      short do_retry
00004033 E9B601                     jmp     SET_ACC_ERR
                                    ; 18/11/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
                                    ;JZ     short do_retry
                                    ;call   SET_ACC_ERR
                                    ;retn
                                do_retry:                       ;
00004036 30C0                       xor     al,al               ;check read access
00004038 E8A943                     call    LOCK_CHECK          ;check lock
0000403B 7305                       JNC     short READLOCK_OK   ;lock ok
0000403D E83C01                     call    READ_LOCK_VIOLATION ;issue I24
00004040 73F4                       JNC     short do_retry      ;retry
                                READLOCK_OK:                        ;
                                dw_ret_label:       ; 09/02/2024
00004042 C3                         retn                        ;

                                ;============================================================================
                                ; DISK2.ASM, MSDOS 6.0, 1991
                                ;============================================================================
                                ; 24/07/2018 - Retro DOS v3.0
                                ; 04/05/2019 - Retro DOS v4.0

                                ;   TITLE   DISK2 - Disk utility routines
                                ;   NAME    Disk2

                                ;** Low level Read and write routines for local SFT I/O on files and devs
                                ;
                                ;   DskRead
                                ;   DWRITE
                                ;   DSKWRITE
                                ;   HarderrRW
                                ;   SETUP
                                ;   BREAKDOWN
                                ;   READ_LOCK_VIOLATION
                                ;   WRITE_LOCK_VIOLATION
                                ;   DISKREAD
                                ;   SET_ACC_ERR_DS
                                ;   SET_ACC_ERR
                                ;   SETSFT
                                ;   SETCLUS
                                ;   AddRec
                                ;
                                ;   Revision history:
                                ;
                                ;          AN000 version 4.00 Jan. 1988
                                ;          M039 DB 10/17/90 - Disk read/write optimization

                                ; 04/05/2019 - Retro DOS v4.0
                                ; DOSCODE:7699h (MSDOS 6.21, MSDOS.SYS)
                                ; 18/11/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
                                ; DOSCODE:7685h (MSDOS 5.0, MSDOS.SYS)

                                ;Break      <DSKREAD -- PHYSICAL DISK READ>
                                ;-----------------------------------------------------------------------------
                                ;
                                ; Procedure Name : DSKREAD
                                ;
                                ; Inputs:
                                ;   DS:BX = Transfer addr
                                ;   CX = Number of sectors
                                ;   [HIGH_SECTOR] = Absolute record number (HIGH)
                                ;   DX = Absolute record number          (LOW)
                                ;   ES:BP = Base of drive parameters
                                ; Function:
                                ;   Call BIOS to perform disk read
                                ; Outputs:
                                ;   DI = CX on entry
                                ;   CX = Number of sectors unsuccessfully transfered
                                ;   AX = Status word as returned by BIOS (error code in AL if error)
                                ;   Zero set if OK (from BIOS) (carry clear)
                                ;   Zero clear if error (carry clear)
                                ; SI Destroyed, others preserved
```

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

                                DSKREAD:
00004043 51                         PUSH    CX
                                    ;mov    ah,[es:bp+17h] ; 04/05/2019
00004044 268A6617                   MOV     AH,[ES:BP+DPB.MEDIA]
                                    ;mov    al,[es:bp+1]
00004048 268A4601                   MOV     AL,[ES:BP+DPB.UNIT]
0000404C 53                         PUSH    BX
0000404D 06                         PUSH    ES
0000404E E8C512                     call    SETREAD
00004051 EB22                       JMP     short DODSKOP

                                ;Break      <DWRITE -- SEE ABOUT WRITING>
                                ;--------------------------------------------------------------------------
                                ;
                                ; Procedure Name : DWRITE
                                ;
                                ; Inputs:
                                ;   DS:BX = Transfer address
                                ;   CX = Number of sectors
                                ;   [HIGH_SECTOR] = Absolute record number (HIGH)
                                ;   DX = Absolute record number          (LOW)
                                ;   ES:BP = Base of drive parameters
                                ;   [ALLOWED] must be set in case HARDERR called
                                ; Function:
                                ;   Calls BIOS to perform disk write. If BIOS reports
                                ;   errors, will call HARDERRRW for further action.
                                ; Output:
                                ;   Carry set if error (currently, user FAILed to I 24)
                                ; BP preserved. All other registers destroyed.
                                ;--------------------------------------------------------------------------

                                    ;entry  DWRITE
                                DWRITE:
00004053 E81100                     CALL    DSKWRITE
00004056 74EA                       jz      short dw_ret_label    ; Carry clear (retz)

                                ;hkn; SS override
00004058 36C606[7505]01             MOV     BYTE [SS:READOP],1    ; Write
0000405E E83500                     call    HARDERRRW
00004061 3C01                       CMP     AL,1                  ; Check for retry
00004063 74EE                       JZ      short DWRITE

                                ; 09/02/2024
                                %if 0
                                    CMP     AL,3                  ; Check for FAIL
                                    CLC
                                    JNZ     short NO_CAR2         ; Ignore
                                    STC
                                NO_CAR2:
                                dw_ret_label:
                                    retn
                                %else
                                    ; 09/02/2024 - Retro DOS v5.0
00004065 EB99                       jmp     short fail_ignore
                                %endif

                                ;Break      <DSKWRITE -- PHYSICAL DISK WRITE>
                                ;--------------------------------------------------------------------------
                                ;
                                ; Procedure Name : DSKWRITE
                                ;
                                ; Inputs:
                                ;   DS:BX = Transfer addr
                                ;   CX = Number of sectors
                                ;   DX = Absolute record number          (LOW)
                                ;   [HIGH_SECTOR] = Absolute record number (HIGH)
                                ;   ES:BP = Base of drive parameters
                                ; Function:
                                ;   Call BIOS to perform disk read
                                ; Outputs:
                                ;   DI = CX on entry
                                ;   CX = Number of sectors unsuccessfully transfered
                                ;   AX = Status word as returned by BIOS (error code in AL if error)
                                ;   Zero set if OK (from BIOS) (carry clear)
                                ;   Zero clear if error (carry clear)
                                ; SI Destroyed, others preserved
                                ;
                                ;--------------------------------------------------------------------------

                                    ;entry  DSKWRITE
                                DSKWRITE:
00004067 51                         PUSH    CX
                                    ;mov    ah,[es:bp+17h] ; 04/05/2019
00004068 268A6617                   MOV     AH,[ES:BP+DPB.MEDIA]
                                    ;mov    al,[es:bp+1]
0000406C 268A4601                   MOV     AL,[ES:BP+DPB.UNIT]
00004070 53                         PUSH    BX
00004071 06                         PUSH    ES
00004072 E8D412                     call    SETWRITE
                                DODSKOP:
00004075 8CD9                       MOV     CX,DS                 ; Save DS
00004077 1F                         POP     DS                    ; DS:BP points to DPB
00004078 1E                         PUSH    DS

                                    ;lds    si,[ds:bp+13h] ; 04/05/2019
00004079 3EC57613                   LDS     SI,[ds:BP+DPB.DRIVER_ADDR] ; 07/09/2018
0000407D E81512                     call    DEVIOCALL2

00004080 8ED9                       MOV     DS,CX                 ; Restore DS
00004082 07                         POP     ES                    ; Restore ES
00004083 5B                         POP     BX

                                ;hkn; SS override
00004084 368B0E[6C03]               MOV     CX,[SS:CALLSCNT] ; Number of sectors transferred
00004089 5F                         POP     DI
0000408A 29F9                       SUB     CX,DI
0000408C F7D9                       NEG     CX                    ; Number of sectors not transferred

                                ;hkn; SS override
0000408E 36A1[5D03]                 MOV     AX,[SS:DEVCALL_REQSTAT]
                                    ;test   ax,8000h
                                    ; 17/12/2022
                                    ;test   ah,80h
00004092 F6C480                     test    ah,(STERR>>8)
                                    ;test   AX,STERR
00004095 C3                         retn

                                ;Break      <HardErrRW - map extended errors and call harderr>
                                ;--------------------------------------------------------------------------
                                ;
                                ; Procedure Name : HardErrRW
                                ;
```

```
                                ; Inputs:
                                ;   AX is error code from read or write
                                ;   Other registers set as per HARDERR
                                ; Function:
                                ;   Checks the error code for special extended
                                ;   errors and maps them if needed. Then invokes
                                ;   Harderr
                                ; Outputs:
                                ;   Of HARDERR
                                ; AX may be modified prior to call to HARDERR.
                                ; No other registers altered.
                                ;
                                ;--------------------------------------------------------------------------

                                    ; 18/11/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
                                HARDERRRW:
                                    ;cmp    al,0Fh
00004096 3C0F                       CMP     AL,error_I24_wrong_disk
00004098 7512                       JNZ     short DO_ERR                ; Nothing to do

                                    ; MSDOS 3.3
                                    ;push   ds
                                    ;push   si
                                    ;lds    si,[ss:CALLVIDRW]
                                    ;mov    [ss:EXTERRPT+2], ds
                                    ;mov    [ss:EXTERRPT], si
                                    ;pop    si
                                    ;pop    ds

                                    ; MSDOS 6.0
0000409A 50                         push    ax
0000409B 36A1[7003]                 mov     ax,[SS:CALLVIDRW]           ; get ptr lo  ;smr;SS Override
0000409F 36A3[2803]                 mov     [ss:EXTERRPT],ax            ; set ext err ptr lo
000040A3 36A1[7203]                 mov     ax,[SS:CALLVIDRW+2]         ; get ptr hi from dev
000040A7 36A3[2A03]                 mov     [ss:EXTERRPT+2],ax          ; set ext err ptr hi
000040AB 58                         pop     ax
                                DO_ERR:
                                    ;;call  HARDERR
                                    ;;retn
                                    ; 16/12/2022
                                    ; 10/06/2019
000040AC E9A51F                     jmp     HARDERR
                                    ; 18/11/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
                                    ;call   HARDERR
                                    ;retn

                                ; 24/07/2018 - Retro DOS v3.0

                                ;Break      <SETUP -- SETUP A DISK READ OR WRITE FROM USER>
                                ;--------------------------------------------------------------------------
                                ;
                                ; Procedure Name : SETUP
                                ;
                                ; Inputs:
                                ;   ES:DI point to SFT (value also in THISSFT)
                                ;   DMAAdd contains transfer address
                                ;   CX = Byte count
                                ;   DS = DOSDATA
                                ;   WARNING Stack must be clean, two ret addrs on stack, 1st of caller,
                                ;           2nd of caller of caller.
                                ; Outputs:
                                ;       CX = byte count
                                ;       [THISDPB] = Base of drive parameters if file
                                ;                   = Pointer to device header if device or NET
                                ;       ES:DI Points to SFT
                                ;       [NEXTADD] = Displacement of disk transfer within segment
                                ;       [TRANS] = 0 (No transfers yet)
                                ;       BytPos = Byte position in file
                                ;
                                ;   The following fields are relevant to local files (not devices) only:
                                ;
                                ;       SecPos = Position of first sector (local files only)
                                ;       [BYTSECPOS] = Byte position in first sector (local files only)
                                ;       [CLUSNUM] = First cluster (local files only)
                                ;       [SECCLUSPOS] = Sector within first cluster (local files only)
                                ;       [THISDRV] = Physical unit number (local files only)
                                ;
                                ;      RETURNS ONE LEVEL UP WITH:
                                ;      CX = 0
                                ;      CARRY = Clear
                                ;   IF AN ERROR IS DETECTED
                                ; All other registers destroyed
                                ;--------------------------------------------------------------------------

                                ;hkn; called from disk.asm. DS has been set up to DOSDATA.

                                ; DOSCODE:770Bh (MSDOS 6.21, MSDOS.SYS)

                                ; 18/11/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
                                ; DOSCODE:76F7h (MSDOS 5.0, MSDOS.SYS)

                                ; 09/02/2024 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMDOS.COM)
                                ; DOSCODE:80D8h (PCDOS 7.1, IBMDOS.COM)

                                SETUP:
                                    ; IBMDOS.COM (MSDOS 3.3) - Offset 411Bh

                                    ;lds    si,[es:di+7]
000040AF 26C57507                   LDS     SI,[ES:DI+SF_ENTRY.sf_devptr]

                                ;hkn; SS override
000040B3 368C1E[8C05]               MOV     [SS:THISDPB+2],DS

                                ;hkn; SS is DOSDATA
000040B8 16                         push    ss
000040B9 1F                         pop     ds

000040BA 8936[8A05]                 MOV     [THISDPB],SI

000040BE 8B1E[2C03]                 MOV     BX,[DMAADD]
000040C2 891E[B805]                 MOV     [NEXTADD],BX            ;Set NEXTADD to start of Xaddr
000040C6 C606[7405]00               MOV     BYTE [TRANS],0          ;No transferes
                                    ;mov    ax,[es:di+15h]
000040CB 268B4515                   MOV     AX,[ES:DI+SF_ENTRY.sf_position]
                                    ;mov    dx,[es:di+17h]
000040CF 268B5517                   MOV     DX,[ES:DI+SF_ENTRY.sf_position+2]
000040D3 8916[D005]                 MOV     [BYTPOS+2],DX           ;Set it
000040D7 A3[CE05]                   MOV     [BYTPOS],AX
                                    ;test   word [es:di+5],8080h
000040DA 26F745058080               TEST    word [ES:DI+SF_ENTRY.sf_flags],sf_isnet+devid_device
000040E0 7555                       JNZ     short NOSETSTUFF        ;Following not done on devs or NET
000040E2 06                         PUSH    ES
000040E3 C42E[8A05]                 LES     BP,[THISDPB]            ;Point at the DPB
```

```
                                    ; 18/11/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
                                    ;;mov   bl,[es:bp+0]
                                    ;MOV    BL,[ES:BP+DPB.DRIVE]
                                    ; 05/12/2022
000040E7 268A5E00                   mov     bl,[es:bp]

000040EB 881E[7605]                 MOV     [THISDRV],BL            ;Set THISDRV
                                    ;mov    bx,[es:bp+2]
000040EF 268B5E02                   MOV     BX,[ES:BP+DPB.SECTOR_SIZE]

                                    ;; MSDOS 3.3
                                    ;cmp    dx,bx
                                    ;jnb    short EOFERR
                                    ;div    bx
                                    ;mov    [SECPOS],ax
                                    ;mov    [BYTSECPOS],dx
                                    ;mov    dx,ax
                                    ;;and   al,[es:bp+4]
                                    ;AND    AL,[ES:BP+DPB.CLUSTER_MASK]
                                    ;mov    [SECCLUSPOS],al
                                    ;mov    ax,cx
                                    ;;mov   cl,[es:bp+5]
                                    ;MOV    CL,[ES:BP+DPB.CLUSTER_SHIFT]
                                    ;shr    dx,cl
                                    ;mov    [CLUSNUM],dx
                                    ;pop    es
                                    ;mov    cx,ax

                                    ; 04/05/2019 - Retro DOS v4.0

                                    ; MSDOS 6.0
                                ;M039: Optimized this section.
000040F3 51                             PUSH    CX                  ;SHR32 and DIV32 use CX.
000040F4 E81406                     call    DIV32                   ;DX:AX/BX = CX:AX + DX (rem)
000040F7 8916[CC05]                 MOV     [BYTSECPOS],DX
000040FB A3[C405]                   MOV     [SECPOS],AX
000040FE 890E[C605]                 MOV     [SECPOS+2],CX
00004102 89CA                       MOV     DX,CX

00004104 89C3                       MOV     BX,AX
                                    ;and    bl,[es:bp+4]
00004106 26225E04                   AND     BL,[ES:BP+DPB.CLUSTER_MASK]
0000410A 881E[7305]                 MOV     [SECCLUSPOS],BL

0000410E E82106                     call    SHR32                   ;(DX:AX SHR dpb_cluster_shift)
00004111 59                         POP     CX                      ;CX = byte count.

                                    ; 09/02/2024 (PCDOS7.1 IBMDOS.COM)
                                    ;JNZ    short EOFERR            ;cluster number above 64k
                                    ;;;
                                    ;cmp    word [es:bp+0Fh],0
00004112 26837E0F00                 cmp     word [es:bp+DPB.FAT_SIZE],0
00004117 750C                       jnz     short setup_1           ; not FAT32 fs
                                    ;cmp    dx,[es:bp+2Fh]
00004119 263B562F                   cmp     dx,[es:bp+DPB.LAST_CLUSTER+2]
0000411D 750E                       jne     short setup_2
                                    ;cmp    ax,[es:bp+2Dh]
0000411F 263B462D                   cmp     ax,[es:bp+DPB.LAST_CLUSTER]
00004123 EB08                       jmp     short setup_2
                                setup_1:    ; FAT12 or FAT16 fs
00004125 09D2                       or      dx,dx
00004127 7523                       jnz     short EOFERR
                                    ;cmp    ax,[es:bp+0Dh]
00004129 263B460D                   cmp     ax,[es:bp+DPB.MAX_CLUSTER]
                                setup_2:
0000412D 771D                       ja      short EOFERR
0000412F 8916[DE0A]                 mov     [CLUSNUM_HW],dx
                                    ;;;

                                    ;;cmp   ax,[es:bp+0Dh]
                                    ;CMP    AX,[ES:BP+DPB.MAX_CLUSTER] ;>32mb  if > disk size ;AN000;
                                    ;JA     short EOFERR               ;>32mb  then EOF       ;AN000;

00004133 A3[BC05]                   MOV     [CLUSNUM],AX
00004136 07                         POP     ES                      ; ES:DI point to SFT
                                ;M039

                                NOSETSTUFF:
00004137 89C8                       MOV     AX,CX           ; AX = Byte count.
00004139 0306[2C03]                 ADD     AX,[DMAADD]     ; See if it will fit in one segment
0000413D 730C                       JNC     short setup_OK ; Must be less than 64K
0000413F A1[2C03]                   MOV     AX,[DMAADD]
00004142 F7D8                       NEG     AX              ; Amount of room left in segment (know
                                                            ;    less than 64K since max value of CX
                                                            ;    is FFFF).
00004144 7501                       JNZ     short NoDec
00004146 48                         DEC     AX

                                NoDec:
00004147 89C1                       MOV     CX,AX           ; Can do this much
00004149 E304                       JCXZ    NOROOM          ; Silly user gave Xaddr of FFFF in segment
                                setup_OK:
0000414B C3                         retn

                                EOFERR:
0000414C 07                         POP     ES              ; ES:DI point to SFT
0000414D 31C9                       XOR     CX,CX           ; No bytes read
                                ;;;;;;;;;;; 7/18/86
                                    ; MSDOS 3.3
                                    ;MOV    BYTE [DISK_FULL],1 ; set disk full flag
                                ;;;;;;;;;;;
                                NOROOM:
0000414F 5B                         POP     BX              ; Kill return address
00004150 F8                         CLC
00004151 C3                         retn                    ; RETURN TO CALLER OF CALLER

                                ;Break      <BREAKDOWN -- CUT A USER READ OR WRITE INTO PIECES>
                                ;-----------------------------------------------------------------------------
                                ;
                                ; Procedure Name : BREAKDOWN
                                ;
                                ; Inputs:
                                ;   CX = Length of disk transfer in bytes
                                ;   ES:BP = Base of drive parameters
                                ;   [BYTSECPOS] = Byte position within first sector
                                ;   DS = DOSDATA
                                ; Outputs:
                                ;   [BYTCNT1] = Bytes to transfer in first sector
                                ;   [SECCNT] = No. of whole sectors to transfer
                                ;   [BYTCNT2] = Bytes to transfer in last sector
                                ; AX, BX, DX destroyed. No other registers affected.
                                ;-----------------------------------------------------------------------------

                                BREAKDOWN:
00004152 A1[CC05]                   MOV     AX,[BYTSECPOS]
```

```
                            MOV     BX,CX
                            OR      AX,AX
                            JZ      short SAVFIR   ; Partial first sector?
                            ;sub    ax,[es:bp+2]
                            SUB     AX,[ES:BP+DPB.SECTOR_SIZE]
                            NEG     AX             ; Max number of bytes left in first sector
                            SUB     BX,AX          ; Subtract from total length
                            JAE     short SAVFIR
                            ADD     AX,BX          ; Don't use all of the rest of the sector
                            XOR     BX,BX          ; And no bytes are left
                        SAVFIR:
                            MOV     [BYTCNT1],AX
                            MOV     AX,BX
                            XOR     DX,DX
                            ;div    word [ES:BP+2]
                            DIV     word [ES:BP+DPB.SECTOR_SIZE]  ; How many whole sectors?
                            MOV     [SECCNT],AX
                            MOV     [BYTCNT2],DX   ; Bytes remaining for last sector
                            ; MSDOS 3.3
                            ;OR     DX,[BYTCNT1]   ; SMR ONESECTORFIX BUGBUG
                            ;retnz                 ; NOT (BYTCNT1 = BYTCNT2 = 0)
                            ;CMP    AX,1
                            ;retnz
                            ;MOV    AX,[ES:BP+DPB.SECTOR_SIZE] ; Buffer EXACT one sector I/O
                            ;MOV    [BYTCNT2],AX
                            ;MOV    [SECCNT],DX    ; DX = 0
                        _RET45:
                            retn

                        ; DOSCODE:77BFh (MSDOS 6.21, MSDOS.SYS)

                        ;--------------------------------------------------------------------------
                        ;
                        ; Procedure Name : READ_LOCK_VIOLATION
                        ;
                        ; ES:DI points to SFT. This entry used by NET_READ
                        ; Carry set if to return error (CX=0,AX=error_sharing_violation).
                        ; Else do retrys.
                        ; ES:DI,DS,CX preserved
                        ;
                        ;--------------------------------------------------------------------------

                        READ_LOCK_VIOLATION:
                            MOV     byte [READOP],0
                        ERR_ON_CHECK:
                            ;;test  word [es:di+2],8000h
                            ;TEST   word [ES:DI+SF_ENTRY.sf_mode],sf_isFCB
                            ;JNZ    short HARD_ERR

                            ; 04/05/2019
                            ;test   byte [es:di+3],80h
                            TEST    byte [ES:DI+SF_ENTRY.sf_mode+1],(sf_isFCB>>8)
                            JNZ     short HARD_ERR

                            ;PUSH   CX
                            ;;mov   cl,[es:di+2]
                            ;MOV    CL,[ES:DI+SF_ENTRY.sf_mode]
                            ;;and   cl,0F0h
                            ;AND    CL,SHARING_MASK
                            ;;cmp   cl,0
                            ;CMP    CL,SHARING_COMPAT
                            ;POP    CX
                            ;JNE    short NO_HARD_ERR
                            ; 21/09/2023
                            mov     al,[ES:DI+SF_ENTRY.sf_mode]
                            and     al,SHARING_MASK
                            ;cmp    al,SHARING_COMPAT
                            ;jne    short NO_HARD_ERR
                            jnz     short NO_HARD_ERR
                        HARD_ERR:
                            call    LOCK_VIOLATION
                            jnc     short _RET45          ; User wants Retrys
                        NO_HARD_ERR:
                            XOR     CX,CX                 ;No bytes transferred
                            ;mov    ax,21h
                            MOV     AX,error_lock_violation
                            STC
                        RET3:               ; 06/02/2024
                            retn

                        ;--------------------------------------------------------------------------
                        ;
                        ; Procedure Name : WRITE_LOCK_VIOLATION
                        ;
                        ; Same as READ_LOCK_VIOLATION except for READOP.
                        ; This entry used by NET_WRITE
                        ;
                        ;--------------------------------------------------------------------------

                        WRITE_LOCK_VIOLATION:
                            MOV     byte [READOP],1
                            JMP     short ERR_ON_CHECK

                        ; 04/05/2019 - Retro DOS v4.0

                        ; DOSCODE:77ECh (MSDOS 6.21, MSDOS.SYS)

                        ;Break      <DISKREAD -- PERFORM USER DISK READ>
                        ;--------------------------------------------------------------------------
                        ;
                        ; Procedure Name : DISKREAD
                        ;
                        ; Inputs:
                        ;   Outputs of SETUP
                        ; Function:
                        ;   Perform disk read
                        ; Outputs:
                        ;    Carry clear
                        ;   CX = No. of bytes read
                        ;   ES:DI point to SFT
                        ;   SFT offset and cluster pointers updated
                        ;    Carry set
                        ;   CX = 0
                        ;   ES:DI point to SFT
                        ;   AX has error code
                        ;--------------------------------------------------------------------------

                        ;hkn; called from disk.asm. DS already set up.

                        ; 18/11/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
                        ; DOSCODE:77D8h (MSDOS 5.0, MSDOS.SYS)

                        ; 09/02/2024
                        ; 06/02/2024 - Retro DOS v5.0
```

```
                                ; PCDOS 7.1 IBMDOS.COM - DOSCODE:81D5h

                                DISKREAD:
                                    ;mov    ax,[es:di+11h]
000041A3 268B4511                   MOV     AX,[ES:DI+SF_ENTRY.sf_size]
                                    ;mov    bx,[es:di+13h]
000041A7 268B5D13                   MOV     BX,[ES:DI+SF_ENTRY.sf_size+2]
000041AB 2B06[CE05]                 SUB     AX,[BYTPOS]
000041AF 1B1E[D005]                 SBB     BX,[BYTPOS+2]
000041B3 7226                       JB      short RDERR             ;Read starts past EOF
000041B5 750A                       JNZ     short ENUF              ;More than 64k to EOF
000041B7 09C0                       OR      AX,AX
000041B9 7420                       JZ      short RDERR             ;Read starts at EOF
000041BB 39C8                       CMP     AX,CX
000041BD 7302                       JAE     short ENUF              ;I/O fits
000041BF 89C1                       MOV     CX,AX                   ;Limit read to up til EOF
                                ENUF:
                                    ; MSDOS 3.3
                                    ;test   byte [es:di+4],8
                                    ;TEST   byte [ES:DI+SF_ENTRY.sf_attr],attr_volume_id
                                    ;jnz    short SET_ACC_ERR
                                    ;call   LOCK_CHECK
                                    ;jnb    short _READ_OK
                                    ;call   READ_LOCK_VIOLATION
                                    ;jnb    short ENUF
                                    ;retn

                                    ; MSDOS 6.0
000041C1 E868FE                     call    CHECK_READ_LOCK             ;IFS. check read lock ;AN000;
                                    ;JNC    short _READ_OK              ; There are no locks
                                    ;retn
                                    ; 06/02/2024
000041C4 72D5                       jc      short RET3

                                _READ_OK:
000041C6 C42E[8A05]                 LES     BP,[THISDPB]
000041CA E885FF                     CALL    BREAKDOWN

                                ; 10/02/2024
                                %if 0
                                    MOV     CX,[CLUSNUM] ; *
                                    ;;;
                                    ; 06/02/2023 (PCDOS 7.1 IBMDOS.COM)
                                    mov     dx,[CLUSNUM_HW] ; *
                                    ;;;
                                    call    FNDCLUS
                                            ; MSDOS 6.0                 ;M022 conditional removed here
                                    JC      short SET_ACC_ERR_DS  ; fix to take care of I24 fail
                                                                  ; migrated from 330a - HKN
                                %else
                                    ; 10/02/2024 - Retro DOS v5.0
000041CD E86715                     call    FNDCLUS_X ; *
000041D0 721A                       jc      short SET_ACC_ERR ; ds=ss
                                %endif
                                    ;;;
                                    ; 09/02/2024 (PCDOS 7.1 IBMDOS.COM)
000041D2 833E[F80A]00               cmp     word [CLSKIP_HW],0
000041D7 7502                       jnz     short RDERR
                                    ;;;

                                    ;OR     CX,CX
                                    ;JZ     short SKIPERR
                                    ; 06/02/2024
000041D9 E318                       jcxz    SKIPERR

                                RDERR:
000041DB B40E                       MOV     AH,0EH                  ;MS. read/data/fail ;AN000;
000041DD E97202                     jmp     WRTERR22

                                ;RDLASTJ:
                                    ;JMP    RDLAST                    ;M039

                                SETSFTJ2:
000041E0 E9B600                     JMP     SETSFT

                                CANOT_READ:
                                    ; MSDOS 3.3
                                    ;POP    CX              ;M039.
                                    ; MSDOS 3.3 & MSDOS 6.0
000041E3 59                         POP     CX              ;Clean stack.
000041E4 5B                         POP     BX
                                    ;;;
                                    ; 09/02/2024 (PCDOS 7.1 IBMDOS.COM)
                                    ;pop    word [CCONTENT_HW] ; PCDOS 7.1 BUG!?
                                            ; I think, this would be 'pop word [ss:CCONTENT_HW]'
                                            ; because ds<>ss while jumping here.
                                            ; Erdogan Tan - 09/02/2024
000041E5 368F06[EC0A]               pop     word [ss:CCONTENT_HW] ; *
                                    ;;;

                                    ;entry  SET_ACC_ERR_DS
                                SET_ACC_ERR_DS:

                                ;hkn; SS is DOSDATA
                                    ;Context DS
000041EA 16                         push    ss
000041EB 1F                         pop     ds

                                    ;entry  SET_ACC_ERR
                                SET_ACC_ERR:
000041EC 31C9                       XOR     CX,CX
                                    ;mov    ax,5
000041EE B80500                     MOV     AX,error_access_denied
000041F1 F9                         STC
000041F2 C3                         retn

                                SKIPERR:
000041F3 8916[BA05]                 MOV     [LASTPOS],DX
000041F7 891E[BC05]                 MOV     [CLUSNUM],BX
                                    ;;;
                                    ; 09/02/2024 (PCDOS 7.1 IBMDOS.COM)
000041FB 8B1E[E80A]                 mov     bx,[CLUSTNUM_HW]
000041FF 891E[DE0A]                 mov     [CLUSNUM_HW],bx
                                    ;;;
00004203 833E[D205]00               CMP     word [BYTCNT1],0
00004208 7405                       JZ      short RDMID

0000420A E82016                     call    BUFRD
                                    ;JC     short SET_ACC_ERR_DS ; ds<>ss ; 10/02/2024
                                    ; 10/02/2024
                                    ; ds=ss
0000420D 72DD                       jc      short SET_ACC_ERR

                                RDMID:
```

```
0000420F 833E[D605]00               CMP     word [SECCNT],0
                                    ;JZ     RDLAST ; 10/08/2018
00004214 7466                       jz      short RDLAST

00004216 E8A816                     call    NEXTSEC
00004219 72C5                       JC      short SETSFTJ2

0000421B C606[7405]01               MOV     BYTE [TRANS],1        ; A transfer is taking place
                                ONSEC:
00004220 8A16[7305]                 MOV     DL,[SECCLUSPOS]       ; (dx/DL = Extent start) ((dh = ?))
00004224 8B0E[D605]                 MOV     CX,[SECCNT]
                                    ;;;
                                    ; 09/02/2024 (PCDOS 7.1 IBMDOS.COM)
00004228 8B1E[DE0A]                 mov     bx,[CLUSNUM_HW]
0000422C 891E[E80A]                 mov     [CLUSTNUM_HW],bx
                                    ;;;
00004230 8B1E[BC05]                 MOV     BX,[CLUSNUM]
                                RDLP:
00004234 E8D116                     call    OPTIMIZE
                                    ;JC     short SET_ACC_ERR_DS ; ds<>ss ; 10/02/2024
                                    ; 10/02/2024
                                    ; ds=ss
00004237 72B3                       jc      short SET_ACC_ERR
                                    ;;;
                                    ; 09/02/2024 (PCDOS 7.1 IBMDOS.COM)
00004239 FF36[EC0A]                 push    word [CCONTENT_HW]    ; (Next physical cluster, hw)
                                    ;;;
0000423D 57                         PUSH    DI                      ;DI = Next physical cluster.
0000423E 50                         PUSH    AX                      ;AX = # of sectors remaining.
0000423F 53                         PUSH    BX                    ;[DMAADD+2]:BX = Transfer address.
                                    ;mov    byte [ALLOWED],38h
00004240 C606[4B03]38               MOV     byte [ALLOWED],Allowed_RETRY+Allowed_FAIL+Allowed_IGNORE
00004245 8E1E[2E03]                 MOV     DS,[DMAADD+2]

00004249 52                         PUSH    DX                      ;[HIGH_SECTOR]:DX = phys. sector #.
0000424A 51                         PUSH    CX                      ;CX = # of contiguous sectors to read.

                                    ; 04/05/2019 - Retro DOS v4.0
                                    ; 09/02/2024 - Retro DOS v5.0
                                    ; PCDOS 7.1 IBMDOS.COM - DOSCODE:8284h
                                    ; Windows ME IO.SYS - BIOSCODE:0B69Eh
                                    ; MSDOS 6.22 IO.SYS - DOSCODE:787Bh

                                    ; NOTE: Secondary Buffer Cache is not used in PCDOS 7.1 IBMDOS.COM.

                                    ;call  nul_sub        ; PCDOS 7.1 (nul_sub: retn)
                                    ;                     ; 'nul_sub:' at DOSCODE:AD57h

                                    ; MSDOS 6.0 (& Windows ME)
                                    ;call  SET_RQ_SC_PARMS              ;LB. do this for SC ;AN000;

                                    ; MSDOS 3.3 (& MSDOS 6.0)
0000424B E8A0FD                     call    DREAD

                                    ; 10/02/2024
                                    ; ds<>ss

                                    ; MSDOS 3.3
                                    ;pop    bx
                                    ;pop    dx
                                    ;jc     short CANOT_READ
                                    ;add    bx,dx   ; (bx = Extent end)
                                    ;mov    al,[es:bp] ; mov al,[es:bp+0]
                                    ;;mov   al,[ES:BP+DPB.DRIVE]
                                    ;call   SETVISIT
                                    ; ->***
                                ;M039
                                    ; MSDOS 6.0
0000424E 59                         pop     cx
0000424F 5A                         pop     dx
00004250 368F06[0C06]               pop     WORD [ss:TEMP_VAR]
00004255 728C                       jc      short CANOT_READ ; * 09/02/2024

00004257 368C1E[0E06]               mov     [ss:TEMP_VAR2],ds

                                ;       CX = # of contiguous sectors read. (These constitute a block of
                                ;            sectors, also termed an "Extent".)
                                ;       [HIGH_SECTOR]:DX = physical sector # of first sector in extent.
                                ;       [TEMP_VAR2]:[TEMP_VAR] = Transfer address (destination data address).
                                ;       ES:BP -> Drive Parameter Block (DPB).
                                ;
                                ;   The Buffer Queue must now be scanned: the contents of any dirty
                                ;   buffers must be "read" into the transfer memory block, so that the
                                ;       transfer memory reflects the most recent data.

0000425C E87600                     call    DskRdBufScan

                                    ;Context DS
0000425F 16                         push    ss
00004260 1F                         pop     ds

00004261 59                         pop     cx
00004262 5B                             pop bx

                                ;       CX = # of sector remaining.
                                ;       BX = Next physical cluster.

                                    ;;;
                                    ; 09/02/2024 (PCDOS 7.1 IBMDOS.COM)
00004263 8F06[E80A]                 pop     word [CLUSTNUM_HW]    ; (Next physical cluster, hw)
                                    ;;;

                                ;M039

                                ;;;;;;;;
                                ;   ; 25/07/2018 - Retro DOS v3.0
                                ;   ; ***->
                                ;   ; MSDOS 3.3
                                ;   ; IBMDOS.COM (1987) - Offset 42BDh
                                ;bufq:
                                ;;  DX = Extent start.
                                ;;  BX = Extent end.
                                ;;   AL = Drive #.
                                ;;     DS:DI-> 1st buffer in queue.
                                ;
                                ;   ;or     byte [di+5],20h
                                ;   or      byte [DI+BUFFINFO.buf_flags],buf_visit ; Bit 5 = reserved
                                ;   ;cmp    al,[di+4]
                                ;   cmp     al,[DI+BUFFINFO.buf_ID]
                                ;   jnz     short bufq3
                                ;   ;cmp    [di+6],dx
                                ;   cmp     [DI+BUFFINFO.buf_sector],dx
                                ;   jb      short bufq3    ; Jump if Extent start > buffer sector.
```

```
                                ;   ;cmp    [di+6],bx
                                ;   cmp     [DI+BUFFINFO.buf_sector],bx
                                ;   jnb     short bufq3    ; Jump if Extent end >= buffer sector.
                                ;
                                ;   ; Buffer sector is in the Extent (contiguous sectors to read)
                                ;
                                ;;      Buffer's sector is in Extent: if it is dirty, copy its contents to
                                ;;      transfer memory; otherwise, just re-position it in the buffer queue
                                ;;      as MRU (Most Recently Used).
                                ;
                                ;   ;test   byte [di+5],40h
                                ;   test    byte [DI+BUFFINFO.buf_flags],buf_dirty ; Bit 6 = dirty flag
                                ;   jz      short bufq2    ; clear buffer, check the next buff sec
                                ;   pop     ax ; transfer address
                                ;   push    ax
                                ;   push    di
                                ;   push    dx
                                ;   ;sub    dx,[di+6]
                                ;   sub     dx,[DI+BUFFINFO.buf_sector]
                                ;   neg     dx
                                ;
                                ;;      DX = offset (in sectors) of buffer sector within Transfer memory
                                ;;           block.
                                ;
                                ;   mov     si,di
                                ;   mov     di,ax
                                ;   mov     ax,dx
                                ;   ;mov    cx,[es:bp+6]
                                ;   mov     cx,[ES:BP+DPB.SECTOR_SIZE] ; CX = sector size (in bytes).
                                ;   mul     cx
                                ;   add     di,ax
                                ;
                                ;   lea     si,[si+16]
                                ;   lea     si,[SI+BUFINSIZ] ;DS:SI -> buffer data.
                                ;   shr     cx,1
                                ;   push    es
                                ;   mov     es,[SS:DMAADD+2]
                                ;
                                ;;      CX = sector size (in WORDs) ; CF=1 if odd # of bytes.
                                ;;      DS:SI-> Buffer sector data.
                                ;;      ES:DI-> Destination within Transfer memory block.
                                ;
                                ;   rep     movsw                   ;Copy buffer sector to Transfer memory
                                ;   ;adc    cx,0                      ;CX=1 if odd # of bytes, else CX=0.
                                ;   ;rep    movsb                     ;Copy last byte.
                                ;   jnc     short bufq1
                                ;   movsb
                                ;bufq1:
                                ;   pop     es
                                ;   pop     dx
                                ;   pop     di
                                ;   mov     al,[es:bp]  ; mov al,[es:bp+0]
                                ;   ;mov    al,[ES:BP+DPB.DRIVE]
                                ;bufq2:
                                ;   call    SCANPLACE
                                ;bufq3:
                                ;   call    SKIPVISIT
                                ;   jnz     short bufq
                                ;
                                ;   push    ss
                                ;   pop     ds
                                ;   pop     cx
                                ;   pop     cx
                                ;   pop     bx
                                ;bufq4:
                                ;;;;;;;
                                
00004267 E313                       JCXZ    RDLAST

00004269 E84C20                     call    IsEOF                   ; test for eof on fat size
0000426C 732B                       JAE     short SETSFT

0000426E B200                       MOV     DL,0
                                    ;INC    word [LASTPOS]          ; We'll be using next cluster
                                    ;;;
                                    ; 09/02/2024 (PCDOS 7.1 IBMDOS.COM)
                                    ;;add   word [LASTPOS],1
                                    ;adc    word [LASTPOS_HW],0
                                    ; 09/02/2024 - Retro DOS v5.0
00004270 FF06[BA05]                 inc     word [LASTPOS]
00004274 75BE                       jnz     short RDLP
00004276 FF06[E20A]                 inc     word [LASTPOS_HW]
                                    ;;;
0000427A EBB8                       JMP     short RDLP ; 19/05/2019

                                RDLAST:
0000427C A1[D405]                   MOV     AX,[BYTCNT2]
0000427F 09C0                       OR      AX,AX
00004281 7416                       JZ      short SETSFT
00004283 A3[D205]                   MOV     [BYTCNT1],AX

00004286 E83816                     call    NEXTSEC
00004289 720E                       JC      short SETSFT

0000428B C706[CC05]0000             MOV     word [BYTSECPOS],0
00004291 E89915                     call    BUFRD
                                    ; 10/08/2018
00004294 7303                       JNC     short SETSFT
                                    ;JMP    SET_ACC_ERR_DS
                                    ; 10/02/2024
                                    ; ds=ss
00004296 E953FF                     jmp     SET_ACC_ERR

                                ;---------------------------------------------------------------------------
                                ;
                                ; Procedure Name : SETSFT
                                ; Inputs:
                                ;   [NEXTADD],[CLUSNUM],[LASTPOS] set to determine transfer size
                                ;           and set cluster fields
                                ; Function:
                                ;   Update [THISSFT] based on the transfer
                                ; Outputs:
                                ;   sf_position, sf_lstclus, and sf_cluspos updated
                                ;   ES:DI points to [THISSFT]
                                ;   CX No. of bytes transferred
                                ;   Carry clear
                                ;
                                ;---------------------------------------------------------------------------

                                    ;entry  SETSFT

                                ; 26/07/2018 - Retro DOS v3.0

                                    ; 09/02/2024 - Retro DOS v5.0
```

```
                                    ; (PCDOS 7.1 IBMDOS.COM)
                                
                                SETSFT:
00004299 C43E[9E05]                 LES     DI,[THISSFT]
                                
                                ; Same as SETSFT except ES:DI already points to SFT
                                    ;entry  SETCLUS
                                SETCLUS:
0000429D 8B0E[B805]                 MOV     CX,[NEXTADD]
000042A1 2B0E[2C03]                 SUB     CX,[DMAADD]           ; Number of bytes transfered
                                    ;;test  word [es:di+5],80h
                                    ;TEST   word [ES:DI+SF_ENTRY.sf_flags],devid_device
                                    ;JNZ    short ADDREC          ; don't set clusters if device
                                
                                    ; 04/05/2019 - Retro DOS v4.0
                                    ;test   byte [es:di+5],80h
000042A5 26F6450580                 TEST    byte [ES:DI+SF_ENTRY.sf_flags],devid_device
000042AA 751C                       JNZ     short ADDREC          ; don't set clusters if device
                                
                                    ;;;
                                    ; 09/02/2024 (PCDOS 7.1 IBMDOS.COM)
000042AC A1[DE0A]                   mov     ax,[CLUSNUM_HW]
000042AF 26894537                   mov     [es:di+37h],ax
                                    ;mov    [es:di+SF_ENTRY.sf_lstclus+2],ax
000042B3 A1[E20A]                   mov     ax,[LASTPOS_HW]
000042B6 2689450B                   mov     [es:di+0Bh],ax
                                    ;mov    [es:di+SF_ENTRY.sf_cluspos_hw],ax ; PCDOS 7.1 & Win ME
                                    ;;mov   [es:di+SF_ENTRY.sf_firclus],ax ; MSDOS 5.0-6.22
                                    ;;;
                                
000042BA A1[BC05]                   MOV     AX,[CLUSNUM]
                                    ;;mov   [es:di+1Bh],ax ; MSDOS 3.3
                                    ;mov    [es:di+35h],ax ; MSDOS 6.0 (& MSDOS 6.21)
000042BD 26894535                   MOV     [ES:DI+SF_ENTRY.sf_lstclus],AX
000042C1 A1[BA05]                   MOV     AX,[LASTPOS]
                                    ;mov    [es:di+19h],ax
000042C4 26894519                   MOV     [ES:DI+SF_ENTRY.sf_cluspos],AX
                                
                                ;--------------------------------------------------------------------------
                                ;
                                ; Procedure : AddRec
                                ; Inputs:
                                ;   ES:DI points to SFT
                                ;   CX is No. Bytes transferred
                                ; Function:
                                ;   Update the SFT offset based on the transfer
                                ; Outputs:
                                ;   sf_position updated to point to first byte after transfer
                                ;   ES:DI points to SFT
                                ;   CX No. of bytes transferred
                                ;   Carry clear
                                ;--------------------------------------------------------------------------
                                
                                    ;entry  AddRec
                                ADDREC:
000042C8 E309                       JCXZ    RET28         ; If no records read, don't change position
                                    ;add    [es:di+15h],cx
000042CA 26014D15                   ADD     [ES:DI+SF_ENTRY.sf_position],CX  ; Update current position
                                    ;adc    word [es:di+17h], 0
000042CE 2683551700                 ADC     WORD [ES:DI+SF_ENTRY.sf_position+2],0
                                RET28:
000042D3 F8                         CLC
000042D4 C3                         retn
                                
                                ; 25/07/2018
                                ; MSDOS 6.0
                                ;Break   <DskRdBufScan -- Disk Read Buffer Scan>
                                ;--------------------------------------------------------------------------
                                ;
                                ; Procedure Name : DskRdBufScan
                                ;
                                ; Inputs:
                                ;       CX = # of contiguous sectors read. (These constitute a block of
                                ;            sectors, also termed an "Extent".)
                                ;       [HIGH_SECTOR]:DX = physical sector # of first sector in extent.
                                ;       [TEMP_VAR2]:[TEMP_VAR] = Transfer address (destination data address).
                                ;       ES:BP -> Drive Parameter Block (DPB).
                                ;
                                ; Function:
                                ;   The Buffer Queue is scanned: the contents of any dirty buffers are
                                ;   "read" into the transfer memory block, so that the transfer memory
                                ;   reflects the most recent data.
                                ;
                                ; Outputs:
                                ;       Transfer memory updated as required.
                                ;
                                ; Uses:
                                ;       DS,AX,BX,CX,SI,DI destroyed.
                                ;       SS override for all global variables.
                                ;
                                ; Notes:
                                ;       FIRST_BUFF_ADDR is set-up to contain the LAST buffer to check, rather
                                ;   than the FIRST.
                                ;--------------------------------------------------------------------------
                                ;M039: Created
                                
                                ; 04/05/2019 - Retro DOS v4.0
                                ; DOSCODE:78F0h (MSDOS 6.21, MSDOS.SYS)
                                
                                ; 18/11/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
                                ; DOSCODE:78DCh (MSDOS 5.0, MSDOS.SYS)
                                
                                ; 09/02/2024 - Retro DOS v5.0
                                ; PCDOS 7.1 IBMDOS.COM - DOSCODE:8313h
                                
                                ;procedure DskRdBufScan,NEAR
                                ;
                                ;ASSUME  DS:NOTHING
                                
                                DskRdBufScan:
000042D5 36833E[7100]00             cmp     word [ss:DirtyBufferCount],0 ; Any dirty buffers?
000042DB 743C                       je      short bufx                   ; -no, skip all work.
                                
000042DD 368B1E[0706]               mov      bx,[ss:HIGH_SECTOR]
000042E2 89DE                       mov      si,bx
000042E4 01D1                       add      cx,dx
000042E6 83D600                     adc      si,0
                                
000042E9 E8CD25                     call    GETCURHEAD            ;DS:DI -> 1st buf in queue.
                                    ;mov    ax,[di+2]
000042EC 8B4502                     mov      ax,[di+BUFFINFO.buf_prev]
000042EF 36A3[7E11]                 mov      [ss:FIRST_BUFF_ADDR],ax
                                
                                    ; 18/11/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
```

```
                                    ;;mov   al,[es:bp+0]
                                    ;mov    al,[es:bp+DPB.DRIVE]
                                    ; 15/12/2022
000042F3 268A4600                   mov     al,[es:bp]

                                ;       BX:DX = Extent start.
                                ;       SI:CX = Extent end + 1.
                                ;          AL = Drive #.
                                ;       DS:DI-> 1st buffer in queue.
                                ;[FIRST_BUFF_ADDR] = Address offset of last buffer in queue.

                                bufq:
                                    ;cmp    al,[di+4]
000042F7 3A4504                     cmp     al,[di+BUFFINFO.buf_ID] ;Same drive?
000042FA 7514                       jne     short bufq1             ;  -no, jump.

                                ;       Cmp32   bx,dx,<WORD PTR [di.buf_sector+2]>,<WORD PTR [di.buf_sector]>
                                ;       ja  short bufq1             ;Jump if Extent start > buffer sector.

                                    ;cmp    bx,[di+8]
000042FC 3B5D08                     cmp     bx,[di+BUFFINFO.buf_sector+2]
000042FF 7503                       jne     short bufq01
                                    ;cmp    dx,[di+6]
00004301 3B5506                     cmp     dx,[di+BUFFINFO.buf_sector]
                                bufq01:
00004304 770A                       ja      short bufq1

                                ;       Cmp32   si,cx,<WORD PTR [di.buf_sector+2]>,<WORD PTR [di.buf_sector]>
                                ;       ja  short bufq2             ;Jump if Extent end >= buffer sector.

                                    ;cmp    si,[di+8]
00004306 3B7508                     cmp     si,[di+BUFFINFO.buf_sector+2]
00004309 7503                       jne     short bufq02
                                    ;cmp    cx,[di+6]
0000430B 3B4D06                     cmp     cx,[di+BUFFINFO.buf_sector]
                                bufq02:
0000430E 770A                       ja      short bufq2
                                bufq1:
00004310 363B3E[7E11]               cmp     di,[ss:FIRST_BUFF_ADDR]     ;Scanned entire buffer queue?
00004315 8B3D                       mov     di,[di]
                                    ;mov    di,[di+BUFFINFO.buf_next] ; Set-up for next buffer.
00004317 75DE                       jne     short bufq              ; -no, do next buffer
                                bufx:
00004319 C3                         retn                            ;Exit.

                                ;       Buffer's sector is in Extent: if it is dirty, copy its contents to
                                ;   transfer memory; otherwise, just re-position it in the buffer queue
                                ;       as MRU (Most Recently Used).

                                bufq2:
0000431A 50                         push    ax
                                    ;test   byte [di+5],40h
0000431B F6450540                   test    byte [di+BUFFINFO.buf_flags],buf_dirty ;Buffer dirty?
0000431F 7430                       jz      short bufq3                     ; -no, jump.

                                ;       SaveReg <cx,dx,si,di,es>
00004321 51                         push    cx
00004322 52                         push    dx
00004323 56                         push    si
00004324 57                         push    di
00004325 06                         push    es

00004326 89D0                       mov     ax,dx
                                    ;sub    ax,[di+6]
00004328 2B4506                     sub     ax,[di+BUFFINFO.buf_sector]
0000432B F7D8                       neg     ax

                                ;       AX = offset (in sectors) of buffer sector within Transfer memory
                                ;            block. (Note: the upper word of the sector # may be ignored
                                ;         since no more than 64k bytes will ever be read. This 64k limit
                                ;             is imposed by the input parameters of the disk read operation.)

                                    ;lea    si,[di+20]
0000432D 8D7518                     lea     si,[di+BUFINSIZ]        ;DS:SI -> buffer data.
                                    ;mov    cx,[es:bp+2]
00004330 268B4E02                   mov     cx,[es:bp+DPB.SECTOR_SIZE] ;CX = sector size (in bytes).
00004334 F7E1                       mul     cx                      ;AX = offset (in bytes) of buf. sector
                                    ;mov    di,[ss:TEMP_VAR]
                                    ; 09/02/2024
00004336 36C43E[0C06]               les     di,[ss:TEMP_VAR]
0000433B 01C7                       add     di,ax
                                    ;mov    es,[ss:TEMP_VAR2]
0000433D D1E9                       shr     cx,1

                                ;      CX = sector size (in WORDs) ; CF=1 if odd # of bytes.
                                ;       DS:SI-> Buffer sector data.
                                ;       ES:DI-> Destination within Transfer memory block.

                                ; 09/02/2024 - Retro DOS v5.0
                                %if 0
                                    rep     movsw                   ;Copy buffer sector to Transfer memory
                                    ;; 04/05/2019
                                    ;;adc   cx,0                     ;CX=1 if odd # of bytes, else CX=0.
                                    ;;rep   movsb                    ;Copy last byte.
                                    ;jnc    short bufq03
                                    ;movsb
                                    ; 18/11/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
                                    ;adc    cx,0
                                    ;rep    movsb
                                    ; 22/09/2023
                                    jnc     short bufq03
                                    movsb
                                %else
                                    ;;;
                                    ; 09/02/2024 (PCDOS 7.1 IBMDOS.COM)
0000433F 36803E[6A00]00             cmp     byte [ss:DDMOVE],0
                                    ;jz     short bufq03+1 ; rep movsw (skip 32bit prefix)
                                    ; 06/04/2024
00004345 7403                       jz      short bufq03
00004347 D1E9                       shr     cx,1
                                ;bufq03:
                                    ;rep    movsd
                                    ; 06/04/2024
00004349 66                         db      66h     ; 32bit prefix
                                bufq03:
0000434A F3A5                       rep     movsw
                                    ;;;
                                %endif

                                ;bufq03:    ; 09/02/2024
                                    ;RestoreReg <es,di,si,dx,cx>
0000434C 07                         pop     es
0000434D 5F                         pop     di
0000434E 5E                         pop     si
```

```
26646 0000434F 5A                          pop     dx
26647 00004350 59                          pop     cx
26648
26649                                   ;       DS:DI -> current buffer.
26650                                   bufq3:
26651 00004351 89F8                        mov     ax,di                ;DS:AX -> Current buffer.
26652                                          ;invoke SCANPLACE
26653 00004353 E87525                      call    SCANPLACE
26654 00004356 363B06[7E11]                cmp     ax,[ss:FIRST_BUFF_ADDR] ;Last buffer?
26655 0000435B 58                          pop     ax
26656                                      ;jne    short bufq           ; -no, jump.
26657                                      ;;jmp   short bufx           ; -yes, exit.
26658                                      ;; 12/06/2019
26659                                      ;retn
26660                                      ; 18/11/2022 (MSDOS 5.0 MSDOS.SYS compability)
26661 0000435C 7599                        jne     short bufq
26662                                      ;jmp    short bufx
26663                                      ; 09/02/2024
26664 0000435E C3                          retn    ; Exit
26665
26666                                   ;EndProc DskRdBufScan
26667
26668                                   ;============================================================================
26669                                   ; DISK3.ASM, MSDOS 6.0, 1991
26670                                   ;============================================================================
26671                                   ; 04/05/2019 - Retro DOS v4.0
26672                                   ; 24/07/2018 - Retro DOS v3.0
26673
26674                                   ;Break   <DISKWRITE -- PERFORM USER DISK WRITE>
26675                                   ;--------------------------------------------------------------------------
26676                                   ;
26677                                   ; Procedure Name : DISKWRITE
26678                                   ;
26679                                   ; Inputs:
26680                                   ;       Outputs of SETUP
26681                                   ; Function:
26682                                   ;       Perform disk write
26683                                   ; Outputs:
26684                                   ;    Carry clear
26685                                   ;       CX = No. of bytes written
26686                                   ;       ES:DI point to SFT
26687                                   ;       SFT offset and cluster pointers updated
26688                                   ;    Carry set
26689                                   ;       CX = 0
26690                                   ;       ES:DI point to SFT
26691                                   ;       AX has error code
26692                                   ;--------------------------------------------------------------------------
26693
26694                                   ; DOSCODE:797Ah (MSDOS 6.21, MSDOS.SYS)
26695
26696                                   ; 20/11/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
26697                                   ; DOSCODE:7966h (MSDOS 5.0, MSDOS.SYS)
26698
26699                                   ; 10/02/2024
26700                                   ; 09/02/2024 - Retro DOS v5.0
26701                                   ; PCDOS 7.1 IBMDOS.COM - DOSCODE:83A3h
26702
26703                                   ; (Windows ME IO.SYS - BIOSCODE:0B7B2h)
26704                                   ; (MSDOS 6.22 MSDOS.SYS - DOSCODE:797Ah)
26705
26706                                   DISKWRITE:
26707                                       ; MSDOS 3.3
26708                                       ; IBMDOS.COM - Offset 436Dh
26709                                       ;;test  byte [es:di+4],8
26710                                       ;TEST   byte [ES:DI+SF_ENTRY.sf_attr],attr_volume_id
26711                                       ;jz     short write_cont
26712                                       ;jmp    SET_ACC_ERR_DS
26713                                   ;write_cont:
26714                                       ;push   cx
26715                                       ;or     cx,cx
26716                                       ;jnz    short Not_Truncate
26717                                       ;;mov   cx,-1
26718                                       ;dec    cx
26719                                   ;Not_Truncate:
26720                                       ;call   LOCK_CHECK
26721                                       ;pop    cx
26722                                       ;jnb    short _WRITE_OK
26723                                       ;call   WRITE_LOCK_VIOLATION
26724                                       ;jnb    short DISKWRITE
26725                                       ;retn
26726
26727                                       ; MSDOS 6.0
26728 0000435F E8ACFC                      call    CHECK_WRITE_LOCK      ;IFS. check write lock;AN000;
26729                                       ; 19/08/2018
26730 00004362 7304                        JNC     short _WRITE_OK              ;IFS. lock check ok   ;AN000;
26731 00004364 C3                          retn
26732
26733                                   WRTEOFJ:
26734 00004365 E95602                      JMP     WRTEOF
26735
26736                                   _WRITE_OK:
26737                                       ; 27/07/2018
26738                                       ; IBMDOS.COM - Offset 438Eh
26739
26740                                       ; MSDOS 3.3 (& MSDOS 6.0)
26741                                       ;and    word [es:di+5],0BFBFh
26742 00004368 26816505BFBF                AND     word [ES:DI+SF_ENTRY.sf_flags],~(sf_close_nodate|devid_file_clean)
26743                                                                   ; Mark file as dirty, clear no date on close
26744                                   ; 10/02/2024
26745                                   %if 0
26746                                       ; 04/05/2019 - Retro DOS v4.0
26747
26748                                       ; MSDOS 6.0
26749                                       ;mov    ax,[es:di+11h]
26750                                       MOV     AX,[ES:DI+SF_ENTRY.sf_size]          ;M039
26751                                           MOV [TEMP_VAR],AX                           ;M039
26752                                       ;mov    ax,[es:di+13h]
26753                                       MOV     AX,[ES:DI+SF_ENTRY.sf_size+2]        ;M039
26754                                           MOV [TEMP_VAR2],AX                          ;M039
26755                                   %else
26756                                       ; 10/02/2024 (PCDOS 7.1 IBMDOS COM)
26757                                       ;les    ax,[es:di+11h]
26758 0000436E 26C44511                    les     ax,[es:di+SF_ENTRY.sf_size]
26759 00004372 8C06[0E06]                  mov     [TEMP_VAR2],es
26760 00004376 A3[0C06]                    mov     [TEMP_VAR],ax
26761                                   %endif
26762
26763                                   ;   TEMP_VAR2:TEMP_VAR = Current file size (sf_size);M039
26764
26765                                       ; MSDOS 3.3 (& MSDOS 6.0)
26766 00004379 C42E[8A05]                  LES     BP,[THISDPB]
26767
26768 0000437D E8D2FD                      call    BREAKDOWN
26769
```

```
26770 00004380 A1[CE05]                 MOV     AX,[BYTPOS]
26771 00004383 8B16[D005]               MOV     DX,[BYTPOS+2]
26772 00004387 E3DC                     JCXZ    WRTEOFJ                 ;Make the file length = sf_position
26773 00004389 01C8                     ADD     AX,CX
26774 0000438B 83D200                   ADC     DX,0                    ;DX:AX = last byte to write + 1.
26775
26776                                   ;;;
26777                                   ; 09/02/2024 (PCDOS 7.1 IBMDOS.COM)
26778 0000438E 7303                     jnc     short dskwrt_1
26779                                 ACC_ERRWJ2:
26780                                   ;;jmp   SET_ACC_ERRW
26781                                   ;jmp    SET_ACC_ERR_DS ; ds<>ss ; 10/02/2024
26782                                   ; 10/02/2024
26783                                   ; ds=ss
26784 00004390 E959FE                   jmp     SET_ACC_ERR
26785
26786                                 dskwrt_1:
26787                                   ;test   byte [es:di+3],10h
26788 00004393 26F6450310               test    byte [es:di+SF_ENTRY.sf_mode+1],10h
26789 00004398 750B                     jnz     short dskwrt_3          ; > 2GB file size (up to 4GB) allowed
26790 0000439A 81FAFF7F                 cmp     dx,7FFFh                ; check for 2GB file size limit
26791 0000439E 7503                     jne     short dskwrt_2
26792 000043A0 83F8FF                   cmp     ax,0FFFFh
26793                                 dskwrt_2:
26794 000043A3 77EB                     ja      short ACC_ERRWJ2 ; error,
26795                                                                   ; file position/pointer overs 2GB limit!
26796                                 dskwrt_3:
26797                                   ;;;
26798
26799                                   ;mov    bx,[es:bp+2]
26800 000043A5 268B5E02                 MOV     BX,[ES:BP+DPB.SECTOR_SIZE]
26801
26802                                   ; MSDOS 3.3
26803                                   ;cmp    dx,bx
26804                                   ;jnb    short WRTERR33
26805                                   ;div    bx
26806                                   ;mov    bx,ax
26807                                   ;OR     DX,DX
26808                                   ;JNZ    short CALCLUS
26809                                   ;dec    ax
26810                                 ;CALCLUS:
26811                                   ; MSDOS 3.3
26812                                   ;mov    cl,[es:bp+5]
26813                                   ;MOV    CL,[ES:BP+DPB.CLUSTER_SHIFT]
26814                                   ;shr    ax,cl
26815                                   ;push   ax
26816                                   ;push   dx
26817                                   ;push   es
26818                                   ;les    di,[THISSFT]
26819                                   ;;mov   ax,[es:di+11h]
26820                                   ;;mov   dx,[es:di+13h]
26821                                   ;mov    ax,[ES:DI+SF_ENTRY.sf_size]
26822                                   ;mov    dx,[ES:DI+SF_ENTRY.sf_size+2]
26823                                   ;pop    es
26824                                   ;;DX:AX = current file size (in bytes).
26825                                   ;;div   word [es:bp+2]
26826                                   ;div    word [ES:BP+DPB.SECTOR_SIZE]
26827                                   ;mov    cx,ax
26828                                   ;or     dx,dx
26829                                   ;jz     short NORND
26830                                   ;inc    ax
26831                                 ;NORND:
26832                                   ; MSDOS 6.0
26833 000043A9 E85F03                   CALL    DIV32                   ;DX:AX/BX = CX:AX + DX (rem.).
26834 000043AC 89C6                     MOV     SI,AX
26835 000043AE 890E[0706]                   MOV [HIGH_SECTOR],CX
26836
26837                                 ;       [HIGH_SECTOR]:SI = Last full sector to write.
26838
26839 000043B2 09D2                     OR      DX,DX
26840 000043B4 52                       PUSH    DX                    ;M039: Free DX for use by SHR32
26841 000043B5 89CA                     MOV     DX,CX                 ;M039
26842 000043B7 7506                     JNZ     short CALCLUS
26843 000043B9 83E801                   SUB     AX,1                    ;AX must be zero base indexed      ;AC000;
26844 000043BC 83DA00                   SBB     DX,0                  ;M039 ;F.C. >32mb             ;AN000;
26845
26846                                 CALCLUS:
26847                                   ; MSDOS 6.0
26848 000043BF E87003                   CALL    SHR32                   ;F.C. >32mb                 ;AN000;
26849                                   ;POP    DX
26850                                   ;;;
26851                                   ; 09/02/2024 (PCDOS 7.1 IBMDOS.COM)
26852 000043C2 59                       pop     cx
26853 000043C3 87CA                     xchg    cx,dx
26854 000043C5 51                       push    cx ; **!
26855                                   ; cx:ax = Last cluster to write
26856                                   ;;;
26857
26858                                 ;       AX = Last cluster to write.
26859                                 ;       DX = # of bytes in last sector to write (the "tail").
26860                                 ;       BX = [ES:BP+DPB.SECTOR_SIZE]
26861
26862 000043C6 50                       PUSH    AX ; *!
26863 000043C7 52                       PUSH    DX
26864                                 ;M039
26865 000043C8 8B16[0E06]               mov     dx,[TEMP_VAR2]
26866 000043CC A1[0C06]                 mov     ax,[TEMP_VAR]           ;DX:AX = current file size (in bytes).
26867 000043CF E83903                   call    DIV32                 ;DX:AX/BX = CX:AX + DX (rem.)
26868 000043D2 890E[0E06]               mov     [TEMP_VAR2],cx
26869 000043D6 890E[CA05]               mov     [VALSEC+2],cx
26870 000043DA 89C1                     mov     cx,ax
26871 000043DC 89F3                     mov     bx,si
26872
26873                                 ;       [HIGH_SECTOR]:BX = Last full sector to write.
26874                                 ;          [VALSEC+2]:CX = Last full sector of current file.
26875                                 ;         [TEMP_VAR2]:CX = Last full sector of current file.
26876                                 ;                          DX = # of bytes in last sector of current file.
26877                                 ;M039
26878 000043DE 09D2                     OR      DX,DX
26879 000043E0 7407                     JZ      short NORND
26880                                   ;ADD    AX,1                    ;Round up if any remainder    ;AC000;
26881                                   ;ADC    word [VALSEC+2],0
26882                                   ; 22/09/2023
26883 000043E2 40                       inc     ax  ; 0FFFFh -> 0
26884 000043E3 7504                     jnz     short NORND
26885 000043E5 FF06[CA05]               inc     word [VALSEC+2]
26886                                 NORND:
26887                                   ; MSDOS 3.3 & MSDOS 6.0
26888 000043E9 A3[C805]                 MOV     [VALSEC],AX
26889
26890                                 ;       [VALSEC] = Last sector of current file.
26891
26892 000043EC 31C0                     XOR     AX,AX
26893 000043EE A3[DE05]                 MOV     [GROWCNT],AX
```

```
                                    MOV     [GROWCNT+2],AX
                                    POP     AX              ; # of bytes in last sector to write (the "tail").

                                    ; MSDOS 6.0
                                    MOV     DI,[HIGH_SECTOR]        ;F.C. >32mb                ;AN000;
                                    CMP     DI,[TEMP_VAR2]          ;M039; F.C. >32mb          ;AN000;
                                    JB      short NOGROW            ;F.C. >32mb                  ;AN000;
                                    JZ      short lowsec            ;F.C. >32mb                  ;AN000;
                                    SUB     BX,CX                     ;F.C. >32mb                  ;AN000;
                                    SBB     DI,[TEMP_VAR2]          ;M039; F.C. >32mb di:bx no. of sectors ;AN000;
                                    JMP     short yesgrow             ;F.C. >32mb                  ;AN000;
                                lowsec:
                                    ;MOV   DI,0                    ;F.C. >32mb
                                    ; 22/09/2023
                                    xor     di,di
                                    ; MSDOS 3.3 & MSDOS 6.0
                                    SUB     BX,CX                   ; Number of full sectors
                                    JB      short NOGROW
                                    JZ      short TESTTAIL
                                yesgrow:
                                    ; MSDOS 3.3 (& MSDOS 6.0)
                                    MOV     CX,DX
                                    XCHG    AX,BX
                                    ;mul    word [es:bp+2]
                                    MUL     word [ES:BP+DPB.SECTOR_SIZE] ; Bytes of full sector growth

                                    ; MSDOS 6.0
                                    MOV     [HIGH_SECTOR],DX        ;F.C. >32mb save dx          ;AN000;
                                    MOV     [TEMP_VAR2],AX          ;M039; F.C. >32mb save ax    ;AN000;
                                    MOV     AX,DI                   ;F.C. >32mb                  ;AN000;
                                    ;mul    word [es:bp+2]
                                    MUL     word [ES:BP+DPB.SECTOR_SIZE] ;F.C. >32mb do higher word multiply ;AN000;

                                    ADD     AX,[HIGH_SECTOR]        ;F.C. >32mb add lower value  ;AN000;
                                    ;MOV   DX,AX                   ;F.C. >32mb DX:AX is the result of ;AN000;
                                    ; 09/02/2024
                                    xchg    ax,dx
                                    MOV     AX,[TEMP_VAR2]          ;M039; F.C. >32mb a 32 bit multiply ;AN000;

                                    ; MSDOS 3.3 (& MSDOS 6.0)
                                    SUB     AX,CX                   ; Take off current "tail"
                                    SBB     DX,0                    ; 32-bit extension
                                    ADD     AX,BX                   ; Add on new "tail"
                                    ADC     DX,0                    ; ripple tim's head off
                                    JMP     SHORT SETGRW

                                HAVSTART:
                                    ;int 3
                                    ;;;
                                    ; 09/02/2024 (PCDOS 7.1 IBMDOS.COM)
                                    mov     [CLSKIP_HW],si
                                    ;;;
                                    MOV     CX,AX
                                    call    SKPCLP
                                    ;;;
                                    ; 09/02/2024
                                    mov     ax,[CLSKIP_HW]
                                    cmp     ax,cx
                                    jne     short HAVSTART_cont
                                    ;;;
                                    ; 10/02/2024
                                    JCXZ    DOWRTJ
                                    ; 16/12/2022
                                    ;jcxz   DOWRT
                                    ; 20/11/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
                                    ;jcxz   DOWRTJ
                                    ;;;
                                    ; 09/02/2024
                                HAVSTART_cont:
                                    ;mov    [CCOUNT_HW],ax
                                    ;call   ALLOCATE
                                    ; 07/07/2024
                                    call    ALLOCATE_X
                                    ; 10/02/2024
                                    JNC     short DOWRTJ
                                    ; 16/12/2022
                                    ;jnc    short DOWRT
                                    ; 20/11/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
                                    ;jnc    short DOWRTJ

                                    ; 10/02/2024
                                    ;entry   WRTERR
                                WRTERR:
                                    MOV     AH,0FH                  ;MS. write/data/fail/abort    ;AN000;

                                    ;entry WRTERR22
                                WRTERR22:
                                    MOV     AL,[THISDRV]            ;MS.                          ;AN000;

                                    ; 27/07/2018
                                WRTERR33:
                                    ;MOV    CX,0                    ;No bytes transferred
                                    XOR     CX,CX

                                    LES     DI,[THISSFT]
                                    ;CLC ; 19/05/2019
                                    ; 20/11/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
                                    ; 16/12/2022
                                    ;clc
                                    retn

                                    ; 10/02/2024
                                    ; 16/12/2022
                                    ; 20/11/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
                                DOWRTJ:
                                    JMP     short DOWRT

                                TESTTAIL:
                                    SUB     AX,DX
                                    JBE     short NOGROW
                                    XOR     DX,DX
                                SETGRW:
                                    MOV     [GROWCNT],AX
                                    MOV     [GROWCNT+2],DX
                                NOGROW:
                                    ; 09/02/2024
                                    ;POP    AX ; *!

                                ; 10/02/2024
                                %if 0
                                    MOV     CX,[CLUSNUM] ; *+ ; First cluster accessed
                                    ;;;
                                    ; 09/02/2024
                                    mov     dx,[CLUSNUM_HW] ; *+
```

```
                                        ;;;
                                        call    FNDCLUS
                                    %else
                                        ; 10/02/2024 - Retro DOS v5.0
0000446B E8C912                         call    FNDCLUS_X ; *+
                                    %endif
                                        ;;;
                                        ; 09/02/2024 (PCDOS 7.1 IBMDOS.COM)
0000446E 58                             pop     ax ; *!
0000446F 5E                             pop     si ; **!        ; si:ax = Last cluster
                                        ;;;
00004470 7279                           JC      short ACC_ERRWJ ; ds=ss
00004472 891E[BC05]                     MOV     [CLUSNUM],BX
00004476 8916[BA05]                     MOV     [LASTPOS],DX

0000447A 29D0                           SUB     AX,DX           ; Last cluster minus current cluster
                                        ; 09/02/2024
                                        ;JZ     short DOWRT     ; If we have last clus, we must have first
                                        ;;;
                                        ; 09/02/2024 (PCDOS 7.1 IBMDOS.COM)
0000447C 1B36[E20A]                     sbb     si,[LASTPOS_HW]
00004480 8B16[E80A]                     mov     dx,[CLUSTNUM_HW]
00004484 8916[DE0A]                     mov     [CLUSNUM_HW],dx
00004488 7504                           jnz     short NOGROW2
0000448A 09C0                           or      ax,ax
0000448C 7449                           jz      short DOWRT
                                    NOGROW2:
0000448E 3B0E[F80A]                     cmp     cx,[CLSKIP_HW]
00004492 7502                           jne     short dskwrt_4
                                        ;;;
00004494 E3A3                           JCXZ    HAVSTART        ; See if no more data
                                    dskwrt_4:   ; 09/02/2024
00004496 51                             PUSH    CX              ; No. of clusters short of first
                                        ;;;
                                        ; 09/02/2024 (PCDOS 7.1 IBMDOS.COM)
00004497 FF36[F80A]                     push    word [CLSKIP_HW]
0000449B 8936[F00A]                     mov     [CCOUNT_HW],si
                                        ;;;
0000449F 89C1                           MOV     CX,AX
000044A1 E83515                         call    ALLOCATE
                                        ;;;
                                        ; 09/02/2024
000044A4 8F06[F80A]                     pop     word [CLSKIP_HW]
                                        ;;;
000044A8 59                             POP     CX
000044A9 72A5                           JC      short WRTERR
000044AB 8B16[BA05]                     MOV     DX,[LASTPOS]

                                    ; 09/02/2024
                                    %if 0
                                        INC     DX
                                        DEC     CX
                                        JZ      short NOSKIP
                                    %else
                                        ;;;
                                        ; 09/02/2024 Retro DOS v5.0
                                        ; (PCDOS 7.1 IBMDOS.COM)
                                        ;add    dx,1
                                        ;adc    word [LASTPOS_HW],0
000044AF 42                             inc     dx
000044B0 7504                           jnz     short dskwrt_10
000044B2 FF06[E20A]                     inc     word [LASTPOS_HW]
                                    dskwrt_10:
000044B6 83E901                         sub     cx,1
000044B9 831E[F80A]00                   sbb     word [CLSKIP_HW],0
000044BE 7504                           jnz     short dskwrt_5
000044C0 09C9                           or      cx,cx
000044C2 7405                           jz      short NOSKIP
                                    dskwrt_5:
                                        ;;;
                                    %endif
000044C4 E8D512                         call    SKPCLP
000044C7 7222                           JC      short ACC_ERRWJ ; ds=ss ; 10/02/2024

                                    NOSKIP:
000044C9 891E[BC05]                     MOV     [CLUSNUM],BX
                                        ;;;
                                        ; 09/02/2024 (PCDOS 7.1 IBMDOS.COM)
000044CD A1[E80A]                       mov     ax,[CLUSTNUM_HW]
000044D0 A3[DE0A]                       mov     [CLUSNUM_HW],ax
                                        ;;;
000044D3 8916[BA05]                     MOV     [LASTPOS],DX
                                    DOWRT:
000044D7 833E[D205]00                   CMP     word [BYTCNT1],0
000044DC 7410                           JZ      short WRTMID
                                        ;;;
                                        ; 09/02/2024
000044DE 8B1E[DE0A]                     mov     bx,[CLUSNUM_HW]
000044E2 891E[E80A]                     mov     [CLUSTNUM_HW],bx
                                        ;;;
                                        ; 09/02/2024
                                        ;MOV    BX,[CLUSNUM]    ; (not used in 'BUFWRT') ; 09/02/2024
000044E6 E87D13                         call    BUFWRT
                                        ;JC     short ACC_ERRWJ
                                        ; 09/02/2024
000044E9 7303                           jnc     short WRTMID

                                        ; 09/02/2024
                                    ACC_ERRWJ:
                                        ; 10/08/2018
                                        ;;JMP   SET_ACC_ERRW
                                        ; 16/12/2022
                                        ;jmp    SET_ACC_ERR_DS ; ds<>ss ; 10/02/2024
                                        ; 10/02/2024
                                        ; ds=ss
000044EB E9FEFC                         jmp     SET_ACC_ERR
                                        ; 20/11/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
                                        ;;jmp   SET_ACC_ERRW

                                    WRTMID:
000044EE A1[D605]                       MOV     AX,[SECCNT]
000044F1 09C0                           OR      AX,AX
                                        ; 20/11/2022
                                        ;JZ     short WRTLAST  ; 24/07/2019   ;M039
                                        ; 10/02/2024
000044F3 7503                           jnz     short WRTMID2
000044F5 E98600                         jmp      WRTLAST
                                    WRTMID2:
000044F8 0106[C405]                     ADD     [SECPOS],AX
                                        ; 19/05/2019
                                        ; MSDOS 6.0
000044FC 8316[C605]00                   ADC     WORD [SECPOS+2],0     ;F.C. >32mb    ;AN000;
00004501 E8BD13                         call    NEXTSEC
                                        ; 16/12/2022
```

```
00004504 72E5                           JC      short ACC_ERRWJ
                                        ;JC     short SET_ACC_ERRW      ;M039
00004506 C606[7405]01                   MOV     BYTE [TRANS],1          ; A transfer is taking place
0000450B 8A16[7305]                     MOV     DL,[SECCLUSPOS]         ; (dx/DL = Extent start) ((dh = ?))
                                        ;;;
                                        ; 09/02/2024 (PCDOS 7.1 IBMDOS.COM)
0000450F 8B1E[DE0A]                     mov     bx,[CLUSNUM_HW]
00004513 891E[E80A]                     mov     [CLUSTNUM_HW],bx
                                        ;;;
00004517 8B1E[BC05]                     MOV     BX,[CLUSNUM]
0000451B 8B0E[D605]                     MOV     CX,[SECCNT]
                                    WRTLP:
0000451F E8E613                         call    OPTIMIZE
                                        ; 09/02/2024
                                        ;JC     short SET_ACC_ERRW
                                        ; 16/12/2022
00004522 72C7                           JC      short ACC_ERRWJ  ; ds=ss ; 10/02/2024

                                    ;M039
                                    ;       DI = Next physical cluster.
                                    ;       AX = # sectors remaining.
                                    ;       [DMAADD+2]:BX = transfer address (source data address).
                                    ;       CX = # of contiguous sectors to write. (These constitute a block of
                                    ;         sectors, also termed an "Extent".)
                                    ;       [HIGH_SECTOR]:DX = physical sector # of first sector in extent.
                                    ;       ES:BP -> Drive Parameter Block (DPB).
                                    ;
                                    ;       Purge the Buffer Queue and the Secondary Cache of any buffers which
                                    ;   are in Extent; they are being over-written.

                                        ;;;
                                        ; 09/02/2024 (PCDOS 7.1 IBMDOS.COM)
00004524 FF36[EC0A]                     push    word [CCONTENT_HW]
                                        ;;;

00004528 57                             push    di              ; CCONTENT_HW:DI = Next physical cluster
00004529 50                             push    ax              ; AX = # sectors remaining

                                        ; MSDOS 3.3
                                        ; IBMDOS.COM (1987) - Offset 4497h
                                        ;push   dx
                                        ;push   bx
                                        ;mov    al,[es:bp]
                                        ;;mov   AL,[ES:BP+DPB.DRIVE] ; mov al,[es:bp+0]
                                        ;mov    bx,cx
                                        ;add    bx,dx   ; (bx = Extent end)

                                    ;   DX = Extent start.
                                    ;   BX = Extent end.
                                    ;   AL = Drive #.

                                        ;call   SETVISIT

                                    ;wbufq1:
                                        ;;or    byte [di+5],20h
                                        ;or     byte [DI+BUFFINFO.buf_flags],buf_visit ; Bit 5 = reserved
                                        ;;cmp   al,[di+4]
                                        ;cmp    al,[DI+BUFFINFO.buf_ID]
                                        ;jnz    short wbufq2   ; Jump if Extent start > buffer sector.
                                        ;;cmp   [di+6],dx
                                        ;cmp    [DI+BUFFINFO.buf_sector],dx
                                        ;jb     short wbufq2
                                        ;;cmp   [di+6],bx
                                        ;cmp    [DI+BUFFINFO.buf_sector],bx
                                        ;jnb    short wbufq2   ; Jump if Extent end >= buffer sector.

                                        ;; Buffer sector is in the Extent

                                        ;;mov   word [di+4],20FFh
                                        ;mov    word [DI+BUFFINFO.buf_ID],20FFh
                                        ;                               ; .buf_ID,    AL = FFh (Free buffer)
                                        ;                               ; .buf_flags, AH = 0, reset/clear
                                        ;call   SCANPLACE
                                    ;wbufq2:
                                        ;call   SKIPVISIT
                                        ;jnz    short wbufq1
                                        ;pop    bx
                                        ;pop    dx

                                           ; MSDOS 6.0
0000452A E89101                         call    DskWrtBufPurge          ;DS trashed.

                                    ;ASSUME DS:NOTHING
                                    ;M039
                                        ; MSDOS 3.3 & MSDOS 6.0
                                    ;hkn; SS override for DMAADD and ALLOWED
0000452D 368E1E[2E03]                   MOV     DS,[SS:DMAADD+2]
                                        ;mov    byte [ss:ALLOWED],38h
00004532 36C606[4B03]38                 MOV     byte [SS:ALLOWED],Allowed_RETRY+Allowed_FAIL+Allowed_IGNORE

                                    ;   put logic from DWRITE in-line here so we can modify it
                                    ;   for DISK FULL conditions.

                                        ; 20/11/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
                                        ; DOSCODE:7AD8h (MSDOS 5.0 MSDOS.SYS)

                                        ; 16/12/2022
                                        ; MSDOS 3.3 (& MSDOS 5.0)
                                        ;call   DWRITE

                                    ;DWRITE_OKAY:

                                        ; 16/12/2022
                                        ; MSDOS 5.0 (& MSDOS 3.3)
                                        ;pop    cx
                                        ;pop    bx
                                        ;push   ss
                                        ;pop    ds
                                        ;jc     short SET_ACC_ERRW
                                        ;jcxz   WRTLAST
                                        ;mov    dl,0
                                        ;inc    word [LASTPOS]
                                        ;jmp    short WRTLP

                                        ; 16/12/2022
                                        ; 20/11/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
                                    DWRITE_LUP:
                                        ; 23/07/2019 - Retro DOS v3.2

                                        ; MSDOS 6.0
00004538 E82CFB                         call    DSKWRITE
0000453B 7417                           jz      short DWRITE_OKAY ; ds<>ss

                                    ;;  int     3
```

```
27266
27267 0000453D 3C27                      cmp     al,error_handle_Disk_Full     ; compressed volume full?
27268 0000453F 742D                      jz      short DWRITE_DISK_FULL
27269
27270                                    ; 16/12/2022
27271
27272                                   ;;hkn; SS override
27273 00004541 36C606[7505]01            MOV     BYTE [SS:READOP],1
27274 00004547 E84CFB                    call    HARDERRRW
27275 0000454A 3C01                      CMP     AL,1            ; Check for retry
27276 0000454C 74EA                      JZ      short DWRITE_LUP
27277
27278                                    ; 16/12/2022
27279                                    ; 23/07/2019
27280                                    ;POP    CX ; *4*
27281                                    ;POP    BX ; *5*
27282                                    ;
27283                                    ;push   ss
27284                                    ;pop    ds
27285                                    ;
27286
27287                                    ; 20/11/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
27288
27289                                    ; 16/12/2022
27290 0000454E 3C03                      CMP     AL,3            ; Check for FAIL
27291 00004550 F8                        CLC
27292 00004551 7501                      JNZ     short DWRITE_OKAY ; Ignore
27293 00004553 F9                        STC
27294
27295                                   DWRITE_OKAY:
27296                                    ; 16/12/2022
27297                                    ; 23/07/2019
27298                                    ; MSDOS 3.3 (& MSDOS 6.0)
27299 00004554 59                        POP     CX ; *4*
27300 00004555 5B                        POP     BX ; *5*
27301
27302                                   ;       CX = # sectors remaining.
27303                                   ;       BX = Next physical cluster.
27304
27305                                   ;hkn; SS override
27306                                             ;Context DS
27307                                    ; 16/12/2022
27308                                    ;push   ss
27309                                    ;pop    ds
27310
27311                                    ; 09/02/2024 - Retro DOS v5.0
27312                                    ; 16/12/2022
27313                                    ;jc     short SET_ACC_ERRW
27314
27315                                    ; 16/12/2022
27316 00004556 16                        push    ss
27317 00004557 1F                        pop     ds
27318
27319                                    ;;;
27320                                    ; 09/02/2024 - Retro DOS v5.0
27321                                    ; (PCDOS 7.1 IBMDOS.COM)
27322 00004558 8F06[E80A]                pop     word [CLUSTNUM_HW] ; CLUSTNUM_HW:BX = Next physical cluster
27323 0000455C 725D                      jc      short SET_ACC_ERRW ; ds=ss
27324                                    ;;;
27325
27326 0000455E E31E                      JCXZ    WRTLAST
27327
27328                                    ; 10/02/2024
27329 00004560 B200                      MOV     DL,0
27330                                    ;xor    dl,dl ; 23/07/2019
27331 00004562 FF06[BA05]                INC     word [LASTPOS]; We'll be using next cluster
27332                                    ;;;
27333                                    ; 09/02/2024 - Retro DOS v5.0
27334                                    ; (PCDOS 7.1 IBMDOS.COM)
27335 00004566 75B7                      jnz     short WRTLP
27336 00004568 FF06[E20A]                inc     word [LASTPOS_HW]
27337                                    ;;;
27338 0000456C EBB1                      JMP     short WRTLP
27339
27340                                    ; 23/07/2019 - Retro DOS v3.2
27341                                    ; 09/08/2018
27342                                    ; MSDOS 6.0
27343                                   DWRITE_DISK_FULL:
27344                                    ;Context DS            ;SQ 3-5-93 DS must be setup on return!
27345                                    ; 16/12/2022
27346 0000456E 16                        push    ss
27347 0000456F 1F                        pop     ds
27348 00004570 59                        pop     cx              ; unjunk stack
27349 00004571 5B                        pop     bx
27350                                    ;;;
27351                                    ; 09/02/2024 (PCDOS 7.1 IBMDOS.COM)
27352 00004572 8F06[E80A]                pop     word [CLUSTNUM_HW]
27353                                    ;;;
27354 00004576 C606[0B06]01              mov     byte [DISK_FULL],1
27355                                    ;stc
27356 0000457B E9D2FE                    jmp     WRTERR ; 24/07/2019 ; go to disk full exit
27357
27358                                   WRTLAST:
27359 0000457E A1[D405]                  MOV     AX,[BYTCNT2]
27360 00004581 09C0                      OR      AX,AX
27361 00004583 7413                      JZ      short FINWRT
27362 00004585 A3[D205]                  MOV     [BYTCNT1],AX
27363 00004588 E83613                    call    NEXTSEC
27364 0000458B 722E                      JC      short SET_ACC_ERRW
27365 0000458D C706[CC05]0000            MOV     word [BYTSECPOS],0
27366 00004593 E8D012                    call    BUFWRT
27367 00004596 7223                      JC      short SET_ACC_ERRW
27368
27369                                   FINWRT:
27370 00004598 C43E[9E05]                LES     DI,[THISSFT]
27371 0000459C A1[DE05]                  MOV     AX,[GROWCNT]
27372 0000459F 8B0E[E005]                MOV     CX,[GROWCNT+2]
27373 000045A3 09C0                      OR      AX,AX
27374 000045A5 7502                      JNZ     short UPDATE_size
27375 000045A7 E30F                      JCXZ    SAMSIZ
27376                                   UPDATE_size:
27377                                    ;add    [es:di+11h],ax
27378 000045A9 26014511                  ADD     [ES:DI+SF_ENTRY.sf_size],AX
27379                                    ;adc    [es:di+13h],cx
27380 000045AD 26114D13                  ADC     [ES:DI+SF_ENTRY.sf_size+2],CX
27381
27382                                   ; Make sure that all other SFT's see this growth also.
27383
27384 000045B1 B80100                    MOV     AX,1
27385                                   ;if installed
27386                                    ;Call   JShare + 14 * 4
27387 000045B4 FF1E[C800]                call    far [JShare+(14*4)]   ; 14 = ShSU
27388                                   ;else
27389                                   ;   Call    ShSU
```

```
                                ;endif
                                
                                SAMSIZ:
000045B8 E9E2FC                     jmp     SETCLUS ; ES:DI already points to SFT
                                
                                    ; 16/12/2022
                                    ; 20/11/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
                                SET_ACC_ERRW:
                                    ;jmp    SET_ACC_ERR_DS ; ds<>ss ; 10/02/2024
                                    ; 10/02/2024
                                    ; ds=ss
000045BB E92EFC                     jmp     SET_ACC_ERR
                                
                                WRTEOF:
000045BE 89C1                       MOV      CX,AX
000045C0 09D1                       OR       CX,DX
                                    ;JZ     short KILLFIL
                                    ; 10/02/2024
000045C2 7503                       jnz     short WRTEOF1
000045C4 E9A600                     jmp     KILLFIL
                                
                                    ;;;
                                    ; 10/02/2024 (PCDOS 7.1 IBMDOS.COM)
                                WRTEOF1:
000045C7 81FAFF7F                   cmp     dx,7FFFh        ; 2GB file size limit
000045CB 7503                       jne     short WRTEOF2
000045CD 83F8FF                     cmp     ax,0FFFFh
                                WRTEOF2:
000045D0 760D                       jna     short WRTEOF3
                                    ;
000045D2 06                         push    es
                                    ;les    di,ds:THISSFT
                                    ; 27/06/2024
000045D3 C43E[9E05]                 les     di,[THISSFT]
                                    ;test   byte [es:di+3],10h
000045D7 26F6450310                 test    byte [es:di+SF_ENTRY.sf_mode+1],10h
000045DC 07                         pop     es
000045DD 74DC                       jz      short SET_ACC_ERRW ; > 2GB file size not allowed
                                WRTEOF3:
                                    ;;;
                                
000045DF 83E801                     SUB      AX,1
000045E2 83DA00                     SBB      DX,0
                                
                                    ; MSDOS 3.3
                                    ;;div   word [es:bp+2]
                                    ;div    word [ES:BP+DPB.SECTOR_SIZE]
                                    ;;mov   cl,[es:bp+5]
                                    ;mov    cl,[ES:BP+DPB.CLUSTER_SHIFT]
                                    ;shr    ax,cl
                                
                                    ; MSDOS 6.0
000045E5 53                         PUSH    BX
                                    ;mov    bx,[es:bp+2]
000045E6 268B5E02                   MOV     BX,[ES:BP+DPB.SECTOR_SIZE]    ;F.C. >32mb                       ;AN000;
000045EA E81E01                     CALL    DIV32                         ;F.C. >32mb                       ;AN000;
000045ED 5B                         POP     BX                          ;F.C. >32mb                     ;AN000;
000045EE 89CA                       MOV     DX,CX                       ;M039
                                        MOV [HIGH_SECTOR],CX              ;M039: Probably extraneous, but not sure.
000045F0 890E[0706]
000045F4 E83B01                     CALL    SHR32                         ;F.C. >32mb                       ;AN000;
                                
000045F7 89C1                       MOV      CX,AX
000045F9 E84311                     call    FNDCLUS
                                SET_ACC_ERRWJ2:
000045FC 72BD                       JC      short SET_ACC_ERRW
                                    ;;;
                                    ; 10/02/2024 (PCDOS 7.1 IBMDOS.COM)
000045FE A1[F80A]                   mov     ax,[CLSKIP_HW]
00004601 39C8                       cmp     ax,cx
00004603 7502                       jne     short dskwrt_6
                                    ;;;
                                
00004605 E324                       JCXZ     RELFILE
                                
                                    ;;;
                                    ; 10/02/2024 (PCDOS 7.1 IBMDOS.COM)
                                dskwrt_6:
                                    ;mov    [CCOUNT_HW],ax
                                    ;;;;
                                    ;
                                    ;call   ALLOCATE
                                    ; 07/07/2024
00004607 E8CC13                     call    ALLOCATE_X
                                    ;JC     short WRTERRJ               ;;;;;;;;; disk full
                                    ; 16/12/2022
0000460A 7303                       jnc     short UPDATE
0000460C E941FE                     JMP     WRTERR
                                
                                UPDATE:
0000460F C43E[9E05]                 LES     DI,[THISSFT]
00004613 A1[CE05]                   MOV     AX,[BYTPOS]
                                    ;mov    [es:di+11h],ax
00004616 26894511                   MOV     [ES:DI+SF_ENTRY.sf_size],AX
0000461A A1[D005]                   MOV     AX,[BYTPOS+2]
                                    ;mov    [es:di+13h],ax
0000461D 26894513                   MOV     [ES:DI+SF_ENTRY.sf_size+2],AX
                                ;
                                ; Make sure that all other SFT's see this growth also.
                                ;
00004621 B80200                     MOV      AX,2
                                ;if installed
                                    ;Call   JShare + 14 * 4
00004624 FF1E[C800]                 call     far [JShare+(14*4)]   ; 14 = ShSU
                                ;else
                                ;   Call    ShSU
                                ;endif
00004628 31C9                       XOR      CX,CX ; 0
                                    ;jmp    ADDREC
                                    ; 08/02/2024
0000462A C3                         retn
                                
                                    ; 16/12/2022
                                ;WRTERRJ:
                                    ;JMP    WRTERR
                                
                                ;;;;;;;;;;;;;;;; 7/18/86
                                ;;;;;;;;;;;;;;;;
                                
                                RELFILE:
                                    ; MSDOS 6.0
0000462B 06                         PUSH    ES                      ;AN002; BL Reset Lstclus and cluspos to
0000462C C43E[9E05]                 LES     DI,[THISSFT]            ;AN002; BL beginning of file if current
                                    ;;;
                                    ; 10/02/2024 (PCDOS 7.1 IBMDOS.COM)
00004630 50                         push    ax                      ; cluspos is past EOF.
```

```
27514 00004631 A1[E20A]                 mov     ax,[LASTPOS_HW]
27515 00004634 263B450B                 cmp     ax,[es:di+0Bh]          ; [ES:DI+SF_ENTRY.sf_firclus]
27516                                   ;cmp    ax,[es:di+SF_ENTRY.sf_cluspos_hw]
27517 00004638 58                       pop     ax
27518 00004639 7504                     jne     short dskwrt_7
27519                                   ;;;
27520                                   ;cmp    dx,[es:di+19h]
27521 0000463B 263B5519                 CMP     DX,[ES:DI+SF_ENTRY.sf_cluspos]       ;AN002; BL cluspos is past EOF.
27522                                 dskwrt_7:   ; 10/02/2024
27523 0000463F 731C                     JAE     short SKIPRESET                      ;AN002; BL
27524                                   ;;;
27525                                   ; 10/02/2024 (PCDOS 7.1 IBMDOS.COM)
27526 00004641 268B552B                 mov     dx,[es:di+2Bh]          ; [ES:DI+SF_ENTRY.sf_chain]
27527                                   ;mov    dx,[es:di+SF_ENTRY_sf_fcluster]      ; first cluster (32 bit) lw !?
27528                                   ;;;
27529                                   ;mov    [es:di+19h],0
27530 00004645 26C745190000             MOV     word [ES:DI+SF_ENTRY.sf_cluspos],0 ;AN002; BL
27531                                 ; 10/02/2024
27532                                 %if 0
27533                                     ;mov    dx,[es:di+0Bh]
27534                                     MOV     DX,[ES:DI+SF_ENTRY.sf_firclus]       ;AN002; BL
27535                                 %else
27536                                     ;;;
27537                                     ; 10/02/2024 (PCDOS 7.1 IBMDOS.COM)
27538 0000464B 26C7450B0000             mov     word [es:di+0Bh],0
27539                                   ;mov    word [es:di+SF_ENTRY.sf_cluspos_hw],0
27540                                   ;;;
27541                                 %endif
27542                                   ;mov    [es:di+35h],dx
27543 00004651 26895535                 MOV     [ES:DI+SF_ENTRY.sf_lstclus],DX       ;AN002; BL
27544                                   ;;;
27545                                   ; 10/02/2024 (PCDOS 7.1 IBMDOS.COM)
27546 00004655 268B552D                 mov     dx,[es:di+2Dh]          ; [ES:DI+SF_ENTRY.sf_chain]
27547                                   ;mov    di,[es:di+SF_ENTRY_sf_fcluster+2] ; first clust (32 bit) hw !?
27548 00004659 26895537                 mov     [es:di+37h],dx
27549                                   ;mov    [es:di+SF_ENTRY.sf_lstclus+2],dx
27550                                   ;;;
27551
27552                                 SKIPRESET:                                       ;AN002; BL
27553 0000465D 07                       POP     ES                                   ;AN002; BL
27554                                 ;
27555                                 ; 10/02/2024
27556                                 %if 0
27557                                     MOV     DX,0FFFFH
27558                                 %else
27559                                     ;;;
27560                                     ; 10/02/2024 (PCDOS 7.1 IBMDOS.COM)
27561 0000465E 31D2                     xor     dx,dx
27562 00004660 4A                       dec     dx
27563 00004661 8916[EA0A]               mov     [CLUSDATA_HW],dx ; 0FFFFh
27564                                   ;;;
27565                                 %endif
27566 00004665 E86E15                   call    RELBLKS
27567                                   ; 16/12/2022
27568                                   ; 20/11/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
27569                                 dskwrt_8:   ; 10/02/2024
27570 00004668 73A5                     jnc     short UPDATE
27571                                 SET_ACC_ERRWJ:
27572                                     ;JC     short SET_ACC_ERRWJ2
27573                                     ;JMP    SHORT UPDATE
27574                                     ; 16/12/2022
27575                                     ;jmp    SET_ACC_ERR_DS ; ds<>ss
27576                                     ; 10/02/2024
27577                                     ; ds=ss
27578 0000466A E97FFB                   jmp     SET_ACC_ERR
27579                                   ; 20/11/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
27580                                   ;JC     short SET_ACC_ERRWJ2
27581                                   ;JMP    SHORT UPDATE
27582
27583                                 KILLFIL:
27584 0000466D 31DB                     XOR     BX,BX
27585 0000466F 06                       PUSH    ES
27586 00004670 C43E[9E05]               LES     DI,[THISSFT]
27587
27588                                 ; 10/02/2024
27589                                 %if 0
27590                                     ;mov    [es:di+19h],bx
27591                                     MOV     [ES:DI+SF_ENTRY.sf_cluspos],BX
27592                                     ;mov    [es:di+35h],bx ; 04/05/2019
27593                                     MOV     [ES:DI+SF_ENTRY.sf_lstclus],BX
27594                                     ;xchg   bx,[es:di+0Bh]
27595                                     XCHG    BX,[ES:DI+SF_ENTRY.sf_firclus]
27596                                 %else
27597                                     ;;;
27598                                     ; 10/02/2024 (PCDOS 7.1 IBMDOS.COM)
27599 00004674 26875D2D                 xchg    bx,[es:di+2Dh] ; [ES:DI+SF_ENTRY.sf_chain+2]
27600                                   ;xchg   bx,[es:di+SF_ENTRY.sf_fcluster+2] ; first clust (32 bit) hw !?
27601 00004678 891E[E80A]               mov     [CLUSTNUM_HW],bx
27602 0000467C 31DB                     xor     bx,bx   ; 0
27603 0000467E 26895D0B                 mov     [es:di+0Bh],bx ; [ES:DI+SF_ENTRY.sf_firclus]
27604                                   ;mov    [es:di+SF_ENTRY.sf_cluspos_hw],bx
27605 00004682 26895D37                 mov     [es:di+37h],bx
27606                                   ;mov    [es:di+SF_ENTRY.sf_lstclus+2],bx
27607                                   ;mov    [es:di+19h],bx
27608 00004686 26895D19                 mov     [es:di+SF_ENTRY.sf_cluspos],bx
27609                                   ;mov    [es:di+35h],bx
27610 0000468A 26895D35                 mov     [es:di+SF_ENTRY.sf_lstclus],bx
27611 0000468E 26875D2B                 xchg    bx,[es:di+2Bh] ; [ES:DI+SF_ENTRY.sf_chain]
27612                                   ;xchg   bx,[es:di+SF_ENTRY.sf_fcluster]      ; first cluster (32 bit) lw !?
27613                                   ;;;
27614                                 %endif
27615
27616 00004692 07                       POP     ES
27617
27618                                   ;;;
27619                                   ; 10/02/2024 (PCDOS 7.1 IBMDOS.COM)
27620 00004693 3B1E[E80A]               cmp     bx,[CLUSTNUM_HW]
27621 00004697 7507                     jnz     short dskwrt_9
27622                                   ;;;
27623
27624 00004699 09DB                     OR      BX,BX
27625                                   ;JZ     short UPDATEJ
27626                                   ; 16/12/2022
27627                                   ;jz     short UPDATE
27628                                   ; 10/12/2024
27629 0000469B 7503                     jnz     short dskwrt_9
27630 0000469D E96FFF                   jmp     UPDATE
27631
27632                                   ; 20/11/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
27633                                   ;jz     short UPDATEJ
27634
27635                                 dskwrt_9:   ; 10/02/2024
27636                                 ;; 10/23/86 FastOpen update
27637 000046A0 06                       PUSH    ES                      ; since first cluster # is 0
```

```
27638 000046A1 55                       PUSH    BP                      ; we must delete the old cache entry
27639 000046A2 50                       PUSH    AX
27640 000046A3 51                       PUSH    CX
27641 000046A4 52                       PUSH    DX
27642 000046A5 C42E[8A05]               LES     BP,[THISDPB]            ; get current DPB
27643                                   ; 15/12/2022
27644 000046A9 268A5600                 mov     dl,[ES:BP] ; mov dl,[es:bp+0]
27645                                   ; 20/11/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
27646                                   ;MOV    DL,[ES:BP+DPB.DRIVE]  ; get current drive
27647 000046AD 89D9                     MOV     CX,BX                   ; first cluster #
27648 000046AF B402                     MOV     AH,2                    ; delete cache entry by drive:firclus
27649 000046B1 E8B3E6                   call    FastOpen_Update                 ; call fastopen
27650 000046B4 5A                       POP     DX
27651 000046B5 59                       POP     CX
27652 000046B6 58                       POP     AX
27653 000046B7 5D                       POP     BP
27654 000046B8 07                       POP     ES
27655                                 ;; 10/23/86 FastOpen update
27656
27657 000046B9 E81415                   call    RELEASE
27658                                   ;JC     short SET_ACC_ERRWJ
27659                                 ;UPDATEJ:
27660                                   ; 20/11/2022
27661                                   ;JMP    short UPDATE ; 10/08/2018
27662                                   ; 10/02/2024
27663 000046BC EBAA                     jmp     short dskwrt_8
27664
27665                                 ;Break   <DskWrtBufPurge -- Disk Write Buffer Purge>
27666                                 ;--------------------------------------------------------------------------
27667                                 ;
27668                                 ; Procedure Name : DskWrtBufPurge
27669                                 ;
27670                                 ; Inputs:
27671                                 ;       CX = # of contiguous sectors to write. (These constitute a block of
27672                                 ;        sectors, also termed an "Extent".)
27673                                 ;       [HIGH_SECTOR]:DX = physical sector # of first sector in extent.
27674                                 ;       ES:BP -> Drive Parameter Block (DPB).
27675                                 ;
27676                                 ; Function:
27677                                 ;       Purge the Buffer Queue and the Secondary Cache of any buffers which
27678                                 ;   are in Extent; they are being over-written.
27679                                 ;
27680                                 ; Outputs:
27681                                 ;       (Same as Input.)
27682                                 ; Uses:
27683                                 ;       All registers except DS,AX,SI,DI preserved.
27684                                 ;       SS override for all global variables.
27685                                 ;--------------------------------------------------------------------------
27686                                 ;M039: Created
27687
27688                                 ;procedure   DskWrtBufPurge,NEAR
27689                                 ;
27690                                 ;ASSUME  DS:NOTHING
27691
27692                                 ; 04/05/2019 - Retro DOS v4.0
27693                                 ; DOSCODE:7C0Eh (MSDOS 6.21, MSDOS.SYS)
27694
27695                                 ; 21/11/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
27696                                 ; DOSCODE:7BD4h (MSDOS 5.0, MSDOS.SYS)
27697
27698                                 ; 10/02/2024 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMDOS.COM)
27699                                 ; DOSCODE:8714h (PCDOS 7.1, IBMDOS.COM)
27700
27701                                 ; NOTE: Secondary (Buffer) Cache is not used in PCDOS 7.1 IBMDOS.COM
27702
27703                                 ; (Win ME IO.SYS - BIOSCODE:BB93h)
27704                                 ; (MSDOS 6.22 IO.SYS - DOSCODE:7C0Eh)
27705
27706                                 DskWrtBufPurge:
27707                                     ;SaveReg <bx,cx>
27708 000046BE 53                       push    bx
27709 000046BF 51                       push    cx
27710
27711 000046C0 368B1E[0706]             mov     bx,[ss:HIGH_SECTOR]   ;BX:DX = Extent start (sector #).
27712 000046C5 89DE                     mov     si,bx
27713 000046C7 01D1                     add     cx,dx
27714 000046C9 83D600                   adc     si,0                    ;SI:CX = Extent end + 1.
27715
27716                                   ; 21/11/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
27717                                   ;;mov   al,[es:bp+0]
27718                                   ;mov    al,[es:bp+DPB.DRIVE]
27719                                   ; 15/12/2022
27720 000046CC 268A4600                 mov     al,[es:bp]
27721
27722                                 ; 10/02/2024 - Retro DOS v5.0
27723                                 ; PCDOS 7.1 IBMDOS.COM
27724                                 %if 0
27725
27726                                 ;   BX:DX = Extent start.
27727                                 ;   SI:CX = Extent end + 1.
27728                                 ;   AL = Drive #
27729
27730                                     cmp     word [ss:SC_CACHE_COUNT],0 ;Secondary cache in-use?
27731                                     je      short nosc              ; -no, jump.
27732
27733                                 ;   If any of the sectors to be written are in the secondary cache (SC),
27734                                 ;   invalidate the entire SC. (This is an optimization; we really only
27735                                 ;   need to invalidate those sectors which intersect, but that's slower.)
27736
27737                                     cmp     al,[ss:CurSC_DRIVE]   ;Same drive?
27738                                     jne     short nosc              ; -no, jump.
27739
27740                                     push    ax
27741                                     mov     ax,[ss:CurSC_SECTOR]
27742                                     mov     di,[ss:CurSC_SECTOR+2]        ;DI:AX = SC start.
27743
27744                                     ;Cmp32  si,cx,di,ax             ;Extent end < SC start?
27745                                     ;jbe    short sc5               ; -yes, jump.
27746
27747                                     cmp     si,di
27748                                     jne     short sc01
27749                                     cmp     cx,ax
27750                                 sc01:
27751                                     jbe     short sc5
27752
27753                                     add     ax,[ss:SC_CACHE_COUNT]
27754                                     adc     di,0                    ;DI:AX = SC end + 1.
27755
27756                                     ;Cmp32  bx,dx,di,ax               ;Extent start > SC end?
27757                                     ;jae    short sc5               ; -yes, jump.
27758
27759                                     cmp     bx,di
27760                                     jne     short sc02
27761                                     cmp     dx,ax
```

```
                                sc02:
                                    jnb     short sc5

                                    mov     word [ss:SC_STATUS],0 ;Extent intersects SC: invalidate SC.
                                sc5:
                                    pop      ax

                                %endif

                                ;   Free any buffered sectors which are in Extent; they are being over-
                                ;   written.

                                nosc:
000046D0 E8E621                     call    GETCURHEAD              ;DS:DI -> first buffer in queue.

                                _bufq:
                                    ;cmpo   al,[di+4]
000046D3 3A4504                     cmp      al,[di+BUFFINFO.buf_ID] ;Same drive?
000046D6 7527                       jne     short bufq5             ; -no, jump.

                                ;       Cmp32   bx,dx,<WORD PTR [di.buf_sector+2]>,<WORD PTR [di.buf_sector]>
                                ;       ja  short bufq5             ;Jump if Extent start > buffer sector.

                                    ;cmp    bx,[di+8]
000046D8 3B5D08                     cmp     bx,[di+BUFFINFO.buf_sector+2]
000046DB 7503                       jne     short bufq04
                                    ;cmp    dx,[di+6]
000046DD 3B5506                     cmp     dx,[di+BUFFINFO.buf_sector]
                                bufq04:
000046E0 771D                       ja      short bufq5

                                ;       Cmp32   si,cx,<WORD PTR [di.buf_sector+2]>,<WORD PTR [di.buf_sector]>
                                ;       jbe short bufq5             ;Jump if Extent end < buffer sector.

                                    ;cmp    si,[di+8]
000046E2 3B7508                     cmp     si,[di+BUFFINFO.buf_sector+2]
000046E5 7503                       jne     short bufq05
                                    ;cmp    cx,[di+6]
000046E7 3B4D06                     cmp     cx,[di+BUFFINFO.buf_sector]
                                bufq05:
000046EA 7613                       jbe     short bufq5

                                ;   Buffer's sector is in Extent, so free it; it is being over-written.

                                    ;test   byte [di+5],40h
000046EC F6450540                   test    byte [di+BUFFINFO.buf_flags],buf_dirty ;Buffer dirty?
000046F0 7403                       jz      short bufq4             ; -no, jump.
000046F2 E8DB24                     call    DEC_DIRTY_COUNT                 ; -yes, decrement dirty count.
                                bufq4:
                                    ;mov    word [di+4],20FFh
000046F5 C74504FF20                 mov      word [di+BUFFINFO.buf_ID],((buf_visit<<8)|0FFh)

000046FA E8CE21                     call    SCANPLACE
000046FD EB02                       jmp      short bufq6
                                bufq5:
000046FF 8B3D                       mov      di,[di]
                                    ;mov    di,[di+BUFFINFO.buf_next]
                                bufq6:
00004701 363B3E[7E11]               cmp     di,[ss:FIRST_BUFF_ADDR]       ;Scanned entire buffer queue?
00004706 75CB                       jne     short _bufq             ; --no, go do next buffer.

                                    ;RestoreReg <cx,bx>
00004708 59                         pop     cx
00004709 5B                         pop     bx
0000470A C3                         retn

                                ;EndProc DskWrtBufPurge

                                ;Break   <DIV32 -- PERFORM 32 BIT DIVIDE>
                                ;-----------------------------------------------------------------------------
                                ;
                                ; Procedure Name : DIV32
                                ;
                                ; Inputs:
                                ;       DX:AX = 32 bit dividend   BX= divisor
                                ; Function:
                                ;       Perform 32 bit division:  DX:AX/BX = CX:AX + DX (rem.)
                                ; Outputs:
                                ;       CX:AX = quotient , DX= remainder
                                ; Uses:
                                ;       All registers except AX,CX,DX preserved.
                                ;-----------------------------------------------------------------------------
                                ;M039: DIV32 optimized for divisor of 512 (common sector size).

                                ; 04/05/2019 - Retro DOS v4.0
                                ; DOSCODE:7C94h (MSDOS 6.21, MSDOS.SYS)

                                ; 21/11/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
                                ; DOSCODE:7C5Ah (MSDOS 5.0, MSDOS.SYS)

                                DIV32:
0000470B 81FB0002                   cmp     bx,512
0000470F 7515                       jne     short div5

00004711 89D1                       mov     cx,dx
00004713 89C2                       mov     dx,ax           ; CX:AX = Dividend
00004715 81E2FF01                   and     dx,(512-1)      ; DX = Remainder
00004719 88E0                       mov     al,ah
0000471B 88CC                       mov     ah,cl
0000471D 88E9                       mov     cl,ch
0000471F 30ED                       xor     ch,ch
00004721 D1E9                       shr     cx,1
00004723 D1D8                       rcr     ax,1
00004725 C3                         retn
                                div5:
00004726 89C1                       mov     cx,ax
00004728 89D0                       mov     ax,dx
0000472A 31D2                       xor     dx,dx
0000472C F7F3                       div     bx              ; 0:AX/BX
0000472E 91                         xchg    cx,ax
0000472F F7F3                       div     bx              ; DX:AX/BX
00004731 C3                         retn

                                ;Break   <SHR32 -- PERFORM 32 BIT SHIFT RIGHT>
                                ;-----------------------------------------------------------------------------
                                ;
                                ; Procedure Name : SHR32
                                ;
                                ; Inputs:
                                ;   DX:AX = 32 bit sector number
                                ; Function:
                                ;       Perform 32 bit shift right
                                ; Outputs:
                                ;   AX = cluster number
```

```
                                ;   ZF = 1 if no error
                                ;      = 0 if error (cluster number > 64k)
                                ; Uses:
                                ;       DX,CX
                                ;-----------------------------------------------------------------------------
                                ; M017      - SHR32 rewritten for better performance
                                ; M039      - Additional optimization

                                ; 04/05/2019 - Retro DOS v4.0
                                ; DOSCODE:7CBBh (MSDOS 6.21, MSDOS.SYS)
                                ; 21/11/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
                                ; DOSCODE:7C81h (MSDOS 5.0, MSDOS.SYS)

                                SHR32:
                                    ;mov    cl,[es:bp+5]
00004732 268A4E05               mov     cl,[ES:BP+DPB.CLUSTER_SHIFT]
00004736 30ED                   xor     ch,ch       ;ZF=1
00004738 E306                   jcxz    norota
                                rotashft2:
0000473A D1EA                       shr     dx,1        ;ZF reflects state of DX.
0000473C D1D8                       rcr     ax,1        ;ZF not affected.
0000473E E2FA                       loop    rotashft2
                                norota:
00004740 C3                         retn

                                ;=============================================================================
                                ; DIR.ASM, MSDOS 6.0, 1991
                                ;=============================================================================
                                ; 27/07/2018 - Retro DOS v3.0
                                ; 19/05/2019 - Retro DOS v4.0

                                ;   TITLE   DIR - Directory and path cracking
                                ;   NAME    Dir

                                ;Break      <FINDENTRY -- LOOK FOR AN ENTRY>
                                ;-----------------------------------------------------------------------------
                                ;
                                ; Procedure Name : FINDENTRY,SEARCH
                                ;
                                ; Inputs:
                                ;   [THISDPB] set
                                ;   [SECCLUSPOS] = 0
                                ;   [DIRSEC] = Starting directory sector number
                                ;   [CLUSNUM] = Next cluster of directory
                                ;   [CLUSFAC] = Sectors/Cluster
                                ;   [NAME1] = Name to look for
                                ; Function:
                                ;   Find file name in disk directory.
                                ;   "?" matches any character.
                                ; Outputs:
                                ;   Carry set if name not found
                                ;   ELSE
                                ;   Zero set if attributes match (always except when creating)
                                ;   AH = Device ID (bit 7 set if not disk)
                                ;   [THISDPB] = Base of drive parameters
                                ;   DS = DOSGROUP
                                ;   ES = DOSGROUP
                                ;   [CURBUF+2]:BX = Pointer into directory buffer
                                ;   [CURBUF+2]:SI = Pointer to First Cluster field in directory entry
                                ;   [CURBUF] has directory record with match
                                ;   [NAME1] has file name
                                ;   [LASTENT] is entry number of the entry
                                ; All other registers destroyed.
                                ;-----------------------------------------------------------------------------

                                ;hkn; called from rename.asm and dir2.asm. DS must be already set up at
                                ;hkn; this point.

                                SEARCH:
                                    ; 21/11/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
                                    ; DOSCODE:7C90h (MSDOS 5.0, MSDOS.SYS)

                                    ; 19/05/2019 - Retro DOS v4.0
                                    ; DOSCODE:7CCAh (MSDOS 6.21, MSDOS.SYS)

                                    ; 27/07/2018 - Retro DOS v3.0
                                    ; IBMDOS.COM (MSDOS 3.3) - Offset 45B3h
                                    ; 15/03/2018 - Retro DOS v2.0

                                    ; 24/01/2024

                                    ; 13/02/2024 - Retro DOS v5.0
                                    ; DOSCODE:8797h (PCDOS 7.1, IBMDOS.COM)

                                    ;entry  FindEntry
                                FINDENTRY:
00004741 E86F06                     call    STARTSRCH
00004744 A0[6B05]                   MOV     AL,[ATTRIB]
                                    ;and    al,9Eh
00004747 24DE                       AND     AL,~attr_ignore             ; Ignore useless bits
                                    ;cmp    al,8
00004749 3C08                       CMP     AL,attr_volume_id       ; Looking for vol ID only ?
0000474B 7503                       JNZ     short NOTVOLSRCH        ; No
0000474D E83B02                     CALL    SETROOTSRCH             ; Yes force search of root
                                NOTVOLSRCH:
00004750 E84C01                     CALL    GETENTRY
                                    ;JNC    short SRCH
                                    ;JMP    SETESRET
                                    ; 24/01/2024
00004753 724F                       jc      short SETESRET

                                    ;entry  Srch
                                SRCH:
                                    ;;;
                                    ; 13/02/2024 (PCDOS 7.1 IBMDOS.COM)
00004755 C706[A50A]0000             mov     word [LNE_COUNT],0    ; reset long name entry count
                                SRCH2:
                                    ;;;

0000475B 1E                         PUSH    DS
0000475C 8E1E[E405]                 MOV     DS,[CURBUF+2]

                                ;   (DS:BX) = directory entry address

00004760 8A27                       mov     ah,[BX]
                                    ;MOV    AH,[BX+dir_entry.dir_name] ; mov ah,[bx+0]
00004762 08E4                       OR      AH,AH                   ; End of directory?
00004764 7461                       JZ      short FREE

                                    ;;;
                                    ; 13/02/2024 (PCDOS 7.1 IBMDOS.COM)
00004766 50                         push    ax
00004767 8A470B                     mov     al,[bx+0Bh]             ; [BX+dir_entry.dir_attr]
0000476A E8A103                     call    check_longname
```

```
28010 0000476D 58                      pop     ax
28011 0000476E 7447                    jz      short NEXTENT2
28012                                  ;;;
28013
28014                                ;hkn; SS override
28015 00004770 363A26[7F05]            CMP     AH,[SS:DELALL]          ; Free entry?
28016 00004775 7450                    JZ      short FREE
28017                                  ;test   byte [bx+0Bh],8
28018 00004777 F6470B08                TEST    byte [BX+dir_entry.dir_attr],attr_volume_id
28019                                                                  ; Volume ID file?
28020 0000477B 7405                    JZ      short CHKFNAM           ; NO
28021
28022                                ;hkn; SS override
28023 0000477D 36FE06[7B05]            INC     BYTE [SS:VOLID]
28024                                CHKFNAM:
28025                                ;   Context ES
28026 00004782 8CD6                    MOV     SI,SS
28027 00004784 8EC6                    MOV     ES,SI
28028 00004786 89DE                    MOV     SI,BX
28029
28030                                ;hkn; NAME1 is in DOSDATA
28031 00004788 BF[4B05]                MOV     DI,NAME1
28032                                ;;;;; 7/29/86
28033
28034                                ;hkn; SS override for NAME1
28035                                  ;CMP    BYTE [SS:NAME1],0E5H  ; special char check
28036                                  ;JNZ    short NO_E5
28037                                  ;MOV    BYTE [SS:NAME1],05H
28038                                  ; 22/09/2023
28039 0000478B 26803DE5                cmp     byte [es:di],0E5h
28040 0000478F 7504                    jnz     short NO_E5
28041 00004791 26C60505                mov     byte [es:di],05h
28042                                NO_E5:
28043                                ;;;;; 7/29/86
28044 00004795 E88100                  CALL    MetaCompare
28045 00004798 7449                    JZ      short FOUND
28046 0000479A 1F                      POP     DS
28047
28048                                  ;entry  NEXTENT
28049                                NEXTENT:
28050 0000479B C42E[8A05]              LES     BP,[THISDPB]
28051 0000479F E88600                  CALL    NEXTENTRY
28052 000047A2 73B1                    JNC     short SRCH
28053                                  ;JMP    SHORT SETESRET
28054                                  ; 24/01/2024
28055                                SETESRET:
28056 000047A4 16                      PUSH    SS
28057 000047A5 07                      POP     ES
28058
28059                                  ;;;
28060                                  ; 13/02/2024 (PCDOS 7.1 IBMDOS.COM)
28061 000047A6 FF36[DA05]              push    word [ENTLAST]
28062 000047AA 8F06[B50A]              pop     word [ENTLAST_PREV]   ; previous ENTLAST
28063 000047AE 7306                    jnc     short SETESRETN
28064 000047B0 C706[A50A]0000          mov     word [LNE_COUNT],0    ; reset long name entry count
28065                                SETESRETN:
28066                                  ;;;
28067
28068 000047B6 C3                      retn
28069
28070                                  ;;;
28071                                  ; 13/02/2024 (PCDOS 7.1 IBMDOS.COM)
28072                                NEXTENT2:
28073 000047B7 1F                      pop     ds
28074 000047B8 FF06[A50A]              inc     word [LNE_COUNT]      ; Long Name entry count
28075                                NEXTENT3:
28076 000047BC C42E[8A05]              les     bp,[THISDPB]
28077 000047C0 E86500                  call    NEXTENTRY
28078 000047C3 7396                    jnc     short SRCH2
28079 000047C5 EBDD                    jmp     short SETESRET
28080                                  ;;;
28081
28082                                FREE:
28083 000047C7 1F                      POP     DS
28084 000047C8 8B0E[4803]              MOV     CX,[LASTENT]
28085 000047CC 3B0E[D805]              CMP     CX,[ENTFREE]
28086 000047D0 7304                    JAE     short TSTALL
28087 000047D2 890E[D805]              MOV     [ENTFREE],CX
28088                                TSTALL:
28089 000047D6 3A26[7F05]              CMP     AH,[DELALL]           ; At end of directory?
28090                                NEXTENTJ:
28091                                  ;je     short NEXTENT         ; No - continue search
28092                                  ;;;
28093                                  ; 13/02/2024
28094 000047DA 74E0                    je      short NEXTENT3        ; No - continue search
28095                                  ;;;
28096 000047DC 890E[DA05]              MOV     [ENTLAST],CX
28097 000047E0 F9                      STC
28098 000047E1 EBC1                    JMP     SHORT SETESRET
28099
28100                                FOUND:
28101                                ; We have a file with a matching name. We must now consider the attributes:
28102                                ; ATTRIB    Action
28103                                ; ------    ------
28104                                ; Volume_ID Is Volume_ID in test?
28105                                ; Otherwise If no create then Is ATTRIB+extra superset of test?
28106                                ;           If create then Is ATTRIB equal to test?
28107
28108 000047E3 8A2C                      MOV     CH,[SI]               ; Attributes of file
28109 000047E5 1F                      POP     DS
28110 000047E6 8A26[6B05]              MOV     AH,[ATTRIB]           ; Attributes of search
28111                                  ;and    ah,9Eh
28112 000047EA 80E4DE                  AND     AH,~attr_ignore
28113                                  ;lea    si,[si+15]
28114 000047ED 8D740F                  LEA     SI,[SI+dir_entry.dir_first-dir_entry.dir_attr]
28115                                                                  ; point to first cluster field
28116                                  ;test   ch,8
28117 000047F0 F6C508                  TEST    CH,attr_volume_id     ; Volume ID file?
28118 000047F3 7409                    JZ      short check_one_volume_id ; Nope check other attributes
28119                                  ;test   ah,8
28120 000047F5 F6C408                  TEST    AH,attr_volume_id     ; Can we find Volume ID?
28121                                  ;JZ     short NEXTENTJ        ; Nope, (not even $FCB_CREATE)
28122                                  ; 16/12/2022
28123 000047F8 74A1                    jz      short NEXTENT ; 19/05/2019
28124                                  ; 21/11/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
28125                                  ;JZ     short NEXTENTJ
28126 000047FA 30E4                    XOR     AH,AH                 ; Set zero flag for $FCB_CREATE
28127 000047FC EB11                    JMP     SHORT RETFF           ; Found Volume ID
28128                                check_one_volume_id:
28129                                  ;CMP    ah,8
28130 000047FE 80FC08                  CMP     AH,attr_volume_id     ; Looking only for Volume ID?
28131                                  ;JZ     short NEXTENTJ        ; Yes, continue search
28132                                  ; 16/12/2022
28133 00004801 7498                    je      short NEXTENT ; 19/05/2019
```

```
                                    ;JZ     short NEXTENTJ
00004803 E8C105                     CALL    MatchAttributes
00004806 7407                       JZ      SHORT RETFF
00004808 F606[7E05]FF               TEST    BYTE [CREATING],-1    ; Pass back mismatch if creating
                                    ; 16/12/2022
                                    ;JZ     short NEXTENTJ        ; Otherwise continue searching
0000480D 748C                       jz      short NEXTENT ; 19/05/2019
                                RETFF:
0000480F C42E[8A05]                 LES     BP,[THISDPB]
                                    ; 21/11/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
                                    ;MOV    AH,[ES:BP+DPB.DRIVE]  ; mov ah,[es:bp+0]
                                    ; 15/12/2022
00004813 268A6600                   MOV     AH,[ES:BP]
                                ;SETESRET:
                                    ;PUSH   SS
                                    ;POP    ES
                                    ;retn
                                    ; 24/01/2024
00004817 EB8B                       jmp     short SETESRET

                                ;--------------------------------------------------------------------------
                                ;
                                ; Procedure Name : MetaCompare
                                ;
                                ; Inputs:
                                ;   DS:SI -> 11 character FCB style name NO '?'
                                ;       Typically this is a directory entry. It MUST be in upper case
                                ;   ES:DI -> 11 character FCB style name with possible '?'
                                ;       Typically this is a FCB or SFT. It MUST be in upper case
                                ; Function:
                                ;   Compare FCB style names allowing for ? match to any char
                                ; Outputs:
                                ;   Zero if match else NZ
                                ; Destroys CX,SI,DI all others preserved
                                ;--------------------------------------------------------------------------

                                    ; 21/11/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
                                    ; DOSCODE:7D3Fh (MSDOS 5.0, MSDOS.SYS)

                                MetaCompare:
00004819 B90B00                     MOV     CX,11
                                WILDCRD:
0000481C F3A6                       REPE    CMPSB
0000481E 7407                       JZ      short MetaRet         ; most of the time we will fail.
                                CHECK_META:
00004820 26807DFF3F                 CMP     BYTE [ES:DI-1],"?"
00004825 74F5                       JZ      short WILDCRD
                                MetaRet:
00004827 C3                         retn                          ; Zero set, Match

                                ;Break      <NEXTENTRY -- STEP THROUGH DIRECTORY>
                                ;--------------------------------------------------------------------------
                                ;
                                ; Procedure Name : NEXTENTRY
                                ;
                                ; Inputs:
                                ;   Same as outputs of GETENTRY, above
                                ; Function:
                                ;   Update BX, and [LASTENT] for next directory entry.
                                ;   Carry set if no more.
                                ;--------------------------------------------------------------------------

                                NEXTENTRY:
                                    ; 21/11/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
                                    ; DOSCODE:7D4Eh (MSDOS 5.0, MSDOS.SYS)

                                    ; 19/05/2019 - Retro DOS v4.0
                                    ; DOSCODE:7D88h (MSDOS 6.21, MSDOS.SYS)

                                    ; 27/07/2018 - Retro DOS v3.0
                                    ; IBMDOS.COM (MSDOS 3.3) - Offset 4671h
                                    ; 15/03/2018 - Retro DOS v2.0

                                    ; 14/02/2024 - Retro DOS v5.0
                                    ; DOSCODE:8884h (PCDOS 7.1, IBMDOS.COM)

00004828 A1[4803]                   MOV     AX,[LASTENT]
0000482B 3B06[DA05]                 CMP     AX,[ENTLAST]
0000482F 741C                       JZ      short NONE
00004831 40                         INC     AX
                                    ;ADD    BX,32
00004832 8D5F20                     LEA     BX,[BX+32]
00004835 39D3                       CMP     BX,DX
                                    ; 21/11/2022 - MSDOS 5.0 MSDOS.SYS (DOSCODE:7D5Dh)
                                    ;JB     short HAVIT ; MSDOS 6.0 src (dir.asm)
                                    ; 16/12/2022
00004837 7540                       jne     short HAVIT ; MSDOS 6.21 (DOSCODE:7D97h)
                                    ;;;
                                    ; 14/02/2024 (PCDOS 7.1 IBMDOS.COM)
00004839 833E[C205]00               cmp     word [DIRSTART],0
0000483E 750F                       jnz     short nextentry_cont
00004840 833E[DC0A]00               cmp     word [DIRSTART_HW],0
00004845 7508                       jnz     short nextentry_cont
                                    ;cmp    ax,[es:bp+9]
00004847 263B4609                   cmp     ax,[es:bp+DPB.ROOT_ENTRIES] ; Number of root dir entries
                                    ;jnb    short NONE
                                    ;jmp    short GETENT
0000484B 7255                       jb      short GETENT
                                NONE:               ; 14/02/2024
0000484D F9                         stc
0000484E C3                         retn

                                nextentry_cont:
                                    ;;;

0000484F 8A1E[7305]                 MOV     BL,[SECCLUSPOS]
00004853 FEC3                       INC     BL
00004855 3A1E[7705]                 CMP     BL,[CLUSFAC]
00004859 7223                       JB      short SAMECLUS
                                    ;;;
                                    ; 14/02/2024 (PCDOS 7.1 IBMDOS.COM)
0000485B 8B1E[E00A]                 mov     bx,[NXTCLUSNUM_HW]
0000485F 891E[E80A]                 mov     [CLUSTNUM_HW],bx
                                    ;;;
00004863 8B1E[DC05]                 MOV     BX,[NXTCLUSNUM]
00004867 E84E1A                     call    IsEOF
0000486A 73E1                       JAE     short NONE
                                    ;;;
                                    ; 14/02/2024
0000486C 833E[E80A]00               cmp     word [CLUSTNUM_HW],0
00004871 752F                       jnz     short GETENT
                                    ;;;
                                    ; 23/07/2019
```

```
00004873 83FB02                     CMP     BX,2
                                    ;JB     short NONE
                                    ;JMP    short GETENT
                                    ; 16/12/2022
00004876 732A                       jnb     short GETENT
                                    ; 21/11/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
                                    ;JB     short NONE
                                    ;JMP    short GETENT
                                    ; 14/02/2024 - Retro DOS v5.0
                                ;NONE:
                                    ;STC
00004878 C3                         retn
                                HAVIT:
00004879 A3[4803]                   MOV     [LASTENT],AX
0000487C F8                         CLC
                                nextentry_retn:
0000487D C3                         retn

                                SAMECLUS:
0000487E 881E[7305]                 MOV     [SECCLUSPOS],BL
00004882 A3[4803]                   MOV     [LASTENT],AX
00004885 1E                         PUSH    DS
00004886 C53E[E205]                 LDS     DI,[CURBUF]
                                    ; 19/05/2019
                                    ; MSDOS 6.0
                                    ;;mov   dx,[di+8]
                                    ; 23/09/2023
                                    ;MOV    DX,[DI+BUFFINFO.buf_sector+2] ;AN000; >32mb
                                ;hkn; SS override
                                    ;MOV    [SS:HIGH_SECTOR],DX           ;AN000; >32mb

                                ; 14/02/2024
                                %if 0
                                    ; 23/09/2023
                                    mov     si,[di+BUFFINFO.buf_sector+2]

                                    ;mov    dx,[di+6]
                                    MOV     DX,[DI+BUFFINFO.buf_sector]   ;AN000; >32mb

                                    ;inc    dx ; MSDOS 3.3
                                    ; MSDOS 6.0
                                    ;ADD    DX,1                          ;AN000; >32mb
                                    ;ADC    word [SS:HIGH_SECTOR],0       ;AN000; >32mb
                                    ; 23/09/2023
                                    inc     dx
                                    jnz     short nextexntry_fc
                                    inc     si
                                    ;inc    word [SS:HIGH_SECTOR]
                                nextexntry_fc:
                                    ; 23/09/2023
                                    mov     [SS:HIGH_SECTOR],si
                                    ; MSDOS 3.3 & MSDOS 6.0
                                    POP     DS
                                %else
                                    ; 14/02/2024 - Retro DOS v5.0
0000488A C55506                     lds     dx,[di+BUFFINFO.buf_sector]
0000488D 8CDE                       mov     si,ds
0000488F 1F                         pop     ds
00004890 42                         inc     dx
00004891 7501                       jnz     short nextexntry_fc
00004893 46                         inc     si
                                nextexntry_fc:
00004894 8936[0706]                 mov     [HIGH_SECTOR],si
                                %endif

00004898 E8DEF6                     call    FIRSTCLUSTER
0000489B 31DB                       XOR     BX,BX
0000489D EB21                       JMP     short SETENTRY

                                ;--------------------------------------------------------------------------
                                ;
                                ; Procedure Name : GETENTRY
                                ;
                                ; Inputs:
                                ;   [LASTENT] has directory entry
                                ;   ES:BP points to drive parameters
                                ;   [DIRSEC],[CLUSNUM],[CLUSFAC],[ENTLAST] set for DIR involved
                                ; Function:
                                ;   Locates directory entry in preparation for search
                                ;   GETENT provides entry for passing desired entry in AX
                                ; Outputs:
                                ;   [CURBUF+2]:BX = Pointer to next directory entry in CURBUF
                                ;   [CURBUF+2]:DX = Pointer to first byte after end of CURBUF
                                ;   [LASTENT] = New directory entry number
                                ;   [NXTCLUSNUM],[SECCLUSPOS] set via DIRREAD
                                ;   Carry set if error (currently user FAILed to I 24)
                                ;--------------------------------------------------------------------------

                                    ; 21/11/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
                                GETENTRY:
                                    ; 27/07/2018 - Retro DOS v3.0
0000489F A1[4803]                   MOV     AX,[LASTENT]

                                    ;entry  GETENT
                                GETENT:
000048A2 A3[4803]                   MOV     [LASTENT],AX
                                ;
                                ; Convert the entry number in AX into a byte offset from the beginning of the
                                ; directory.
                                ;
000048A5 B105                       mov     cl,5                    ; shift left by 5 = mult by 32
000048A7 D3C0                       rol     ax,cl                   ; keep hight order bits
000048A9 89C2                       mov     dx,ax
                                    ; 19/05/2019 - Retro DOS v4.0
                                    ;and    ax,0FFE0h
                                    ; 21/11/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
                                    ;and    ax,~(32-1)              ; mask off high order bits
                                    ; 16/12/2022
000048AB 24E0                       and     al,0E0h ; ~31
000048AD 83E21F                     and     dx,1Fh
                                    ;and    dx,32-1                 ; mask off low order bits
                                ;
                                ; DX:AX contain the byte offset of the required directory entry from the
                                ; beginning of the directory. Convert this to a sector number. Round the
                                ; sector size down to a multiple of 32.
                                ;
                                    ;mov    bx,[es:bp+2]
000048B0 268B5E02                   MOV     BX,[ES:BP+DPB.SECTOR_SIZE]
000048B4 80E3E0                     and     bl,0E0h
                                    ;AND    BL,255-31               ; Must be multiple of 32
000048B7 F7F3                       DIV     BX
                                    ; 14/02/2024
                                    ;MOV    BX,DX                   ; Position within sector
                                                            ; NOTE: This BX value is not used in DIRREAD
```

```
                                                        ; Erdogan Tan - 14/02/2024
                                    ;PUSH   BX
000048B9 52                         push    dx
                                    ;
000048BA E832F6                     call    DIRREAD
000048BD 5B                         POP     BX
                                    ;retc
000048BE 72BD                       jc      short nextentry_retn
                                SETENTRY:
000048C0 8B16[E205]                 MOV     DX,[CURBUF]
                                    ;;;add  dx,16 ; MSDOS 3.3
                                    ;;add   dx,20 ; MSDOS 6.0
                                    ;add    dx,24 ; PCDOS 7.1 ; 14/02/2024
000048C4 83C218                     ADD     DX,BUFINSIZ
000048C7 01D3                       ADD     BX,DX
                                    ;add    dx,[es:bp+2]
000048C9 26035602                   ADD     DX,[ES:BP+DPB.SECTOR_SIZE]  ; Always clears carry
                                    ; 29/12/2022
                                    ; MSDOS 6.21 MSDOS.SYS contains a 'CLC' here, at DOSCODE:7E15h
                                    ; 27/06/2024
                                    ; PCDOS 7.1 IBMDOS.COM contains 'CLC' here, at DOSCODE:8931h
000048CD F8                         clc
000048CE C3                         retn

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

                                ; 23/01/2024 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMDOS.COM)
                                ; PCDOS 7.1 IBMDOS.COM - DOSCODE:8933h

                                sft_fcb_table:
000048CF 00<rep 14h>                times 20 db 0
                                sftfcb.cluster:
000048E3 00000000                   dd      0
                                sftfcb.direntry:
000048E7 0000                       dw      0
                                sftfcb_entry_size equ $ - sftfcb.cluster ; = 6

000048E9 00<rep 72h>                times 114 db 0 ; 6*20 = 120 ; entry size = 6 bytes

                                SRCH_CLUSTER:
0000495B 0000                       dw      0
                                SRCH_CLUSTER_HW:
0000495D 0000                       dw      0

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

                                ;Break      <SETDIRSRCH SETROOTSRCH -- Set Search environments>
                                ;--------------------------------------------------------------------------
                                ;
                                ; Procedure Name : SETDIRSRCH,SETROOTSRCH
                                ;
                                ; Inputs:
                                ;   BX cluster number of start of directory
                                ;   ES:BP Points to DPB
                                ;   DI next cluster number from fastopen extended info. DOS 3.3 only
                                ; Function:
                                ;   Set up a directory search
                                ; Outputs:
                                ;   [DIRSTART] = BX
                                ;   [CLUSFAC],[CLUSNUM],[SECCLUSPOS],[DIRSEC] set
                                ;   Carry set if error (currently user FAILed to I 24)
                                ; destroys AX,DX,BX
                                ;--------------------------------------------------------------------------

                                ; 23/01/2024 - Retro DOS v5.0
                                %if 0
                                    ; 21/11/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
                                SETDIRSRCH:
                                    OR      BX,BX
                                    JZ      short SETROOTSRCH
                                    MOV     [DIRSTART],BX
                                    ;mov    al,[es:bp+4]
                                    MOV     AL,[ES:BP+DPB.CLUSTER_MASK]
                                    INC     AL
                                    MOV     [CLUSFAC],AL

                                ; DOS 3.3 for FastOPen    F.C. 6/12/86
                                    ;SAVE   <SI>
                                    push    si
                                    ;test   byte [FastOpenFlg],2
                                    TEST    byte [FastOpenFlg],Lookup_Success
                                    JNZ     short UNP_OK

                                ; DOS 3.3 for FastOPen    F.C. 6/12/86
                                    ;invoke UNPACK
                                    call    UNPACK
                                    JNC     short UNP_OK
                                    ;RESTORE <SI>
                                    pop     si
                                    ;return
                                    retn

                                UNP_OK:
                                    MOV     [CLUSNUM],DI
                                    MOV     DX,BX
                                    XOR     BL,BL
                                    MOV     [SECCLUSPOS],BL
                                    ;invoke FIGREC
                                    call    FIGREC
                                    ;RESTORE <SI>
                                    pop     si

                                    ; 19/05/2019 - Retro DOS v4.0

                                    ; MSDOS 6.0
                                    ;PUSH   DX                          ;AN000; >32mb
                                    ;MOV    DX,[HIGH_SECTOR]            ;AN000; >32mb
                                    ;MOV    [DIRSEC+2],DX               ;AN000; >32mb
                                    ;POP    DX                          ;AN000; >32mb

                                    ; 21/11/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
                                    ;push   dx
                                    ;mov    dx,[HIGH_SECTOR]
                                    ;mov    [DIRSEC+2],dx
                                    ;pop    dx
                                    ;MOV    [DIRSEC],dx
                                    ; 16/12/2022
                                    mov     ax,[HIGH_SECTOR]
                                    mov     [DIRSEC+2],AX
                                    MOV     [DIRSEC],DX

                                    ; 16/12/2022
                                    ; cf=0 (at the return of FIGREC)
                                    ;CLC
```

```
                                    retn
                                    ;entry  SETROOTSRCH
                                SETROOTSRCH:
                                    XOR     AX,AX
                                    MOV     [DIRSTART],AX
                                    ; 22/09/2023
                                    mov     [DIRSEC+2],ax ; 0
                                    MOV     [SECCLUSPOS],AL
                                    DEC     AX
                                    MOV     [CLUSNUM],AX
                                    ;mov    ax,[es:bp+0Bh]
                                    MOV     AX,[ES:BP+DPB.FIRST_SECTOR]
                                    ; 19/05/2019
                                    ;;mov   dx,[es:bp+10h] ; MSDOS 3.3
                                    ;mov    dx,[es:bp+11h] ; MSDOS 6.0
                                    MOV     DX,[ES:BP+DPB.DIR_SECTOR]
                                    SUB     AX,DX
                                    MOV     [CLUSFAC],AL
                                    MOV     [DIRSEC],DX                 ;F.C. >32mb
                                    ; 22/09/2023
                                    ; MSDOS 6.0
                                    ;MOV    WORD [DIRSEC+2],0           ;F.C. >32mb
                                    CLC
                                    retn
                                %else
                                    ;;;
                                    ; 23/01/2024 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMDOS.COM)
                                    ; PCDOS 7.1 IBMDOS.COM - DOSCODE:89C3h
                                    ;;;
                                SETDIRSRCH:
0000495F A1[EE0A]                   mov     ax,[ROOTCLUST_HW]
                                    ;cmp    word [ROOTCLUST_HW],0
00004962 21C0                       and     ax,ax
00004964 7504                       jnz     short SETDIRSRCH_FAT32

00004966 09DB                       or      bx,bx
                                    ;jz     short SETROOTSRCH
00004968 7423                       jz      short SETROOTSRCH2 ; ax = 0
                                SETDIRSRCH_FAT32:
                                    ;mov    ax,[ROOTCLUST_HW]
0000496A A3[DC0A]                   mov     [DIRSTART_HW],ax
0000496D A3[E80A]                   mov     [CLUSTNUM_HW],ax

00004970 891E[C205]                 mov     [DIRSTART],bx
                                    ;mov    al,[es:bp+4]
00004974 268A4604                   mov     al,[es:bp+DPB.CLUSTER_MASK]
                                    ;inc    al
00004978 40                         inc     ax
00004979 A2[7705]                   mov     [CLUSFAC],al

0000497C 56                         push    si
                                    ;test   byte [FastOpenFlg],2
0000497D F606[4611]02               test    byte [FastOpenFlg],Lookup_Success
00004982 7534                       jnz     short UNP_OK

00004984 E85A19                     call    UNPACK
00004987 732F                       jnc     short UNP_OK

00004989 5E                         pop     si
0000498A C3                         retn

                                SETROOTSRCH:
0000498B 31C0                       xor     ax,ax ; 0
                                SETROOTSRCH2:
0000498D A3[EE0A]                   mov     [ROOTCLUST_HW],ax ;0
                                    ;cmp    word [es:bp+0Fh],0
00004990 2639460F                   cmp     [es:bp+DPB.FAT_SIZE],ax ; 0
00004994 7556                       jnz     short SETROOTSRCH_FAT ; not FAT32
                                SETROOTSRCH_FAT32:
                                    ;;mov   bx,[es:bp+37h]
                                    ;mov    bx,[es:bp+DPB.ROOT_CLUSTER+2]
                                    ;mov    [ROOTCLUST_HW],bx
                                    ;;cmp   bx,[es:bp+2Fh]
                                    ;cmp    bx,[es:bp+DPB.LAST_CLUSTER+2]
                                    ; 27/06/2024 - Retro DOS v5.0
00004996 268B4637                   mov     ax,[es:bp+DPB.ROOT_CLUSTER+2]
0000499A A3[EE0A]                   mov     [ROOTCLUST_HW],ax
0000499D 263B462F                   cmp     ax,[es:bp+DPB.LAST_CLUSTER+2]
                                    ;mov    bx,[es:bp+35h]
000049A1 268B5E35                   mov     bx,[es:bp+DPB.ROOT_CLUSTER]
000049A5 7504                       jne     short sdsrch_fat32_1
                                    ;cmp    bx,[es:bp+2Dh]
000049A7 263B5E2D                   cmp     bx,[es:bp+DPB.LAST_CLUSTER]
                                sdsrch_fat32_1:
000049AB 7709                       ja      short sdsrch_fat32_3 ; **
                                    ; 27/06/2024
000049AD 21C0                       and     ax,ax ; [ROOTCLUST_HW]
                                    ;cmp    word [ROOTCLUST_HW],0
                                    ;jnz    short sdsrtch_fat32_2
000049AF 75B9                       jnz     short SETDIRSRCH_FAT32
000049B1 83FB02                     cmp     bx,2
                                    ;jb     short sdsrch_fat32_3
                                sdsrch_fat32_2:
                                    ;jmp    SETDIRSRCH_FAT32
000049B4 73B4                       jnb     short SETDIRSRCH_FAT32

                                sdsrch_fat32_3:
000049B6 F9                         stc     ; **
                                    ;jmp    short setdirsrch_retn
000049B7 C3                         retn

                                UNP_OK:
                                    ; CCONTENT_HW:DI = Contents of FAT for given cluster
                                    ;                       (from UNPACK)
000049B8 893E[BC05]                 mov     [CLUSNUM],di
000049BC 8B16[EC0A]                 mov     dx,[CCONTENT_HW]
000049C0 8916[DE0A]                 mov     [CLUSNUM_HW],dx
000049C4 2E891E[5B49]               mov     [cs:SRCH_CLUSTER],bx ; Directory start cluster number
                                                                 ; for searching/locating (directory entry)
000049C9 8B16[E80A]                 mov     dx,[CLUSTNUM_HW]
000049CD 2E8916[5D49]               mov     [cs:SRCH_CLUSTER_HW],dx

000049D2 89DA                       mov     dx,bx
000049D4 30DB                       xor     bl,bl
000049D6 881E[7305]                 mov     [SECCLUSPOS],bl

                                    ;CLUSTNUM_HW:DX = Physical cluster number (to FIGREC)

000049DA E8BA0F                     call    FIGREC

000049DD 8B36[0706]                 mov     si,[HIGH_SECTOR]
000049E1 8936[C005]                 mov     [DIRSEC+2],si
000049E5 5E                         pop     si
```

```
                                SETROOTSRCH3:
000049E6 8916[BE05]                 mov     [DIRSEC],dx ; *

                                    ;pop    si
000049EA F8                         clc     ; (this may not be needed,
                                            ; FIGREC clears cf except an abnormal sector calc overflow)
000049EB C3                         retn

                                SETROOTSRCH_FAT:
                                    ;xor    ax,ax
000049EC A3[C005]                   mov     [DIRSEC+2],ax ; 0
000049EF A3[C205]                   mov     [DIRSTART],ax ; 0
000049F2 A3[DC0A]                   mov     [DIRSTART_HW],ax ; 0
000049F5 2EA3[5D49]                 mov     [cs:SRCH_CLUSTER_HW],ax ; 0
000049F9 40                         inc     ax              ; search start dir cluster num = 1
                                                            ; (root directory)
000049FA 2EA3[5B49]                 mov     [cs:SRCH_CLUSTER],ax ; 1 ; FAT root directory (<2)
000049FE 48                         dec     ax ; 0
000049FF A2[7305]                   mov     [SECCLUSPOS],al
00004A02 48                         dec     ax ; -1
00004A03 A3[BC05]                   mov     [CLUSNUM],ax ; 0FFFFh
00004A06 A3[DE0A]                   mov     [CLUSNUM_HW],ax ; 0FFFFh

                                    ;mov    ax,[es:bp+0Bh]
00004A09 268B460B                   mov     ax,[es:bp+DPB.FIRST_SECTOR]
                                    ;mov    dx,[es:bp+11h]
00004A0D 268B5611                   mov     dx,[es:bp+DPB.DIR_SECTOR]
00004A11 29D0                       sub     ax,dx
00004A13 A2[7705]                   mov     [CLUSFAC],al
                                    ;mov    [DIRSEC],dx ; *
                                    ;clc
                                ;setdirsrch_retn:
                                    ;retn
00004A16 EBCE                       jmp     short SETROOTSRCH3
                                    ;;;
                                %endif


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

                                ; 23/01/2024 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMDOS.COM)
                                ; PCDOS 7.1 IBMDOS.COM - DOSCODE:8A90h

                                ; set SFT number and entry in the new internal table (only for FCB calls)

                                set_sftfcb_entry:
00004A18 50                         push    ax
00004A19 51                         push    cx
00004A1A 52                         push    dx
00004A1B 53                         push    bx

                                    ;push   bp
                                    ;push   si
                                    ;push   di              ; ES:DI = SFT entry

00004A1C E84B00                     call    find_sft_entry_number
                                                            ; ax = SFT entry index number
                                                            ;      of the last SFT entry
00004A1F 31DB                       xor     bx,bx
00004A21 B91400                     mov     cx,20           ; find empty entry (slot) in the table
                                    ; Retro DOS v5.0
00004A24 31D2                       xor     dx,dx ; *
                                set_sftfcbe_1:
                                    ;;cmp   cs:(sftfcb_cluster+2)[bx],0
                                    ;cmp    word [cs:bx+sftfcb.cluster+2],0
                                                            ; directory (search) starting cluster, hw
00004A26 2E3997[E548]               cmp     [cs:bx+sftfcb.cluster+2],dx ; 0
                                    ;jnz    short set_sftfcbe_2
                                    ; 27/06/2024
00004A2B 7533                       jnz     short set_sftfcbe_3

                                    ;;cmp   cs:sftfcb_cluster[bx],
                                    ;cmp    word [cs:bx+sftfcb.cluster],0
                                                            ; directory (search) starting cluster, lw
00004A2D 2E3997[E348]               cmp     [cs:bx+sftfcb.cluster],dx ; 0
                                set_sftfcbe_2:
00004A32 752C                       jnz     short set_sftfcbe_3
                                                            ; not empty (sfcb table) entry
                                    ; Retro DOS v5.0
                                    ;push   cx

00004A34 2E8B0E[5B49]               mov     cx,[cs:SRCH_CLUSTER]
                                                            ; search start dir cluster number
                                    ;mov    cs:sftfcb_cluster[bx],cx
00004A39 2E898F[E348]               mov     [cs:bx+sftfcb.cluster],cx
                                                            ; directory start cluster
00004A3E 2E8B0E[5D49]               mov     cx,[cs:SRCH_CLUSTER_HW]

                                    ; PCDOS 7.1 BUG! (This would be '[cs:bx:sftfcb.cluster+2],cx')
                                    ; Erdogan Tan - 23/01/2024
                                    ;mov    cs:sftfcb_cluster[bx],cx
                                    ;mov    [cs:bx:sftfcb.cluster],cx ; PCDOS 7.1 IBMDOS.COM - DOSCODE:8ABFh
                                    ; Retro DOS v5.0
00004A43 2E898F[E548]               mov     [cs:bx+sftfcb.cluster+2],cx

                                    ;pop    cx

                                    ;mov    dx,[ss:LASTENT]         ; LAST (found) entry in the directory
                                    ;;mov   cs:sftfcb_direntry[bx],dx
                                    ;mov    [cs:bx+sftfcb.direntry],dx
                                                            ; directory entry number
00004A48 368B0E[4803]               mov     cx,[ss:LASTENT]
00004A4D 2E898F[E748]               mov     [cs:bx+sftfcb.direntry],cx

00004A52 93                         xchg    ax,bx
                                    ;xor    dx,dx ; *
                                    ; dx = 0
00004A53 B90600                     mov     cx,6            ; sftfcb table has 6 byte entries
00004A56 F7F1                       div     cx
00004A58 93                         xchg    ax,bx
00004A59 2E8887[CF48]               mov     [cs:bx+sft_fcb_table],al
                                                            ; put SFT entry number in SFT-FCB
                                                            ; table (offset is FCB index number 0 to 19)
00004A5E EB05                       jmp     short set_setfcbe_4
                                set_sftfcbe_3:
                                    ;add    bx,6
00004A60 83C306                     add     bx,sftfcb_entry_size ; 6
00004A63 E2C1                       loop    set_sftfcbe_1

                                set_setfcbe_4:
                                    ;pop    di
                                    ;pop    si
                                    ;pop    bp

00004A65 5B                         pop     bx
```

```
28754 00004A66 5A                       pop     dx
28755 00004A67 59                       pop     cx
28756 00004A68 58                       pop     ax
28757 00004A69 C3                       retn
28758
28759                                ;--------------------------------------------------------------------------
28760
28761                                ; 24/01/2024 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMDOS.COM)
28762                                ; PCDOS 7.1 IBMDOS.COM - DOSCODE:8AECh
28763
28764                                    ; 14/02/2024
28765                                find_sft_entry_number:
28766 00004A6A 06                       push    es              ; es:di = SFT entry
28767 00004A6B 31C9                     xor     cx,cx
28768 00004A6D 8CC2                     mov     dx,es
28769 00004A6F 2E8E06[0700]             mov     es,[cs:DosDSeg]
28770 00004A74 26C41E[2A00]             les     bx,[es:SFT_ADDR] ; address of the first SFT
28771                                f_sfte_1:
28772 00004A79 8CC0                     mov     ax,es
28773 00004A7B 39D0                     cmp     ax,dx           ; same SFT segment ?
28774 00004A7D 7512                     jne     short f_sfte_2 ; no
28775 00004A7F 89F8                     mov     ax,di
28776 00004A81 29D8                     sub     ax,bx           ; ax = entry offset
28777                                   ;sub    ax,6            ; ax = offset from start of the SFT table
28778 00004A83 83E806                   sub     ax,SFT.SFTable
28779                                   ;mov    bx,59
28780 00004A86 BB3B00                   mov     bx,SF_ENTRY.size ; (SFT entry size)
28781 00004A89 31D2                     xor     dx,dx
28782 00004A8B F7F3                     div     bx              ; ax = SFT entry index in the table
28783 00004A8D 01C8                     add     ax,cx           ; ax = SFT index number (for requested SFT entry)
28784 00004A8F 07                       pop     es
28785 00004A90 C3                       retn
28786                                f_sfte_2:
28787 00004A91 26034F04                 add     cx,[es:bx+4]   ; SFT.SFCount ; number of entries in the table
28788 00004A95 26C41F                   les     bx,[es:bx]     ; SFT.SFLink
28789 00004A98 83FBFF                   cmp     bx,0FFFFh ; -1 ; the last SFT
28790 00004A9B 75DC                     jnz     short f_sfte_1
28791 00004A9D F9                       stc                     ; (not found)
28792 00004A9E 07                       pop     es
28793 00004A9F C3                       retn
28794
28795                                ;--------------------------------------------------------------------------
28796
28797                                ; 24/01/2024 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMDOS.COM)
28798                                ; PCDOS 7.1 IBMDOS.COM - DOSCODE:8B22h
28799
28800                                    ; 14/02/2024
28801                                int_2Fh_1230h:              ; Windows95 - FIND SFT ENTRY IN INTERNAL FILE TABLES
28802 00004AA0 E8C7FF                   call    find_sft_entry_number
28803 00004AA3 7242                     jc      short find_sfte_i_error
28804 00004AA5 06                       push    es      ; ax = SFT enty index number (of the requested SFT entry)
28805 00004AA6 57                       push    di
28806 00004AA7 0E                       push    cs
28807 00004AA8 07                       pop     es
28808 00004AA9 B91400                   mov     cx,20           ; statically allocated with 20 entries,
28809                                                           ; and used only for FCB calls
28810 00004AAC BF[CF48]                 mov     di,sft_fcb_table ; new file system (internal) table
28811                                                           ;  only for fcb calls
28812                                scan_next_sftfcb:
28813 00004AAF F2AE                     repne scasb
28814 00004AB1 F9                       stc
28815 00004AB2 7531                     jnz     short sfte_i_notfound ; not found (cf=1)
28816 00004AB4 8D5DFF                   lea     bx,[di-1]       ; offset of the entry in the table
28817 00004AB7 81EB[CF48]               sub     bx,sft_fcb_table ; index into new file system table
28818 00004ABB 89DA                     mov     dx,bx
28819 00004ABD 01DB                     add     bx,bx           ; 2*bx
28820 00004ABF 01D3                     add     bx,dx           ; 3*bx
28821 00004AC1 01DB                     add     bx,bx           ; bx = 6*bx  ; entry size = 6 bytes
28822 00004AC3 268BB7[E548]             mov     si,[es:bx+sftfcb.cluster+2] ; dir start cluster number, hw
28823 00004AC8 268B97[E748]             mov     dx,[es:bx+sftfcb.direntry] ; directory entry number
28824 00004ACD 51                       push    cx
28825 00004ACE 268B8F[E348]             mov     cx,[es:bx+sftfcb.cluster] ; dir start cluster number, lw
28826 00004AD3 09C9                     or      cx,cx
28827 00004AD5 750C                     jnz     short sfte_i_found
28828 00004AD7 09F6                     or      si,si
28829 00004AD9 7508                     jnz     short sfte_i_found
28830 00004ADB 59                       pop     cx
28831 00004ADC 09C9                     or      cx,cx
28832 00004ADE 75CF                     jnz     short scan_next_sftfcb
28833 00004AE0 F9                       stc                     ; not found (cf=1)
28834 00004AE1 EB02                     jmp     short sfte_i_notfound
28835
28836                                sfte_i_found:
28837 00004AE3 58                       pop     ax              ; clear stack
28838 00004AE4 F8                       clc                     ; found (cf=0)
28839                                sfte_i_notfound:
28840 00004AE5 5F                       pop     di
28841 00004AE6 07                       pop     es
28842                                find_sfte_i_error:
28843 00004AE7 B80000                   mov     ax,0
28844 00004AEA C3                       retn
28845
28846                                ;--------------------------------------------------------------------------
28847
28848                                ; 24/01/2024 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMDOS.COM)
28849                                ; PCDOS 7.1 IBMDOS.COM - DOSCODE:8B6Dh
28850
28851                                    ; 14/02/2024
28852                                SFT_FREE:
28853 00004AEB 50                       push    ax
28854 00004AEC 51                       push    cx
28855 00004AED 52                       push    dx
28856 00004AEE 53                       push    bx
28857                                   ;push   bp      ; 14/02/2024 - Retro DOS v5.0
28858 00004AEF 56                       push    si
28859                                   ;push   di
28860 00004AF0 26C7050000               mov     word [es:di],0 ; [ES:DI+SF_ENTRY.sf_ref_Count]
28861 00004AF5 E8A8FF                   call    int_2Fh_1230h
28862 00004AF8 720E                     jc      short sftf_1
28863 00004AFA 2EC787[E548]0000         mov     word [cs:bx+sftfcb.cluster+2],0 ; clear directory start cluster
28864                                                           ;  in the new (sftfcb) table
28865 00004B01 2EC787[E348]0000         mov     word [cs:bx+sftfcb.cluster],0 ; ((empty entry))
28866                                sftf_1:
28867                                   ;pop    di
28868 00004B08 5E                       pop     si
28869                                   ;pop    bp
28870 00004B09 5B                       pop     bx
28871 00004B0A 5A                       pop     dx
28872 00004B0B 59                       pop     cx
28873 00004B0C 58                       pop     ax
28874 00004B0D C3                       retn
28875
28876                                ; =============== S U B R O U T I N E =======================================
28877
```

```
                                ; 13/02/2024 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMDOS.COM)
                                ; PCDOS 7.1 IBMDOS.COM - DOSCODE:8B94h

                                check_longname:
00004B0E 240F                       and     al,0Fh
00004B10 3C0F                       cmp     al,0Fh
00004B12 C3                         retn

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

                                ;==========================================================================
                                ; DIR2.ASM, MSDOS 6.0, 1991
                                ;==========================================================================
                                ; 27/07/2018 - Retro DOS v3.0
                                ; 19/05/2019 - Retro DOS v4.0
                                ; 14/02/2024 - Retro DOS v5.0

                                ;   TITLE   DIR2 - Directory and path cracking
                                ;   NAME    Dir2

                                ;Break      <GETPATH -- PARSE A WFP>
                                ;--------------------------------------------------------------------------
                                ;
                                ; Procedure Name : GETPATH
                                ;
                                ; Inputs:
                                ;   [WFP_START] Points to WFP string ("d:\" must be first 3 chars, NUL
                                ;           terminated; d:/ (note forward slash) indicates a real device).
                                ;   [CURR_DIR_END] Points to end of Current dir part of string
                                ;           ( = -1 if current dir not involved, else
                                ;            Points to first char after last "/" of current dir part)
                                ;   [THISCDS] Points to CDS being used
                                ;   [SATTRIB] Is attribute of search, determines what files can be found
                                ;   [NoSetDir] set
                                ;   [THISDPB] set to DPB if disk otherwise garbage.
                                ; Function:
                                ;   Crack the path
                                ; Outputs:
                                ;   Sets EXTERR_LOCUS = errLOC_Disk if disk file
                                ;   Sets EXTERR_LOCUS = errLOC_Unk if char device
                                ;   ID1 field of [THISCDS] updated appropriately
                                ;   [ATTRIB] = [SATTRIB]
                                ;   ES:BP Points to DPB
                                ;   Carry set if bad path
                                ;      SI Points to path element causing failure
                                ;      Zero set
                                ;         [DIRSTART],[DIRSEC],[CLUSNUM], and [CLUSFAC] are set up to
                                ;         start a search on the last directory
                                ;         CL is zero if there is a bad name in the path
                                ;         CL is non-zero if the name was simply not found
                                ;            [ENTFREE] may have free spot in directory
                                ;            [NAME1] is the name.
                                ;            CL = 81H if '*'s or '?' in NAME1, 80H otherwise
                                ;      Zero reset
                                ;         File in middle of path or bad name in path or attribute mismatch
                                ;           or path too long or malformed path
                                ;   ELSE
                                ;      [CurBuf] = -1 if root directory
                                ;      [CURBUF] contains directory record with match
                                ;      [CURBUF+2]:BX Points into [CURBUF] to start of entry
                                ;      [CURBUF+2]:SI Points into [CURBUF] to dir_first field for entry
                                ;      AH = device ID
                                ;         bit 7 of AH set if device SI and BX
                                ;         will point DOSGROUP relative The firclus
                                ;         field of the device entry contains the device pointer
                                ;      [NAME1] Has name looked for
                                ;      If last element is a directory zero is set and:
                                ;         [DIRSTART],[SECCLUSPOS],[DIRSEC],[CLUSNUM], and [CLUSFAC]
                                ;         are set up to start a search on it.
                                ;         unless [NoSetDir] is non zero in which case the return is
                                ;         like that for a file (except for zero flag)
                                ;      If last element is a file zero is reset
                                ;         [DIRSEC],[CLUSNUM],[CLUSFAC],[NXTCLUSNUM],[SECCLUSPOS],
                                ;         [LASTENT], [ENTLAST] are set to continue search of last
                                ;         directory for furthur matches on NAME1 via the NEXTENT
                                ;         entry point in FindEntry (or GETENT entry in GETENTRY in
                                ;         which case [NXTCLUSNUM] and [SECCLUSPOS] need not be valid)
                                ; DS preserved, Others destroyed
                                ;--------------------------------------------------------------------------

                                ;hkn; called from delete.asm, finfo.asm, mknode.asm and rename.asm.
                                ;hkn; DS already set up at this point.

                                    ; 21/11/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)

                                    ; 15/02/2024 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMDOS.COM)
                                    ; PCDOS 7.1 IBMDOS.COM - DOSCODE:8B99h

                                GETPATH:
                                    ;mov    word [CREATING],0E500h
00004B13 C706[7E05]00E5             MOV     WORD [CREATING],DIRFREE*256+0 ; Not Creating, not DEL *.*

                                ; Same as GetPath only CREATING and DELALL already set

                                    ;entry  GetPathNoSet
                                GetPathNoSet:
                                    ;;mov   byte [EXTERR_LOCUS],2
                                    ;MOV    byte [EXTERR_LOCUS],errLOC_Disk
                                    ; 15/02/2024 (PCDOS 7.1 IBMDOS.COM)
00004B19 E8CAC7                     call    set_exerr_locus_disk
                                    ;
00004B1C C706[E205]FFFF             MOV     word [CURBUF],-1      ; initial setting

                                ; See if the input indicates a device that has already been detected. If so,
                                ; go build the guy quickly. Otherwise, let findpath find the device.

00004B22 8B3E[B205]                 MOV     DI,[WFP_START]        ; point to the beginning of the name
                                    ;cmp    word [DI+1],5C3Ah
                                    ;CMP    WORD [DI+1],'\' << 8 + ':'
00004B26 817D013A5C                 cmp     word [DI+1],':\'
00004B2B 7435                       JZ      short CrackIt

                                ; Let ChkDev find it in the device list

00004B2D 83C703                     ADD     DI,3
                                    ; 18/08/2018
                                    ;MOV    SI,DI                 ; let CHKDEV see the original name
                                    ; 21/11/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
                                    ; 16/12/2022
                                    ;mov    si,di ; not required ! (it is written in CHKDEV proc already!)
00004B30 E8B200                     CALL    CHKDEV
00004B33 722B                       JC      short InternalError

                                Build_devJ:
```

```
29002 00004B35 A0[6D05]                 MOV     AL,[SATTRIB]
29003 00004B38 A2[6B05]                 MOV     [ATTRIB],AL
29004                                   ; 15/02/2024
29005                                   ;;mov   byte [EXTERR_LOCUS],1
29006                                   ;MOV    byte [EXTERR_LOCUS],errLOC_Unk ; In the particular case of
29007                                                                   ; "finding" a char device
29008                                                                   ; set LOCUS to Unknown. This makes
29009                                                                   ; certain idiotic problems reported
29010                                                                   ; by a certain 3 letter OEM go away.
29011                                   ; 15/02/2024 (PCDOS 7.1 IBMDOS.COM)
29012 00004B3B E89FC7                   call    set_exerr_locus_unk
29013
29014                                ; Take name in name1 and pack it back into where wfp_start points. This
29015                                ; guarantees wfp_start pointing to a canonical representation of a device.
29016                                ; We are allowed to do this as GetPath is *ALWAYS* called before entering a
29017                                ; wfp into the share set.
29018                                ;
29019                                ; We copy chars from name1 to wfp_start remembering the position of the last
29020                                ; non-space seen +1.  This position is kept in DX.
29021
29022                                ;hkn; SS is DOSDATA
29023 00004B3E 16                       push    ss
29024 00004B3F 07                       pop     es
29025
29026                                ;hkn; NAME1 is in DOSDATA
29027 00004B40 BE[4B05]                 mov     si,NAME1
29028 00004B43 8B3E[B205]               mov     di,[WFP_START]
29029 00004B47 89FA                     mov     dx,di
29030 00004B49 B90800                   mov     cx,8                    ; 8 chars in device name
29031                                MoveLoop:
29032 00004B4C AC                       lodsb
29033 00004B4D AA                       stosb
29034 00004B4E 3C20                     cmp     al," "
29035 00004B50 7402                     jz      short NoSave
29036
29037 00004B52 89FA                     mov     dx,di
29038                                NoSave:
29039 00004B54 E2F6                     loop    MoveLoop
29040
29041                                ; DX is the position of the last seen non-space + 1. We terminate the name
29042                                ; at this point.
29043
29044 00004B56 89D7                      mov     di,dx
29045                                   ;mov    byte [di],0             ; end of string
29046                                   ; 15/02/2024
29047 00004B58 880D                     mov     [di],cl ; 0
29048 00004B5A E8D502                   call    Build_device_ent        ; Clears carry sets zero
29049 00004B5D FEC0                     INC     AL                      ; reset zero
29050 00004B5F C3                       retn
29051
29052                                InternalError:
29053                                InternalError_loop:
29054 00004B60 EBFE                     JMP     short InternalError_loop ; freeze
29055
29056                                ; Start off at the correct spot. Optimize if the current dir part is valid.
29057
29058                                CrackIt:
29059                                ; 15/02/2024
29060                                %if 0
29061                                   MOV     SI,[CURR_DIR_END]       ; get current directory pointer
29062                                   CMP     SI,-1                   ; valid?
29063                                   JNZ     short LOOK_SING                 ; Yes, use it.
29064                                   LEA     SI,[DI+3]               ; skip D:\.
29065                                LOOK_SING:
29066                                %endif
29067                                   ;mov    byte [ATTRIB],16h
29068 00004B62 C606[6B05]16            MOV     byte [ATTRIB],attr_directory+attr_system+attr_hidden
29069                                                                   ; Attributes to search through Dirs
29070 00004B67 C43E[A205]               LES     DI,[THISCDS]
29071 00004B6B B8FFFF                   MOV     AX,-1 ; 0FFFFh
29072                                   ;;;
29073                                   ; 15/02/2024 (PCDOS 7.1 IBMDOS.COM)
29074                                   ;mov    bx,[es:di+4Bh]
29075 00004B6E 268B5D4B                 mov     bx,[es:di+curdir.ID+2]
29076 00004B72 891E[EE0A]               mov     [ROOTCLUST_HW],bx
29077                                   ;;;
29078                                   ;mov    bx,[es:di+73]
29079 00004B76 268B5D49                 MOV     BX,[ES:DI+curdir.ID]
29080 00004B7A 8B36[B605]               MOV     SI,[CURR_DIR_END]
29081
29082                                ; AX = -1
29083                                ; BX = cluster number of current directory. This number is -1 if the media
29084                                ;      has been uncertainly changed.
29085                                ; SI = offset in DOSGroup into path to end of current directory text. This
29086                                ;      may be -1 if no current directory part has been used.
29087
29088 00004B7E 39C6                       CMP     SI,AX                   ; if Current directory is not part
29089 00004B80 7449                     JZ      short NO_CURR_D                 ; then we must crack from root
29090                                   ;;;
29091                                   ; 15/02/2024 (PCDOS 7.1 IBMDOS.COM)
29092 00004B82 3906[EE0A]               cmp     [ROOTCLUST_HW],ax ; 0FFFFh
29093 00004B86 7504                     jnz     short CrackIt2
29094                                   ;;;
29095 00004B88 39C3                     CMP     BX,AX                   ; is the current directory cluster valid
29096
29097                                ; DOS 3.3  6/25/86
29098 00004B8A 743F                     JZ      short NO_CURR_D                 ; no, crack from the root
29099
29100                                CrackIt2:   ; 15/02/2024 - Retro DOS v5.0
29101
29102                                   ;test   byte [FastOpenFlg],1
29103 00004B8C F606[4611]01             TEST    byte [FastOpenFlg],FastOpen_Set ; for fastopen ?
29104 00004B91 7445                     JZ      short GOT_SEARCH_CLUSTER        ; no
29105 00004B93 06                       PUSH    ES                      ; save registers
29106 00004B94 57                       PUSH    DI
29107 00004B95 51                       PUSH    CX
29108 00004B96 FF74FF                   PUSH    word [SI-1]             ; save \ and 1st char of next element
29109 00004B99 56                       PUSH    SI
29110 00004B9A 53                       PUSH    BX
29111                                   ;;; 15/02/2024
29112 00004B9B FF36[EE0A]               push    word [ROOTCLUST_HW]
29113                                   ;;;
29114 00004B9F C644FF00                 MOV     BYTE [SI-1],0           ; call fastopen to look up cur dir info
29115 00004BA3 8B36[B205]               MOV     SI,[WFP_START]
29116
29117                                ;hkn; FastOpenTable, Dir_Info_Buff & FastOpen_Ext_Info are in DOSDATA
29118 00004BA7 BB[3C11]                 MOV     BX,FastOpenTable
29119 00004BAA BF[7C0D]                 MOV     DI,Dir_Info_Buff
29120 00004BAD B9[4711]                 MOV     CX,FastOpen_Ext_Info
29121                                   ;mov    al,1
29122 00004BB0 B001                     MOV     AL,FONC_Look_up
29123 00004BB2 1E                       PUSH    DS
29124 00004BB3 07                       POP     ES
29125                                   ;call   far [BX+2]
```

```
29126 00004BB4 FF5F02                   CALL    far [BX+fastopen_entry.name_caching]
29127 00004BB7 7203                     JC      short GO_Chk_end1     ;fastopen not installed, or wrong drive.
29128                                                                 ; Go to Got_Srch_cluster
29129                                   ; 29/12/2022
29130                                   ;CMP    BYTE [SI],0           ;fastopen has current dir info?
29131                                   ;JE     short GO_Chk_end      ;yes. Go to got_search_cluster
29132                                   ;stc
29133                                   ;jmp    short GO_Chk_end      ;Go to No_Curr_D
29134
29135 00004BB9 803C01                   cmp     byte [si],1
29136                                 GO_Chk_end1:       ; 29/12/2022
29137 00004BBC F5                       cmc
29138                                   ; [si] = 0 -> cf = 0
29139                                   ; [si] > 0 -> cf = 1
29140
29141                                 ;GO_Chk_end1:
29142                                   ; 29/12/2022
29143                                   ;clc
29144
29145                                 GO_Chk_end:                         ; restore registers
29146                                   ;;; 15/02/2024
29147 00004BBD 8F06[EE0A]               pop     word [ROOTCLUST_HW]
29148                                   ;;;
29149 00004BC1 5B                       POP     BX
29150 00004BC2 5E                       POP     SI
29151 00004BC3 8F44FF                   POP     word [SI-1]
29152 00004BC6 59                       POP     CX
29153 00004BC7 5F                       POP     DI
29154 00004BC8 07                       POP     ES
29155 00004BC9 730D                     JNC     short GOT_SEARCH_CLUSTER ; crack based on cur dir
29156
29157                                 ; DOS 3.3  6/25/86
29158                                 ;
29159                                 ; We must cract the path beginning at the root. Advance pointer to beginning
29160                                 ; of path and go crack from root.
29161
29162                                 NO_CURR_D:
29163 00004BCB 8B36[B205]               MOV     SI,[WFP_START]
29164                                   ;LEA    SI,[SI+3]             ; Skip "d:/"
29165                                   ; 15/02/2024
29166 00004BCF 83C603                   add     si,3
29167 00004BD2 C42E[8A05]               LES     BP,[THISDPB]          ; Get ES:BP
29168 00004BD6 EB37                     JMP     short ROOTPATH
29169
29170                                 ; We are able to crack from the current directory part. Go set up for search
29171                                 ; of specified cluster.
29172
29173                                 GOT_SEARCH_CLUSTER:
29174 00004BD8 C42E[8A05]               LES     BP,[THISDPB]          ; Get ES:BP
29175 00004BDC E880FD                   call    SETDIRSRCH
29176                                   ;JC     short SETFERR
29177                                   ;JMP    short FINDPATH
29178                                   ; 16/12/2022
29179 00004BDF 733E                     jnc     short FINDPATH ; 17/08/2018
29180                                   ; 21/11/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
29181                                   ;JC     short SETFERR
29182                                   ;JMP    short FINDPATH
29183                                 SETFERR:
29184 00004BE1 30C9                     XOR     CL,CL                 ; set zero
29185 00004BE3 F9                       STC
29186 00004BE4 C3                       retn
29187
29188                                 ;--------------------------------------------------------------------------
29189                                 ;
29190                                 ; Procedure Name : ChkDev
29191                                 ;
29192                                 ; Check to see if the name at DS:DI is a device. Returns carry set if not a
29193                                 ;   device.
29194                                 ; Blasts CX,SI,DI,AX,BX
29195                                 ;--------------------------------------------------------------------------
29196
29197                                 CHKDEV:
29198 00004BE5 89FE                     MOV     SI,DI
29199                                   ;MOV    DI,SS
29200                                   ;MOV    ES,DI
29201                                   ; 27/06/2024
29202 00004BE7 16                       push    ss
29203 00004BE8 07                       pop     es
29204
29205 00004BE9 BF[4B05]                 MOV     DI,NAME1
29206 00004BEC B90900                   MOV     CX,9
29207                                 TESTLOOP:
29208 00004BEF E8A411                   call    GETLET
29209
29210 00004BF2 3C2E                     CMP     AL,'.'
29211 00004BF4 740E                     JZ      short TESTDEVICE
29212 00004BF6 E8F111                   call    PATHCHRCMP
29213 00004BF9 7407                     JZ      short NOTDEV
29214 00004BFB 08C0                     OR      AL,AL
29215 00004BFD 7405                     JZ      short TESTDEVICE
29216
29217 00004BFF AA                       STOSB
29218 00004C00 E2ED                     LOOP    TESTLOOP
29219                                 NOTDEV:
29220 00004C02 F9                       STC
29221 00004C03 C3                       retn
29222
29223                                 TESTDEVICE:
29224                                   ;ADD    CX,2
29225                                   ; 24/09/2023
29226 00004C04 41                       inc     cx
29227 00004C05 41                       inc     cx
29228 00004C06 B020                     MOV     AL,' '
29229 00004C08 F3AA                     REP     STOSB
29230                                   ;MOV    AX,SS
29231                                   ;MOV    DS,AX
29232                                   ; 27/06/2024
29233 00004C0A 16                       push    ss
29234 00004C0B 1F                       pop     ds
29235                                   ;call   DEVNAME
29236                                   ;retn
29237                                   ; 18/12/2022
29238 00004C0C E9C501                   jmp     DEVNAME
29239
29240                                 ;Break      <ROOTPATH, FINDPATH -- PARSE A PATH>
29241                                 ;--------------------------------------------------------------------------
29242                                 ;
29243                                 ; Procedure Name : ROOTPATH,FINDPATH
29244                                 ;
29245                                 ; Inputs:
29246                                 ;   Same as FINDPATH but,
29247                                 ;   SI Points to asciz string of path which is assumed to start at
29248                                 ;           the root (no leading '/').
29249                                 ; Function:
```

```
                                ;   Search from root for path
                                ; Outputs:
                                ;   Same as FINDPATH but:
                                ;   If root directory specified, [CURBUF] and [NAME1] are NOT set, and
                                ;   [NoSetDir] is ignored.
                                ;-----------------------------------------------------------------------------

                                    ; 21/11/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
                                    ; DOSCODE:7F47h (MSDOS 5.0, MSDOS.SYS)

                                    ; 15/02/2024 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMDOS.COM)
                                    ; PCDOS 7.1 IBMDOS.COM - DOSCODE:8C9Dh

                                    ; (MSDOS 6.22 MSDOS.SYS - DOSCODE:7F82h)
                                    ; (Windows ME IO.SYS - BIOSCODE:829Eh)

                                ROOTPATH:
00004C0F E879FD                     call    SETROOTSRCH
                                    ; 24/09/2023
00004C12 30E4                       xor     ah,ah
                                    ;CMP    BYTE [SI],0
00004C14 3824                       cmp     [si],ah ; 0
00004C16 7507                       JNZ     short FINDPATH

                                    ;;;
                                    ;; 15/02/2024
                                    ; Windows ME IO.SYS - BIOSCODE:82A6h
                                    ;mov    word [CURBUF],0FFFFh
                                    ;;;

                                ; Root dir specified
00004C18 A0[6D05]                   MOV     AL,[SATTRIB]
00004C1B A2[6B05]                   MOV     [ATTRIB],AL
                                    ; 24/09/2023
                                    ;XOR    AH,AH                   ; Sets "device ID" byte, sets zero
                                                                    ; (dir), clears carry.
00004C1E C3                         retn

                                ; Inputs:
                                ;   [ATTRIB] Set to get through directories
                                ;   [SATTRIB] Set to find last element
                                ;   ES:BP Points to DPB
                                ;   SI Points to asciz string of path (no leading '/').
                                ;   [SECCLUSPOS] = 0
                                ;   [DIRSEC] = Phys sec # of first sector of directory
                                ;   [CLUSNUM] = Cluster # of next cluster
                                ;   [CLUSFAC] = Sectors per cluster
                                ;   [NoSetDir] set
                                ;   [CURR_DIR_END] Points to end of Current dir part of string
                                ;           ( = -1 if current dir not involved, else
                                ;            Points to first char after last "/" of current dir part)
                                ;   [THISCDS] Points to CDS being used
                                ;   [CREATING] and [DELALL] set
                                ; Function:
                                ;   Parse path name
                                ; Outputs:
                                ;   ID1 field of [THISCDS] updated appropriately
                                ;   [ATTRIB] = [SATTRIB]
                                ;   ES:BP Points to DPB
                                ;   [THISDPB] = ES:BP
                                ;   Carry set if bad path
                                ;      SI Points to path element causing failure
                                ;      Zero set
                                ;         [DIRSTART],[DIRSEC],[CLUSNUM], and [CLUSFAC] are set up to
                                ;         start a search on the last directory
                                ;         CL is zero if there is a bad name in the path
                                ;         CL is non-zero if the name was simply not found
                                ;            [ENTFREE] may have free spot in directory
                                ;            [NAME1] is the name.
                                ;            CL = 81H if '*'s or '?' in NAME1, 80H otherwise
                                ;      Zero reset
                                ;         File in middle of path or bad name in path
                                ;           or path too long or malformed path
                                ;   ELSE
                                ;      [CURBUF] contains directory record with match
                                ;      [CURBUF+2]:BX Points into [CURBUF] to start of entry
                                ;      [CURBUF+2]:SI Points to fcb_FIRCLUS field for entry
                                ;      [NAME1] Has name looked for
                                ;      AH = device ID
                                ;         bit 7 of AH set if device SI and BX
                                ;         will point DOSGROUP relative The firclus
                                ;         field of the device entry contains the device pointer
                                ;      If last element is a directory zero is set and:
                                ;         [DIRSTART],[SECCLUSPOS],[DIRSEC],[CLUSNUM], and [CLUSFAC]
                                ;         are set up to start a search on it,
                                ;         unless [NoSetDir] is non zero in which case the return is
                                ;         like that for a file (except for zero flag)
                                ;      If last element is a file zero is reset
                                ;         [DIRSEC],[CLUSNUM],[CLUSFAC],[NXTCLUSNUM],[SECCLUSPOS],
                                ;         [LASTENT], [ENTLAST] are set to continue search of last
                                ;         directory for furthur matches on NAME1 via the NEXTENT
                                ;         entry point in FindEntry (or GETENT entry in GETENTRY in
                                ;         which case [NXTCLUSNUM] and [SECCLUSPOS] need not be valid)
                                ; Destroys all other registers

                                    ; 21/11/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
                                    ; DOSCODE:7F58h (MSDOS 5.0, MSDOS.SYS)

                                    ; 15/02/2024 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMDOS.COM)
                                    ; PCDOS 7.1 IBMDOS.COM - DOSCODE:8CAEh

                                    ;entry  FINDPATH
                                FINDPATH:
00004C1F 06                         PUSH    ES                      ; Save ES:BP
00004C20 56                         PUSH    SI
00004C21 89F7                       MOV     DI,SI
00004C23 8B0E[C205]                 MOV     CX,[DIRSTART]           ; Get start clus of dir being searched
00004C27 833E[B605]FF               CMP     word [CURR_DIR_END],-1
00004C2C 7415                       JZ      short NOIDS             ; No current dir part
00004C2E 3B3E[B605]                 CMP     DI,[CURR_DIR_END]
00004C32 750F                       JNZ     short NOIDS             ; Not to current dir end yet
00004C34 C43E[A205]                 LES     DI,[THISCDS]
                                    ;;;
                                    ; 15/02/2024 (PCDOS 7.1 IBMDOS.COM)
00004C38 A1[DC0A]                   mov     ax,[DIRSTART_HW]
                                    ;mov    [es:di+4Bh],ax
00004C3B 2689454B                   mov     [es:di+curdir.ID+2],ax
                                    ;;;
                                    ;mov    [es:di+73],cx
00004C3F 26894D49                   MOV     [ES:DI+curdir.ID],CX    ; Set current directory cluster
                                NOIDS:

                                ; Parse the name off of DS:SI into NAME1. AL = 1 if there was a meta
                                ; character in the string. CX,DI may be destroyed.
```

```
                                ;
                                ;   invoke  NAMETRANS
                                ;   MOV     CL,AL
                                ;
                                ; The above is the slow method. The name has *already* been munged by
                                ; TransPath so no special casing needs to be done. All we do is try to copy
                                ; the name until ., \ or 0 is hit.

                                    ;MOV    AX,SS
                                    ;MOV    ES,AX
                                    ; 15/02/2024 (PCDOS 7.1 IBMDOS.COM)
00004C43 16                         push    ss
00004C44 07                         pop     es

                                ;hkn; Name1 is in DOSDATA
00004C45 BF[4B05]                   MOV     DI,NAME1
00004C48 B82020                     MOV     AX,'  ' ; 2020h
00004C4B AA                         STOSB
00004C4C AB                         STOSW
00004C4D AB                         STOSW
00004C4E AB                         STOSW
00004C4F AB                         STOSW
00004C50 AB                         STOSW

                                ;hkn; Name1 is in DOSDATA
00004C51 BF[4B05]                   MOV     DI,NAME1
00004C54 30E4                       XOR     AH,AH                   ; bits for CL
                                GetNam:
                                    ; 19/05/2019 - Retro DOS v4.0
                                    ;INC    CL ; ?*! ; MSDOS 6.0  ;AN000; KK increment volid count

                                    ; 21/11/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
                                    ; 16/12/2022
                                    ;inc    cl ; not required !

00004C56 AC                         LODSB
00004C57 3C2E                       CMP     AL,'.' ; 2Eh
00004C59 7412                       JZ      short _SetExt
00004C5B 08C0                       OR      AL,AL
00004C5D 7424                       JZ      short _GetDone
00004C5F 3C5C                       CMP     AL,'\' ; 5Ch
00004C61 7420                       JZ      short _GetDone
00004C63 3C3F                       CMP     AL,'?' ; 3Fh
00004C65 7503                       JNZ     short StoNam
00004C67 80CC01                     OR      AH,1
                                StoNam:
00004C6A AA                         STOSB
00004C6B EBE9                       JMP     short GetNam
                                _SetExt:
00004C6D BF[5305]                   MOV     DI,NAME1+8
                                GetExt:
00004C70 AC                         LODSB
00004C71 08C0                       OR      AL,AL
00004C73 740E                       JZ      short _GetDone
00004C75 3C5C                       CMP     AL,'\'
00004C77 740A                       JZ      short _GetDone
00004C79 3C3F                       CMP     AL,'?'
00004C7B 7503                       JNZ     short StoExt
00004C7D 80CC01                     OR      AH,1
                                StoExt:
00004C80 AA                         STOSB
00004C81 EBED                       JMP     short GetExt
                                _GetDone:
00004C83 4E                         DEC     SI
00004C84 88E1                       MOV     CL,AH  ; 0 or 1 ; 29/12/2022
00004C86 80C980                     OR      CL,80H
00004C89 5F                         POP     DI                      ; Start of this element
00004C8A 07                         POP     ES                      ; Restore ES:BP
00004C8B 39FE                       CMP     SI,DI
00004C8D 7503                       JNZ     short check_device
00004C8F E9ED00                     JMP     _BADPATH                ; NUL parse (two delims most likely)
                                check_device:
00004C92 56                         PUSH    SI                      ; Start of next element
                                    ;MOV    AL,[SI]
                                    ; 15/02/2024
00004C93 08C0                       OR      AL,AL
                                    ; 23/09/2023
                                    ;cmp    byte [si],0
00004C95 7508                       JNZ     short NOT_LAST

                                ; for last element of the path switch to the correct search attributes

00004C97 8A3E[6D05]                 MOV     BH,[SATTRIB]
00004C9B 883E[6B05]                 MOV     [ATTRIB],BH

                                NOT_LAST:

                                ; check name1 to see if we have a device...

00004C9F 06                         PUSH    ES                      ; Save ES:BP

                                ;hkn; SS is DOSDATA
                                    ;context ES
00004CA0 16                         push    ss
00004CA1 07                         pop     es
00004CA2 E82F01                     call    DEVNAME                 ; blast BX
00004CA5 07                         POP     ES                      ; Restore ES:BP
00004CA6 7208                       JC      short FindFile          ; Not a device
00004CA8 08C0                       OR      AL,AL                   ; Test next char again
                                    ;JZ     short GO_BDEV
                                    ;JMP    FILEINPATH              ; Device name in middle of path
                                    ; 27/06/2024
00004CAA 752D                       jnz     short FILEINPATH_j

                                GO_BDEV:
00004CAC 5E                         POP     SI                      ; Points to NUL at end of path
00004CAD E985FE                     JMP     Build_devJ

                                FindFile:
                                ;;;; 7/28/86
00004CB0 803E[4B05]E5               CMP     BYTE [NAME1],0E5H       ; if 1st char = E5
00004CB5 7505                       JNZ     short NOE5              ; no
00004CB7 C606[4B05]05               MOV     BYTE [NAME1],05H        ; change it to 05
                                NOE5:
                                ;;;; 7/28/86
00004CBC 57                         PUSH    DI                      ; Start of this element
00004CBD 06                         PUSH    ES                      ; Save ES:BP
00004CBE 51                         PUSH    CX                      ; CL return from NameTrans
                                ;DOS 3.3 FastOPen 6/12/86 F.C.

00004CBF E8B002                     CALL    LookupPath              ; call fastopen to get dir entry
00004CC2 7303                       JNC     short DIR_FOUND                 ; found dir entry

                                ;DOS 3.3 FastOPen 6/12/86 F.C.
```

```
29498 00004CC4 E87AFA                  call    FINDENTRY
29499                                  DIR_FOUND:
29500 00004CC7 59                      POP     CX
29501 00004CC8 07                      POP     ES
29502 00004CC9 5F                      POP     DI
29503 00004CCA 7303                    JNC     short LOAD_BUF
29504 00004CCC E9D500                  JMP     BADPATHPOP
29505
29506                                  LOAD_BUF:
29507 00004CCF C53E[E205]              LDS     DI,[CURBUF]
29508                                  ;test   byte [bx+0Bh],10h
29509 00004CD3 F6470B10                TEST    BYTE [BX+dir_entry.dir_attr],attr_directory
29510 00004CD7 7503                    JNZ     short GO_NEXT           ; DOS 3.3
29511                                  FILEINPATH_j:      ; 27/06/2024
29512 00004CD9 E9A700                  JMP     FILEINPATH              ; Error or end of path
29513
29514                                  ; if we are not setting the directory, then check for end of string
29515
29516                                  GO_NEXT:
29517                                  ;hkn; SS override
29518 00004CDC 36803E[4C03]00          CMP     BYTE [SS:NoSetDir],0
29519 00004CE2 7423                    JZ      short SetDir
29520 00004CE4 89FA                    MOV     DX,DI                   ; Save pointer to entry
29521 00004CE6 8CD9                    MOV     CX,DS
29522
29523                                  ;hkn; SS is DOSDATA
29524                                      ;context DS
29525 00004CE8 16                      push    ss
29526 00004CE9 1F                      pop     ds
29527 00004CEA 5F                      POP     DI                      ; Start of next element
29528                                  ; 19/05/2019 - Retro DOS v4.0
29529                                  ; MSDOS 6.0
29530 00004CEB F606[4611]01            TEST    byte [FastOpenFlg],FastOpen_Set ;only DOSOPEN can take advantage of
29531 00004CF0 740B                    JZ      short _nofast                   ; the FastOpen
29532 00004CF2 F606[4611]02            TEST    byte [FastOpenFlg],Lookup_Success ; Lookup just happened
29533 00004CF7 7404                    JZ      short _nofast                   ; no
29534 00004CF9 8B3E[9C0D]              MOV     DI,[Next_Element_Start]         ; no need to insert it again
29535                                  _nofast:
29536 00004CFD 803D00                  CMP     BYTE [DI],0
29537                                  ;;JNZ   short NEXT_ONE          ; DOS 3.3
29538                                  ;;JMP   _SETRET  ; retn                 ; Got it
29539                                  ;retn   ; 05/09/2018
29540                                  ; 21/11/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
29541                                  ;jmp    _SETRET
29542                                  ; 16/12/2022
29543 00004D00 7431                    jz      short _SETRET
29544
29545                                  NEXT_ONE:
29546 00004D02 57                      PUSH    DI                      ; Put start of next element back on stack
29547 00004D03 89D7                    MOV     DI,DX
29548 00004D05 8ED9                    MOV     DS,CX                   ; Get back pointer to entry
29549                                  SetDir:
29550                                      ;;;
29551                                      ; 15/02/2024 (PCDOS 7.1 IBMDOS.COM)
29552 00004D07 31D2                    xor     dx,dx ; 0
29553                                      ;cmp    [es:bp+0Fh],dx
29554 00004D09 2639560F                cmp     [es:bp+DPB.FAT_SIZE],dx ; 0
29555 00004D0D 7503                    jnz      short SetDir2 ; not FAT32
29556 00004D0F 8B54FA                  mov      dx,[si-6]              ; dir_entry.dir_fclus_hi
29557                                  SetDir2:
29558 00004D12 368916[EE0A]            mov     [ss:ROOTCLUST_HW],dx ; 0
29559                                      ;;;
29560 00004D17 8B14                    MOV     DX,[SI]                 ; dir_entry.dir_first
29561
29562                                  ;DOS 3.3 FastOPen 6/12/86 F.C.
29563 00004D19 1E                      PUSH    DS                      ; save [curbuf+2]
29564                                  ;hkn; SS is DOSDATA
29565 00004D1A 16                      push    ss
29566 00004D1B 1F                      pop     ds                      ; set DS Dosgroup
29567                                  ;test   byte [FastOpenFlg],2
29568 00004D1C F606[4611]02            TEST    byte [FastOpenFlg],Lookup_Success
29569 00004D21 7411                    JZ      short DO_NORMAL             ; fastopen not in memory or path not
29570 00004D23 89D3                    MOV     BX,DX                   ; not found
29571 00004D25 8B3E[BC05]              MOV     DI,[CLUSNUM]            ; clusnum was set in LookupPath
29572 00004D29 50                      PUSH    AX                      ; save device id (AH)
29573 00004D2A E832FC                  call    SETDIRSRCH
29574 00004D2D 58                      POP     AX                      ; restore device id (AH)
29575 00004D2E 83C402                  ADD     SP,2                    ; pop ds in stack
29576 00004D31 EB36                    JMP     short FAST_OPEN_SKIP
29577
29578                                      ; 16/12/2022
29579                                  _SETRET:
29580 00004D33 C3                      retn
29581
29582                                  DO_NORMAL:
29583 00004D34 1F                      POP     DS                      ; DS = [curbuf + 2]
29584                                  ;DOS 3.3 FastOPen 6/12/86 F.C.
29585
29586 00004D35 29FB                    SUB     BX,DI                   ; Offset into sector of start of entry
29587 00004D37 29FE                    SUB     SI,DI                   ; Offset into sector of dir_first
29588 00004D39 53                      PUSH    BX
29589 00004D3A 50                      PUSH    AX
29590 00004D3B 56                      PUSH    SI
29591 00004D3C 51                      PUSH    CX
29592
29593                                  ; 15/02/2024
29594                                  %if 0
29595                                      ;push   word [di+6]
29596                                      PUSH    WORD [DI+BUFFINFO.buf_sector] ;AN000;>32mb
29597                                      ; 19/05/2019
29598                                      ; MSDOS 6.0
29599                                      ;push   word [di+8]
29600                                      PUSH    WORD [DI+BUFFINFO.buf_sector+2]      ;AN000;>32mb
29601                                  %else
29602                                      ; 15/02/2024 (PCDOS 7.1 IBMDOS.COM)
29603                                      ;lds    bx,[di+6]
29604 00004D3D C55D06                  lds     bx,[di+BUFFINFO.buf_sector]
29605 00004D40 53                      push    bx
29606 00004D41 1E                      push    ds
29607                                  %endif
29608
29609 00004D42 89D3                    MOV     BX,DX
29610
29611                                  ;hkn; SS is DOSDATA
29612                                      ;context DS
29613 00004D44 16                      push    ss
29614 00004D45 1F                      pop     ds
29615                                      ;invoke SETDIRSRCH          ; This uses UNPACK which might blow
29616 00004D46 E816FC                  call    SETDIRSRCH              ; the entry sector buffer
29617                                  ; 19/05/2019
29618                                  ; MSDOS 6.0
29619 00004D49 8F06[0706]              POP     word [HIGH_SECTOR]
29620 00004D4D 5A                      POP     DX
29621 00004D4E 7203                    JC      short SKIP_GETB
```

```
                                    ; 22/09/2023
                                    ;;mov   byte [ALLOWED],18h
                                    ;MOV    byte [ALLOWED],Allowed_RETRY+Allowed_FAIL ; *
                                    ;XOR    AL,AL ; *
                                    ;;invoke GETBUFFR               ; Get the entry buffer back
                                    ;call   GETBUFFR
00004D50 E8D61B                     call    GETBUFFER ; * ; pre-read
                                SKIP_GETB:
00004D53 59                         POP     CX
00004D54 5E                         POP     SI
00004D55 58                         POP     AX
00004D56 5B                         POP     BX
00004D57 7305                       JNC     short SET_THE_BUF
00004D59 5F                         POP     DI                      ; Start of next element
00004D5A 89FE                       MOV     SI,DI                   ; Point with SI
00004D5C EB21                       JMP     SHORT _BADPATH

                                SET_THE_BUF:
00004D5E E81DF2                     call    SET_BUF_AS_DIR
00004D61 8B3E[E205]                 MOV     DI,[CURBUF]
00004D65 01FE                       ADD     SI,DI                   ; Get the offsets back
00004D67 01FB                       ADD     BX,DI
                                ; DOS 3.3 FastOpen 6/12/86  F.C.
                                FAST_OPEN_SKIP:
00004D69 5F                         POP     DI                      ; Start of next element
00004D6A E8C202                     CALL    InsertPath              ; insert dir entry info
                                ; DOS 3.3 FastOpen 6/12/86  F.C.
00004D6D 8A05                       MOV     AL,[DI]
00004D6F 08C0                       OR      AL,AL
00004D71 74C0                       JZ      short _SETRET           ; At end
00004D73 47                         INC     DI                      ; Skip over "/"
00004D74 89FE                       MOV     SI,DI                   ; Point with SI
00004D76 E87110                     call    PATHCHRCMP
00004D79 7503                       JNZ     short find_bad_name     ; oops
00004D7B E9A1FE                     JMP     FINDPATH                ; Next element

                                find_bad_name:
00004D7E 4E                         DEC     SI                      ; Undo above INC to get failure point
                                _BADPATH:
00004D7F 30C9                       XOR     CL,CL                   ; Set zero
00004D81 EB28                       JMP     SHORT BADPRET

                                FILEINPATH:
00004D83 5F                         POP     DI                      ; Start of next element

                                ;hkn; SS is DOSDATA
                                    ;context DS                     ; Got to from one place with DS gone
00004D84 16                         push    ss
00004D85 1F                         pop     ds

                                ; DOS 3.3 FastOpen
                                    ;test   byte [FastOpenFlg],1
00004D86 F606[4611]01               TEST    byte [FastOpenFlg],FastOpen_Set  ; do this here is we don't want to
00004D8B 740B                       JZ      short NO_FAST           ; device info to fastopen
                                    ;test   byte [FastOpenFlg],2
00004D8D F606[4611]02               TEST    byte [FastOpenFlg],Lookup_Success
00004D92 7404                       JZ      short NO_FAST
00004D94 8B3E[9C0D]                 MOV     DI,[Next_Element_Start]  ; This takes care of one time lookup
                                                                     ; success
                                NO_FAST:
                                ; DOS 3.3 FastOpen
00004D98 8A05                       MOV     AL,[DI]
00004D9A 08C0                       OR      AL,AL
                                    ;JZ     short INCRET
                                    ;MOV    SI,DI                   ; Path too long
                                    ;JMP    SHORT BADPRET
                                    ; 27/06/2024
00004D9C 750B                       jnz     short BADPRET_X

                                INCRET:
                                ; DOS 3.3 FasOpen 6/12/86  F.C.

00004D9E E88E02                     CALL    InsertPath              ; insert dir entry info

                                ; DOS 3.3 FasOpen 6/12/86  F.C.
00004DA1 FEC0                       INC     AL                      ; Reset zero
                                    ; 16/12/2022
                                ;_SETRET:
00004DA3 C3                         retn

                                BADPATHPOP:
00004DA4 5E                         POP     SI                      ; Start of next element
00004DA5 8A04                       MOV     AL,[SI]
                                    ; 27/06/2024
                                    ;MOV    SI,DI                   ; Start of bad element
00004DA7 08C0                       OR      AL,AL                   ; zero if bad element is last, non-zero if path too long
                                BADPRET_X:  ; 27/06/2024
00004DA9 89FE                       mov     si,di
                                BADPRET:
00004DAB A0[6D05]                   MOV     AL,[SATTRIB]
00004DAE A2[6B05]                   MOV     [ATTRIB],AL             ; Make sure return correct
00004DB1 F9                         STC
00004DB2 C3                         retn

                                ;Break      <STARTSRCH -- INITIATE DIRECTORY SEARCH>
                                ;-----------------------------------------------------------------------
                                ;
                                ; Procedure Name : STARTSRCH
                                ;
                                ; Inputs:
                                ;   [THISDPB] Set
                                ; Function:
                                ;   Set up a search for GETENTRY and NEXTENTRY
                                ; Outputs:
                                ;   ES:BP = Drive parameters
                                ;   Sets up LASTENT, ENTFREE=ENTLAST=-1, VOLID=0
                                ; Destroys ES,BP,AX
                                ;-----------------------------------------------------------------------

                                STARTSRCH:
00004DB3 C42E[8A05]                 LES     BP,[THISDPB]
00004DB7 31C0                       XOR     AX,AX
00004DB9 A3[4803]                   MOV     [LASTENT],AX
00004DBC A2[7B05]                   MOV     [VOLID],AL              ; No volume ID found
00004DBF 48                         DEC     AX
00004DC0 A3[D805]                   MOV     [ENTFREE],AX
00004DC3 A3[DA05]                   MOV     [ENTLAST],AX
00004DC6 C3                         retn

                                ;BREAK <MatchAttributes - the final check for attribute matching>
                                ;-----------------------------------------------------------------------
                                ; Procedure Name : MatchAttributes
                                ;
                                ; Input:    [Attrib] = attribute to search for
```

```
29746                                   ;       CH = found attribute
29747                                   ; Output:   JZ <match>
29748                                   ;       JNZ <nomatch>
29749                                   ; Registers modified: noneski
29750                                   ;-----------------------------------------------------------------------------
29751
29752                                   MatchAttributes:
29753 00004DC7 50                           PUSH    AX
29754
29755                                   ;hkn; SS override
29756 00004DC8 36A0[6B05]                   MOV     AL,[ss:ATTRIB]          ; AL <- SearchSet
29757 00004DCC F6D0                         NOT     AL                      ; AL <- SearchSet'
29758 00004DCE 20E8                         AND     AL,CH                   ; AL <- SearchSet' and FoundSet
29759                                       ;and    al,16h
29760 00004DD0 2416                         AND     AL,attr_all     ; AL <- SearchSet' and FoundSet and Important
29761                                   ;
29762                                   ; the result is non-zero if an attribute is not in the search set
29763                                   ; and in the found set and in the important set. This means that we do not
29764                                   ; have a match. Do a JNZ <nomatch> or JZ <match>
29765                                   ;
29766 00004DD2 58                           POP     AX
29767 00004DD3 C3                           retn
29768
29769                                   ; 19/05/2019 - Retro DOS v4.0
29770                                   ; DOSCODE:8148h (MSDOS 6.21, MSDOS.SYS)
29771
29772                                   ; 21/11/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
29773                                   ; DOSCODE:810Dh (MSDOS 5.0, MSDOS.SYS)
29774
29775                                   ; 16/02/2024 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMDOS.COM)
29776                                   ; DOSCODE:8E74h (PCDOS 7.1, IBMDOS.COM)
29777
29778                                   ; (MSDOS 6.22 MSDOS.SYS - DOSCODE:8148h)
29779                                   ; (Windows ME IO.SYS - BIOSCODE:843Ch)
29780
29781                                   ;Break <DevName - Look for name of device>
29782                                   ;-----------------------------------------------------------------------------
29783                                   ;
29784                                   ; Procedure Name : DevName
29785                                   ;
29786                                   ; Inputs:
29787                                   ;   DS,ES:DOSDATA
29788                                   ;   Filename in NAME1
29789                                   ;   ATTRIB set so that we can error out if looking for Volume IDs
29790                                   ; Function:
29791                                   ;   Determine if file is in list of I/O drivers
29792                                   ; Outputs:
29793                                   ;   Carry set if not a device
29794                                   ;   ELSE
29795                                   ;   Zero flag set
29796                                   ;   BH = Bit 7,6 = 1, bit 5 = 0 (cooked mode)
29797                                   ;        bits 0-4 set from low byte of attribute word
29798                                   ;   DEVPT = DWORD pointer to Device header of device
29799                                   ; BX destroyed, others preserved
29800                                   ;-----------------------------------------------------------------------------
29801
29802                                   DEVNAME:
29803                                       ; 28/07/2018 - Retro DOS v3.0
29804                                       ; IBMDOS.COM (MSDOS 3.3) - Offset 49FBh
29805
29806 00004DD4 56                           PUSH    SI
29807 00004DD5 57                           PUSH    DI
29808 00004DD6 51                           PUSH    CX
29809 00004DD7 50                           PUSH    AX
29810
29811                                   ; E5 special code
29812 00004DD8 FF36[4B05]                   PUSH    WORD [NAME1]
29813 00004DDC 803E[4B05]05                 CMP     byte [NAME1],5
29814 00004DE1 7505                         JNZ     short NOKTR
29815 00004DE3 C606[4B05]E5                 MOV     byte [NAME1],0E5h
29816                                   NOKTR:
29817                                       ;test   byte [ATTRIB],8
29818 00004DE8 F606[6B05]08                 TEST    byte [ATTRIB],attr_volume_id
29819                                                                       ; If looking for VOL id don't find devs
29820 00004DED 7520                         JNZ     short RET31
29821
29822                                   ;hkn; NULDEV is in DOSDATA
29823 00004DEF BE[4800]                     MOV     SI,NULDEV
29824                                   LOOKIO:
29825                                       ; 21/11/2022
29826                                       ;test   byte [SI+SYSDEV.ATT+1],80h
29827                                       ; 17/12/2022
29828                                       ;test   byte [si+5],80h
29829 00004DF2 F6440580                     test    byte [SI+SYSDEV.ATT+1],(DEVTYP>>8)
29830                                       ;;test  word [si+4],8000h
29831                                       ;TEST   word [SI+SYSDEV.ATT],DEVTYP
29832 00004DF6 7410                         JZ      short SKIPDEV           ; Skip block devices (NET and LOCAL)
29833 00004DF8 89F0                         MOV     AX,SI
29834                                       ;add    si,10
29835 00004DFA 83C60A                       ADD     SI,SYSDEV.NAME
29836
29837                                   ;hkn; NAME1 is in DOSDATA
29838 00004DFD BF[4B05]                     MOV     DI,NAME1
29839 00004E00 B90400                       MOV     CX,4                    ; All devices are 8 letters
29840 00004E03 F3A7                         REPE    CMPSW                   ; Check for name in list
29841                                       ;MOV    SI,AX
29842                                       ; 27/06/2024
29843 00004E05 96                           xchg    ax,si
29844 00004E06 7415                         JZ      short IOCHK             ; Found it?
29845                                   SKIPDEV:
29846 00004E08 C534                         LDS     SI,[SI]                 ; Get address of next device
29847 00004E0A 83FEFF                       CMP     SI,-1                   ; At end of list?
29848 00004E0D 75E3                         JNZ     short LOOKIO
29849                                   RET31:
29850 00004E0F F9                           STC                             ; Not found
29851                                   RETNV:
29852 00004E10 8CD1                         MOV     CX,SS
29853 00004E12 8ED9                         MOV     DS,CX
29854
29855 00004E14 8F06[4B05]                   POP     WORD [NAME1]
29856 00004E18 58                           POP     AX
29857 00004E19 59                           POP     CX
29858 00004E1A 5F                           POP     DI
29859 00004E1B 5E                           POP     SI
29860 00004E1C C3                           RETN
29861
29862                                   IOCHK:
29863                                   ;hkn; SS override for DEVPT
29864 00004E1D 368C1E[9C05]                 MOV     [SS:DEVPT+2],DS             ; Save pointer to device
29865                                       ;mov    bh,[si+4]
29866 00004E22 8A7C04                       MOV     BH,[SI+SYSDEV.ATT]
29867 00004E25 80CFC0                       OR      BH,0C0h
29868 00004E28 80E7DF                       and     bh,0DFh
29869                                       ;AND    BH,~(020h)              ; Clears Carry
```

```
29870 00004E2B 368936[9A05]               MOV     [SS:DEVPT],SI
29871 00004E30 EBDE                       JMP     short RETNV
29872
29873                                   ;BREAK <Build_device_ent - Make a Directory entry>
29874                                   ;-----------------------------------------------------------------------------
29875                                   ; Procedure Name : Build_device_ent
29876                                   ;
29877                                   ; Inputs:
29878                                   ;   [NAME1] has name
29879                                   ;   BH is attribute field (supplied by DEVNAME)
29880                                   ;   [DEVPT] points to device header (supplied by DEVNAME)
29881                                   ; Function:
29882                                   ;   Build a directory entry for a device at DEVFCB
29883                                   ; Outputs:
29884                                   ;   BX points to DEVFCB
29885                                   ;   SI points to dir_first field
29886                                   ;   AH = input BH
29887                                   ;   AL = 0
29888                                   ;   dir_first = DEVPT
29889                                   ;   Zero Set, Carry Clear
29890                                   ; DS,ES,BP preserved, others destroyed
29891                                   ;-----------------------------------------------------------------------------
29892
29893                                   Build_device_ent:
29894 00004E32 B82020                       MOV     AX,"  " ; 2020h
29895
29896                                   ;hkn; DEVFCB is in DOSDATA
29897 00004E35 BF[5305]                    MOV     DI,DEVFCB+8             ; Point to extent field
29898
29899                                   ;   Fill dir_ext  BUGBUG - use ERRNZs for this stuff!
29900
29901 00004E38 AB                          STOSW
29902 00004E39 AA                          STOSB                           ; Blank out extent field
29903                                      ;mov   al,40h
29904 00004E3A B040                        MOV     AL,attr_device
29905
29906                                   ;   Fill Dir_attr
29907
29908 00004E3C AA                          STOSB                           ; Set attribute field
29909 00004E3D 31C0                        XOR     AX,AX
29910 00004E3F B90A00                      MOV     CX,10
29911
29912                                   ; Fill dir_pad
29913
29914 00004E42 F3AB                        REP     STOSW                   ; Fill rest with zeros
29915 00004E44 E819BD                      call    DATE16
29916
29917                                   ;hkn; DEVFCB is in DOSDATA
29918 00004E47 BF[6105]                    MOV     DI,DEVFCB+dir_entry.dir_time ; 09/08/2018
29919 00004E4A 92                          XCHG    AX,DX
29920
29921                                   ; Fill dir_time
29922
29923 00004E4B AB                          STOSW
29924 00004E4C 92                          XCHG    AX,DX
29925
29926                                   ; Fill dir_date
29927
29928 00004E4D AB                          STOSW
29929 00004E4E 89FE                        MOV     SI,DI                   ; SI points to dir_first field
29930 00004E50 A1[9A05]                    MOV     AX,[DEVPT]
29931
29932                                   ; Fill dir_first
29933
29934 00004E53 AB                          STOSW                           ; Dir_first points to device
29935 00004E54 A1[9C05]                    MOV     AX,[DEVPT+2]
29936                                   ;
29937                                   ; Fill dir_size_l
29938                                   ;
29939 00004E57 AB                          STOSW
29940 00004E58 88FC                        MOV     AH,BH                   ; Put device atts in AH
29941
29942                                   ;hkn; DEVFCB is in DOSDATA
29943 00004E5A BB[4B05]                    MOV     BX,DEVFCB
29944 00004E5D 30C0                        XOR     AL,AL                   ; Set zero, clear carry
29945 00004E5F C3                          retn
29946
29947                                   ;Break      <ValidateCDS - given a CDS, validate the media and the current directory>
29948                                   ;-----------------------------------------------------------------------------
29949                                   ;
29950                                   ;   ValidateCDS - Get current CDS. Splice it. Call FatReadCDS to check
29951                                   ;   media. If media has been changed, do DOS_Chdir to validate path.
29952                                   ;   If invalid, reset original CDS to root.
29953                                   ;
29954                                   ;   Inputs: ThisCDS points to CDS of interest
29955                                   ;           SS:DI points to temp buffer
29956                                   ;   Outputs:       The current directory string is validated on the appropriate
29957                                   ;                drive
29958                                   ;           ThisDPB changed
29959                                   ;           ES:DI point to CDS
29960                                   ;           Carry set if error (currently user FAILed to I 24)
29961                                   ;   Registers modified: all
29962                                   ;-----------------------------------------------------------------------------
29963
29964                                       ; 21/11/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
29965                                       ; DOSCODE:819Bh (MSDOS 5.0, MSDOS.SYS)
29966
29967                                       ; 16/02/2024 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMDOS.COM)
29968                                       ; DOSCODE:8F01h (PCDOS 7.1, IBMDOS.COM)
29969
29970                                       ; (MSDOS 6.22 MSDOS.SYS - DOSCODE:81D6h)
29971                                       ; (Windows ME IO.SYS - BIOSCODE:84CBh)
29972
29973                                   ValidateCDS:
29974                                       ; 19/05/2019 - Retro DOS v4.0
29975                                       ; 28/07/2018 - Retro DOS v3.0
29976
29977                                      %define  Temp   [bp-2]  ; word
29978                                      %define  SaveCDS       [bp-6]  ; dword
29979                                      %define  SaveCDSL      [bp-6]  ; word
29980                                      %define  SaveCDSH      [bp-4]  ; word
29981
29982                                       ;Enter
29983 00004E60 55                          push    bp
29984 00004E61 89E5                        mov     bp,sp
29985 00004E63 83EC06                      sub     sp,6
29986
29987 00004E66 897EFE                      MOV     Temp,DI
29988
29989                                   ;hkn; SS override
29990 00004E69 36C536[A205]                LDS     SI,[SS:THISCDS]
29991 00004E6E 8976FA                      MOV     SaveCDSL,SI
29992 00004E71 8C5EFC                      MOV     SaveCDSH,DS
29993                                       ;EnterCrit critDisk
```

```
29994 00004E74 E86BCA                      call    ECritDisk
29995                                      ; 21/11/2022
29996                                      ;test   byte [SI+curdir.flags+1],80h
29997                                      ;test   word [si+67],8000h
29998                                      ; 17/12/2022
29999                                      ;test   byte [SI+68],80h
30000 00004E77 F6444480                    test    byte [SI+curdir.flags+1],(curdir_isnet>>8)
30001                                      ;TEST   word [SI+curdir.flags],curdir_isnet  ; Clears carry
30002 00004E7B 7403                        JZ      short _DoSplice
30003 00004E7D E9A300                      JMP     FatFail
30004                                  _DoSplice:
30005 00004E80 30D2                        XOR     DL,DL
30006 00004E82 368616[4C03]                XCHG    DL,[SS:NoSetDir]
30007
30008                                  ;hkn; SS is DOSDATA
30009                                      ;Context ES
30010 00004E87 16                          push    ss
30011 00004E88 07                          pop     es
30012                                      ;Invoke FStrcpy
30013 00004E89 E834C9                      call    FStrCpy
30014 00004E8C 8B76FE                      MOV     SI,Temp
30015
30016                                  ;hkn; SS is DOSDATA
30017                                      ;Context DS
30018 00004E8F 16                          push    ss
30019 00004E90 1F                          pop     ds
30020                                      ;Invoke Splice
30021 00004E91 E84A30                      call    Splice
30022
30023                                   ;hkn; SS is DOSDATA
30024                                      ;Context DS                     ;   FatReadCDS (ThisCDS);
30025 00004E94 16                          push    ss
30026 00004E95 1F                          pop     ds
30027 00004E96 8816[4C03]                  MOV     [NoSetDir],DL
30028 00004E9A C43E[A205]                  LES     DI,[THISCDS]
30029                                      ;SAVE   <BP>
30030 00004E9E 55                          push    bp
30031                                      ;Invoke FATREAD_CDS
30032 00004E9F E8B916                      call    FATREAD_CDS
30033                                      ;RESTORE <BP>
30034 00004EA2 5D                          pop     bp
30035 00004EA3 727E                        JC      short FatFail
30036
30037 00004EA5 C536[A205]                  LDS     SI,[THISCDS]            ;   if (ThisCDS->ID == -1) {
30038
30039                                      ;;;
30040                                      ; 16/02/2024 (PCDOS 7.1 IBMDOS.COM)
30041                                      ;cmp    word [si+4Bh],0FFFFh
30042 00004EA9 837C4BFF                    cmp     word [si+curdir.ID+2],-1
30043 00004EAD 7566                        jnz     short RestoreCDS
30044                                      ;;;
30045
30046                                      ;cmp    word [si+73],-1
30047 00004EAF 837C49FF                    CMP     word [SI+curdir.ID],-1
30048 00004EB3 7560                        JNZ     short RestoreCDS
30049
30050                                  ;hkn; SS is DOSDATA
30051                                      ;Context ES
30052 00004EB5 16                          push    ss
30053 00004EB6 07                          pop     es
30054
30055                                  ;hkn; SS override
30056                                      ;SAVE   <wfp_Start>             ;       t = wfp_Start;
30057 00004EB7 36FF36[B205]                push    word [SS:WFP_START]
30058                                      ;cmp    si,[bp-6]
30059 00004EBC 3B76FA                      CMP     SI,SaveCDSL             ; if not spliced
30060 00004EBF 750B                        JNZ     short DoChdir
30061                                      ;mov    di,[bp-2]
30062 00004EC1 8B7EFE                      MOV     DI,Temp
30063
30064                                  ;hkn; SS override
30065 00004EC4 36893E[B205]                MOV     [SS:WFP_START],DI       ;       wfp_start = d;
30066                                      ;Invoke FStrCpy                 ;       strcpy (d, ThisCDS->Text);
30067 00004EC9 E8F4C8                      call    FStrCpy
30068                                  DoChdir:
30069                                  ;hkn; SS is DOSDATA
30070                                      ;Context DS
30071 00004ECC 16                          push    ss
30072 00004ECD 1F                          pop     ds
30073                                      ;SAVE   <<WORD PTR SAttrib>,BP> ;       c = DOSChDir ();
30074 00004ECE FF36[6D05]                  push    word [SATTRIB]
30075 00004ED2 55                          push    bp
30076                                      ;Invoke DOS_ChDir
30077 00004ED3 E85CEB                      call    DOS_CHDIR
30078                                      ;RESTORE <BP,BX,wfp_start>      ;       wfp_Start = t;
30079 00004ED6 5D                          pop     bp
30080 00004ED7 5B                          pop     bx
30081 00004ED8 8F06[B205]                  pop     word [WFP_START]
30082 00004EDC 881E[6D05]                  MOV     [SATTRIB],BL
30083                                      ;;;
30084                                      ; 16/02/2024 (PCDOS 7.1 IBMDOS.COM)
30085 00004EE0 8B3E[DC0A]                  mov     di,[DIRSTART_HW]
30086                                      ;;;
30087 00004EE4 C576FA                      LDS     SI,SaveCDS
30088 00004EE7 7311                        JNC     short SetCluster        ;       if (c == -1) {
30089
30090                                  ;hkn; SS override for THISCDS
30091 00004EE9 368936[A205]                MOV     [SS:THISCDS],SI                 ;           ThisCDS = TmpCDS;
30092 00004EEE 368C1E[A405]                MOV     [SS:THISCDS+2],DS
30093 00004EF3 31C9                        XOR     CX,CX                   ;           TmpCDS->text[3] = c = 0;
30094                                      ;;;
30095                                      ; 16/02/2024 (PCDOS 7.1 IBMDOS.COM)
30096 00004EF5 31FF                        xor     di,di
30097                                      ;;;
30098 00004EF7 884C03                      MOV     [SI+3],CL               ;           }
30099                                  SetCluster:
30100                                      ; 16/02/2024
30101                                      ;;mov   word [si+73],0FFFFh
30102                                      ;MOV    word [SI+curdir.ID],-1;     TmpCDS->ID = -1;
30103                                      ;
30104 00004EFA 36C536[A205]                LDS     SI,[SS:THISCDS]                 ;       ThisCDS->ID = c;
30105                                      ; 21/11/2022
30106                                      ;test   byte [si+curdir.flags+1],20h
30107                                      ; 19/05/2019
30108                                      ; MSDOS 6.0
30109                                      ; 17/12/2022
30110                                      ;test   byte [si+68],20h
30111 00004EFF F6444420                    test    byte [SI+curdir.flags+1],(curdir_splice>>8)
30112                                      ;;test  word [si+67],2000h
30113                                      ;TEST   word [SI+curdir.flags],curdir_splice ;AN000;;MS. for Join and Subst
30114 00004F03 7405                        JZ      short _setdirclus                   ;AN000;;MS.
30115 00004F05 B9FFFF                      MOV     CX,-1  ; 0FFFFh                     ;AN000;;MS.
30116                                      ;;;
30117                                      ; 16/02/2024 (PCDOS 7.1 IBMDOS.COM)
```

```
30118 00004F08 89CF                     mov     di,cx
30119                                   ;;;
30120                                 _setdirclus:
30121                                   ;;;
30122                                   ; 16/02/2024 (PCDOS 7.1 IBMDOS.COM)
30123 00004F0A 36893E[DC0A]             mov     [ss:DIRSTART_HW],di
30124                                   ;mov    [si+4Bh],di
30125 00004F0F 897C4B                   mov     [si+curdir.ID+2],di
30126                                   ;;;
30127                                   ;mov    [si+73],cx
30128 00004F12 894C49                   MOV     [SI+curdir.ID],CX       ;       }
30129                                 RestoreCDS:
30130 00004F15 C47EFA                   LES     DI,SaveCDS
30131 00004F18 36893E[A205]             MOV     [SS:THISCDS],DI
30132 00004F1D 368C06[A405]             MOV     [SS:THISCDS+2],ES
30133 00004F22 F8                       CLC
30134                                 FatFail:
30135                                   ;LeaveCrit critDisk
30136 00004F23 E8E9C9                   call    LCritDisk
30137
30138                                   ;les    di,[bp-6]
30139 00004F26 C47EFA                   LES     DI,SaveCDS
30140                                   ;Leave
30141 00004F29 89EC                     mov     sp,bp
30142 00004F2B 5D                       pop     bp
30143 00004F2C C3                       retn
30144
30145                                 ; 28/07/2018 - Retro DOS v3.0
30146                                 ; IBMDOS.COM (MSDOS 3.3, 1987) - offset 43BDh
30147
30148                                 ;Break      <CheckThisDevice - Check for being a device>
30149                                 ;--------------------------------------------------------------------------
30150                                 ;
30151                                 ;   CheckThisDevice - Examine the area at DS:SI to see if there is a valid
30152                                 ;   device specified. We will return carry if there is a device present.
30153                                 ;   The forms of devices we will recognize are:
30154                                 ;
30155                                 ;   [path]device
30156                                 ;
30157                                 ;   Note that the drive letter has *already* been removed. All other forms
30158                                 ;   are not considered to be devices. If such a device is found we change
30159                                 ;   the source pointer to point to the device component.
30160                                 ;
30161                                 ;   Inputs: ES is DOSDATA
30162                                 ;           DS:SI contains name
30163                                 ;   Outputs:       ES is DOSDATA
30164                                 ;           DS:SI point to name or device
30165                                 ;           Carry flag set if device was found
30166                                 ;           Carry flag reset otherwise
30167                                 ;   Registers Modified: all except ES:DI, DS
30168                                 ;--------------------------------------------------------------------------
30169
30170                                 CheckThisDevice:
30171 00004F2D 57                       push    di
30172 00004F2E 56                       push    si
30173 00004F2F 89F7                     MOV     DI,SI
30174
30175                                 ; Check for presence of \dev\ (Dam multiplan!)
30176
30177 00004F31 8A04                     MOV     AL,[SI]
30178 00004F33 E8B40E                   call    PATHCHRCMP              ; is it a path char?
30179 00004F36 7517                     JNZ     short ParseDev          ; no, go attempt to parse device
30180 00004F38 46                       INC     SI                      ; simulate LODSB
30181
30182                                 ; We have the leading path separator. Look for DEV part.
30183
30184 00004F39 AD                       LODSW
30185 00004F3A 0D2020                   OR      AX,2020h
30186 00004F3D 3D6465                   cmp     ax,"de"
30187                                   ;CMP    AX,"e"<< 8 + "d"
30188 00004F40 752D                     JNZ     short NotDevice                 ; not "de", assume not device
30189 00004F42 AC                       LODSB
30190 00004F43 0C20                     OR      AL,20h
30191 00004F45 3C76                     CMP     AL,"v"                    ; Not "v", assume not device
30192 00004F47 7526                     JNZ     short NotDevice
30193 00004F49 AC                       LODSB
30194 00004F4A E89D0E                   call    PATHCHRCMP              ; do we have the last path separator?
30195 00004F4D 7520                     JNZ     short NotDevice                 ; no. go for it.
30196
30197                                 ; DS:SI now points to a potential drive. Preserve them as NameTrans advances
30198                                 ; SI and DevName may destroy DS.
30199
30200                                 ParseDev:
30201 00004F4F 1E                       push    ds
30202 00004F50 56                       push    si                      ; preserve the source pointer
30203 00004F51 E8DA0D                   call    NameTrans               ; advance DS:SI
30204 00004F54 803C00                   CMP     BYTE [SI],0             ; parse entire string?
30205 00004F57 F9                       STC                             ; simulate a Carry return from DevName
30206 00004F58 750B                     JNZ     short SkipSearch        ; no parse. simulate a file return.
30207
30208                                 ;hkn; SS is DOSDATA
30209 00004F5A 16                       push    ss
30210 00004F5B 1F                       pop     ds
30211
30212                                 ; M026 - start - fix ported from ROMDOS2 for bug # 2849
30213                                 ;
30214                                 ; SR;
30215                                 ; We have to set Attrib before invoking DevName. Otherwise, the value from
30216                                 ; a previous DOS call is used and DevName thinks it is not a device if the
30217                                 ; old call set the volume attribute bit.
30218
30219 00004F5C A0[6D05]                   mov     al,[SATTRIB]
30220 00004F5F A2[6B05]                 mov     [ATTRIB],al             ;set Attrib for DevName
30221
30222                                 ; M026 - end
30223
30224 00004F62 E86FFE                   call    DEVNAME
30225
30226                                 SkipSearch:
30227 00004F65 5E                       pop     si
30228 00004F66 1F                       pop     ds
30229
30230                                 ; SI points to the beginning of the potential device. If we have a device
30231                                 ; then we do not change SI. If we have a file, then we reset SI back to the
30232                                 ; original value. At this point Carry set indicates FILE.
30233
30234                                 CheckReturn:
30235 00004F67 5F                       pop     di                      ; get original SI
30236 00004F68 7302                     JNC     short Check_Done        ; if device then do not reset pointer
30237 00004F6A 89FE                     MOV     SI,DI
30238                                 Check_Done:
30239 00004F6C 5F                       pop     di
30240 00004F6D F5                       CMC                             ; invert carry. Carry => device
30241 00004F6E C3                       retn
```

```
30242                                   NotDevice:
30243 00004F6F F9                           STC
30244 00004F70 EBF5                         JMP     short CheckReturn
30245
30246                                   ;BREAK <LookupPath - call fastopen to get dir entry info>
30247                                   ;--------------------------------------------------------------------------
30248                                   ;
30249                                   ; Procedure Name : LookupPath
30250                                   ;
30251                                   ; Output  DS:SI -> path name,
30252                                   ;     ES:DI -> dir entry info buffer
30253                                   ;     ES:CX -> extended dir info buffer
30254                                   ;
30255                                   ;     carry flag clear : tables pointed by ES:DI and ES:CX are filled by
30256                                   ;                        FastOpen, DS:SI points to char just one after
30257                                   ;                        the last char of path name which is fully or
30258                                   ;                        partially found in FastOPen
30259                                   ;     carry flag set : FastOpen not in memory or path name not found
30260                                   ;
30261                                   ;--------------------------------------------------------------------------
30262
30263                                       ; 21/02/2024 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMDOS.COM)
30264                                       ; PCDOS 7.1 IBMDOS.COM - DOSCODE:9015h
30265
30266                                       ; (MSDOS 6.22 MSDOS.SYS - DOSCODE:82D9h)
30267
30268                                   LookupPath:
30269                                   ;   PUSH    AX
30270
30271                                   ;hkn; SS override
30272                                       ;test   byte [ss:FastOpenFlg],1
30273 00004F72 36F606[4611]01             TEST    byte [ss:FastOpenFlg],FastOpen_Set  ; flg is set in DOSOPEN
30274 00004F78 7515                       JNZ     short FASTINST                      ; and this routine is
30275                                   NOLOOK:
30276 00004F7A EB11                       JMP     NOLOOKUP                            ; executed once
30277
30278                                       ; 21/02/2024
30279                                   NOTFOUND:
30280 00004F7C 83F8FF                     CMP     AX,-1                               ; not in memory ?
30281 00004F7F 7506                       JNZ     short Partial_Success               ; yes, in memory
30282 00004F81 36C606[4611]00             MOV     byte [SS:FastOpenFlg],0                 ; no more fastopen
30283                                   Partial_Success:
30284                                       ;and    byte [SS:FastOpenFlg],0FBh
30285 00004F87 368026[4611]FB             AND     byte [SS:FastOpenFlg],Special_Fill_Reset
30286                                   NOLOOKUP:
30287                                   ;   POP     AX
30288 00004F8D F9                         STC
30289 00004F8E C3                         RETN
30290
30291                                   FASTINST:
30292                                   ;hkn; SS override
30293                                       ;test   byte [ss:FastOpenFlg],8
30294 00004F8F 36F606[4611]08             TEST    byte [ss:FastOpenFlg],No_Lookup         ; no more lookup?
30295 00004F95 75E3                       JNZ     short NOLOOK                        ; yes
30296
30297 00004F97 BB[3C11]                   MOV     BX,FastOpenTable                    ; get fastopen related tab
30298
30299                                   ;hkn; SS override
30300 00004F9A 368B36[B205]               MOV     SI,[SS:WFP_START]                   ; si points to path name
30301 00004F9F BF[7C0D]                   MOV     DI,Dir_Info_Buff
30302 00004FA2 B9[4711]                   MOV     CX,FastOpen_Ext_Info
30303 00004FA5 B001                       MOV     AL,FONC_Look_up                     ; al = 1
30304 00004FA7 1E                         PUSH    DS
30305 00004FA8 07                         POP     ES
30306
30307                                   ;hkn; SS override
30308                                       ;call   far [bx+2]
30309 00004FA9 FF5F02                     CALL    far [BX+fastopen_entry.name_caching] ;call fastopen
30310 00004FAC 72CE                       JC      short NOTFOUND                      ; fastopen not in memory
30311
30312 00004FAE 8D5CFE                     LEA     BX,[SI-2]
30313
30314                                   ;hkn; SS override
30315 00004FB1 363B1E[B205]               CMP     BX,[SS:WFP_START]                   ; path found ?
30316 00004FB6 74C4                       JZ      short NOTFOUND                      ; no
30317
30318                                       ; 19/05/2019 - Retro DOS v4.0
30319
30320                                       ; MSDOS 6.0                             ; fully or partially found
30321 00004FB8 803C00                     CMP     BYTE [SI],0 ; * ; 21/02/2024      ;AN000;FO.
30322 00004FBB 752D                       JNZ     short parfnd                        ;AN000;FO.; partiallyfound
30323 00004FBD 51                         PUSH    CX                                  ;AN000;FO.; is attribute matched ?
30324
30325                                   ;hkn; SS override for attrib/sattrib
30326 00004FBE 368A0E[6B05]               MOV     CL,[ss:ATTRIB]                      ;AN000;FO.;
30327 00004FC3 368A2E[6D05]               MOV     CH,[ss:SATTRIB]                         ;AN000;FO.; attrib=sattrib
30328 00004FC8 36882E[6B05]               MOV     [ss:ATTRIB],CH                      ;AN000;FO.;
30329                                     ;mov    ch,[es:di+0Bh]
30330 00004FCD 268A6D0B                   MOV     CH,[ES:DI+dir_entry.dir_attr]       ;AN000;FO.;
30331 00004FD1 E8F3FD                     call    MatchAttributes                     ;AN000;FO.;
30332                                   ;;; MOV     [ss:ATTRIB],CL                      ;AN001;FO.; restore attrib
30333 00004FD4 59                         POP     CX                                  ;AN000;FO.;
30334 00004FD5 75B6                       JNZ     short NOLOOKUP                      ;AN000;FO.; not matched
30335
30336                                       ; 21/02/2024 - Retro DOS v5.0
30337                                       ; PCDOS 7.1 IBMDOS.COM
30338                                       ;;;
30339 00004FD7 36803E[4C03]00             cmp     byte [ss:NoSetDir],0
30340 00004FDD 740B                       jz      short parfnd
30341                                     ;
30342                                     ;cmp    byte [si],0  ; * ; byte [si] = 0
30343                                     ;jnz    short parfnd
30344                                     ;
30345                                     ;test   byte [es:di+0Bh],10h
30346 00004FDF 26F6450B10                 test    byte [es:di+dir_entry.dir_attr],attr_directory
30347 00004FE4 7404                       jz      short parfnd
30348 00004FE6 89CB                       mov     bx,cx
30349 00004FE8 EB17                       jmp     short parfnd2
30350                                   parfnd:
30351 00004FEA 89CB                       mov     bx,cx
30352                                     ;mov    ax,[bx+0Bh]
30353 00004FEC 8B470B                     mov     ax,[bx+FEI.dirstart]
30354 00004FEF 36A3[C205]                 mov     [ss:DIRSTART],ax
30355                                     ; 27/06/2024
30356                                     ;mov    ax,[bx+7]
30357 00004FF3 8B4707                     mov     ax,[bx+FEI.clusnum+2]
30358 00004FF6 36A3[DE0A]                 mov     [ss:CLUSNUM_HW],ax
30359                                     ;mov    ax,[bx+5]
30360 00004FFA 8B4705                     mov     ax,[bx+FEI.clusnum]
30361 00004FFD 36A3[BC05]                 mov     [ss:CLUSNUM],ax
30362                                   parfnd2:
30363 00005001 368936[9C0D]               mov     [ss:Next_Element_Start],si
30364                                     ;mov    ax,[bx+9]
30365 00005006 8B4709                     mov     ax,[bx+FEI.lastent]
```

```
30366 00005009 36A3[4803]                mov     [ss:LASTENT],ax
30367                                    ;;;
30368
30369                                  ; 21/02/2024
30370                                  %if 0
30371                                  parfnd:
30372                                  ;hkn; SS override
30373                                      MOV     [SS:Next_Element_Start],SI       ; save si
30374                                      MOV     BX,CX
30375                                      ; MSDOS 6.0
30376                                      ;mov    ax,[bx+7]
30377                                      MOV     AX,[BX+FEI.lastent]              ;AN000;;FO. restore lastentry
30378                                  ;hkn; SS override for LASTENT, DIRSTART, CLUSNUM
30379                                      MOV     [SS:LASTENT],AX                          ;AN000;;FO.
30380                                      MOV     AX,[BX+FEI.dirstart]             ;AN001;;FO. restore dirstart
30381                                      MOV     [SS:DIRSTART],AX                 ;AN001;;FO.
30382                                      ; MSDOS 3.3 (& MSDOS 6.0)
30383                                      ;;mov   ax,[bx+3] ; MSDOS 3.3
30384                                      ;mov    ax,[bx+5] ; MSDOS 6.0
30385                                      MOV     AX,[BX+FEI.clusnum]              ; restore next cluster num
30386                                      MOV     [SS:CLUSNUM],AX                          ;
30387                                  %endif
30388
30389 0000500D 06                        PUSH    ES                               ; save ES
30390                                  ;hkn; SS override
30391 0000500E 36C41E[8A05]              LES     BX,[SS:THISDPB]                          ; put drive id
30392 00005013 268A27                    mov     ah,[ES:BX] ; 15/08/2018
30393                                    ;MOV    AH,[ES:BX+DPB.DRIVE]             ; in AH for DOOPEN
30394 00005016 07                        POP     ES                               ; pop ES
30395                                  ;SR;
30396                                  ; We cannot have a root dir if we have come here. So, we zero out CurBuf to
30397                                  ;indicate it is not a root dir
30398
30399 00005017 36C706[E205]0000           mov     word [SS:CURBUF],0               ; indicate not root dir
30400 0000501E 368C06[E405]              MOV     WORD [SS:CURBUF+2],ES            ; [curbuf+2].bx points to
30401 00005023 89FB                      MOV     BX,DI                            ; start of entry
30402                                    ;lea    si,[di+1Ah]
30403 00005025 8D751A                    LEA     SI,[DI+dir_entry.dir_first]      ; [curbuf+2]:si points to
30404                                                                             ; dir_first field in the
30405                                                                             ; dir entry
30406                                  ;hkn; SS override for FastOpenFlg
30407                                      ;or     byte [ss:FastOpenFlg],12h ; 29/12/2022
30408 00005028 36800E[4611]12            OR      byte [SS:FastOpenFlg],Lookup_Success+Set_For_Search
30409                                  ;   POP     AX
30410 0000502E C3                        RETN
30411
30412                                  ;BREAK <InsertPath - call fastopen to insert dir entry info>
30413                                  ;--------------------------------------------------------------------------
30414                                  ;
30415                                  ; Procedure Name : InsertPath
30416                                  ; Input:  FastOpen_Set flag set when from DOSOPEN otherwise 0
30417                                  ;     Lookup_Success flag set when got dir entry info from FASTOPEN
30418                                  ;     DS = DOSDATA
30419                                  ; Output: FastOPen_Ext_Info is set and path dir info is inserted
30420                                  ;
30421                                  ;--------------------------------------------------------------------------
30422
30423                                      ; 21/11/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
30424                                      ; 21/02/2024 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMDOS.COM)
30425
30426                                  InsertPath:
30427 0000502F 9C                        PUSHF
30428                                  ;hkn; SS override for FastOpenFlag
30429                                      ;test   byte [SS:FastOpenFlg], 1
30430 00005030 36F606[4611]01            TEST    byte [SS:FastOpenFlg],FastOpen_Set ;only DOSOPEN can take advantage of
30431 00005036 747C                      JZ      short GET_NEXT_ELEMENT           ; the FastOpen
30432                                    ;test   byte [ss:FastOpenFlg],2
30433 00005038 36F606[4611]02            TEST    byte [SS:FastOpenFlg],Lookup_Success ; Lookup just happened
30434 0000503E 740D                      JZ      short INSERT_DIR_INFO            ; no
30435                                    ;and    byte [ss:FastOpenFlg],0FDh
30436 00005040 368026[4611]FD            AND     byte [SS:FastOpenFlg],Lookup_Reset  ; we got dir info from fastopen so
30437 00005046 368B3E[9C0D]              MOV     DI,[SS:Next_Element_Start]       ; no need to insert it again
30438 0000504B EB61                      JMP     short GET_NEXT2
30439
30440                                  INSERT_DIR_INFO:                         ; save registers
30441 0000504D 1E                        PUSH    DS
30442 0000504E 06                        PUSH    ES
30443 0000504F 53                        PUSH    BX
30444 00005050 56                        PUSH    SI
30445 00005051 57                        PUSH    DI
30446 00005052 51                        PUSH    CX
30447 00005053 50                        PUSH    AX
30448
30449                                  ;hkn; SS override
30450 00005054 36C53E[E205]              LDS     DI,[SS:CURBUF]                   ; DS:DI -> buffer header
30451 00005059 BE[4711]                  MOV     SI,FastOpen_Ext_Info
30452
30453                                  ; 21/02/2024
30454                                  %if 0
30455                                      ;mov    ax,[di+6]
30456                                      MOV     AX,[DI+BUFFINFO.buf_sector]   ; get directory sector
30457                                      ; MSDOS 6.0
30458                                      ;mov    [ss:si+1],ax
30459                                      MOV     [SS:SI+FEI.dirsec],AX        ;AN000; >32mb save dir sector
30460                                      ; 19/05/2019 - Retro DOS v4.0
30461                                      MOV     AX,[DI+BUFFINFO.buf_sector+2] ;AN000; >32mb
30462
30463                                  ;hkn; SS is DOSDATA
30464                                      push    ss
30465                                      pop     ds
30466                                      ; MSDOS 3.3
30467                                      ;;mov   [si+1],ax
30468                                      ;MOV    [SI+FEI.dirsec],AX
30469                                      ; MSDOS 6.0
30470                                      ;mov    [si+3],ax
30471                                      MOV     [SI+FEI.dirsec+2],AX         ;AN000;>32mb save high dir sector
30472                                  %else
30473                                      ;lds    ax,[di+6]
30474 0000505C C54506                    lds     ax,[di+BUFFINFO.buf_sector]   ; get directory sector
30475                                    ;mov    [ss:si+1],ax
30476                                    ; 27/06/2024
30477                                    ;mov    [ss:si+FEI.dirsec],ax
30478                                    ;
30479                                    ;mov    [ss:si+3],ax
30480 0000505F 368C5C03                  mov     [ss:si+FEI.dirsec+2],ds
30481 00005063 16                        push    ss
30482 00005064 1F                        pop     ds
30483                                    ; 27/06/2024
30484 00005065 894401                    mov     [si+FEI.dirsec],ax
30485                                  %endif
30486
30487                                      ; 21/02/2024 (PCDOS 7.1 IBMDOS.COM)
30488                                      ;;;
30489 00005068 A1[DE0A]                  mov     ax,[CLUSNUM_HW]
```

```
30490                                   ;mov    [si+7],ax
30491 0000506B 894407                   mov     [si+FEI.clusnum+2],ax
30492                                   ;;;
30493
30494                                   ; MSDOS 3.3 (& MSDOS 6.0)
30495 0000506E A1[BC05]                 MOV     AX,[CLUSNUM]            ; save next cluster number
30496                                   ;mov    [si+5],ax ; MSDOS 6.0
30497                                   ;;mov   [si+3],ax ; MSDOS 3.3
30498 00005071 894405                   MOV     [SI+FEI.clusnum],AX
30499                                   ; MSDOS 6.0
30500 00005074 A1[4803]                 MOV     AX,[LASTENT]            ;AN000;FO. save lastentry for search first
30501                                   ;;mov   [si+7],ax
30502                                   ;mov    [si+9],ax ; PCDOS 7.1 ; 21/02/2024
30503 00005077 894409                   MOV     [SI+FEI.lastent],AX     ;AN000;FO.
30504 0000507A A1[C205]                 MOV     AX,[DIRSTART]           ;AN001;FO. save  for search first
30505                                   ;;mov   [si+9],ax
30506                                   ;mov    [si+11],ax ; PCDOS 7.1 ; 21/02/2024
30507 0000507D 89440B                   MOV     [SI+FEI.dirstart],AX    ;AN001;FO.
30508
30509                                   ; MSDOS 3.3 (& MSDOS 6.0)
30510 00005080 89D8                     MOV     AX,BX
30511                                   ;;;add  di,16  ; MSDOS 3.3
30512                                   ;;add   di,20  ; MSDOS 6.0
30513                                   ;add    di,24  ; PCDOS 7.1 ; 21/02/2024
30514 00005082 83C718                   ADD     DI,BUFINSIZ             ; DS:DI -> start of data in buffer
30515 00005085 29F8                     SUB     AX,DI                   ; AX=BX relative to start of sector
30516                                   ;mov    cl,32
30517 00005087 B120                     MOV     CL,dir_entry.size
30518 00005089 F6F1                     DIV     CL
30519                                   ;MOV    [SI+FEI.dirpos],AL      ; save directory entry # in buffer
30520 0000508B 8804                     mov     [si],al
30521
30522 0000508D 1E                       PUSH    DS
30523 0000508E 07                       POP     ES
30524
30525 0000508F 8E1E[E405]               MOV     DS,[CURBUF+2]
30526 00005093 89DF                     MOV     DI,BX                   ; DS:DI -> dir entry info
30527                                       ;cmp    word [di+1Ah],0
30528 00005095 837D1A00                     CMP     word [DI+dir_entry.dir_first],0
30529                                                                   ; never insert info when file is empty
30530 00005099 740C                         JZ      short SKIP_INSERT       ; e.g. newly created file
30531
30532                                       ; 21/02/2024 - Erdogan Tan
30533                                       ; Note: PCDOS 7.1 IBMDOS.COM code doesn't check high word
30534                                       ;       of the 1st cluster here
30535                                       ;;;
30536                                       ; 21/02/2024 - Retro DOS v5.0
30537                                       ;jnz    short dont_skip_insert
30538                                       ;;cmp   word [di+14h],0
30539                                       ;cmp    word [di+dir_entry.dir_fclus_hi],0
30540                                       ;jz     short SKIP_INSERT
30541                                   ;dont_skip_insert: ; Retro DOS v5.0
30542                                       ;;;
30543
30544 0000509B 56                           PUSH    SI                      ; ES:BX -> extended info
30545 0000509C 5B                           POP     BX
30546
30547                                       ;mov    al,2
30548 0000509D B002                         MOV     AL,FONC_insert          ; call fastopen insert operation
30549 0000509F BE[3C11]                     MOV     SI,FastOpenTable
30550                                       ;call   far [es:si+2]   ; call dword ptr es:[si+2] ; 29/12/2022
30551                                       ; 07/12/2022
30552 000050A2 26FF5C02                     CALL    far [ES:SI+fastopen_entry.name_caching]
30553
30554 000050A6 F8                           CLC
30555                                   SKIP_INSERT:
30556 000050A7 58                           POP     AX
30557 000050A8 59                           POP     CX                      ; restore registers
30558 000050A9 5F                           POP     DI
30559 000050AA 5E                           POP     SI
30560 000050AB 5B                           POP     BX
30561 000050AC 07                           POP     ES
30562 000050AD 1F                           POP     DS
30563                                   GET_NEXT2:
30564                                       ;or     [ss:FastOpenFlg],8
30565 000050AE 36800E[4611]08               OR      byte [SS:FastOpenFlg],No_Lookup
30566                                                                   ; we got dir info from fastopen so
30567                                   GET_NEXT_ELEMENT:
30568 000050B4 9D                           POPF
30569 000050B5 C3                           RETN
30570
30571                                   ;=============================================================================
30572                                   ; DEV.ASM (MSDOS 6.0, 1991)
30573                                   ;=============================================================================
30574                                   ; 17/07/2018 - Retro DOS v3.0
30575                                   ; 30/04/2019 - Retro DOS v4.0
30576                                   ; 21/02/2024 - Retro DOS v5.0
30577
30578                                   ;** Misc Routines to do 1-12 low level I/O and call devices
30579
30580                                   ; Offset 12B8h of IBMDOS.COM (MSDOS 3.3), 1987
30581
30582                                   ;DOSCODE:8401h (MSDOS 6.21 & 6.22, MSDOS.SYS) ; 21/02/2024
30583                                   ;BIOSCODE:8608h (Windows ME, IO.SYS) ; 21/02/2024
30584
30585                                   ;DOSCODE:9163h (PCDOS 7.1, IBMDOS.COM) ; 21/02/2024
30586
30587                                   ;Public DEV001S, DEV001E          ; Pathgen labels
30588                                   ;DEV001s:
30589                                   ;           length of packets
30590 000050B6 160E160D0D0E             LenTab:      DB     DRDWRHL, DRDNDHL, DRDWRHL, DSTATHL, DFLSHL, DRDNDHL
30591                                   ; 21/02/2024
30592                                   ;;LenTab: db       22,14,22,13,15,14 ; MSDOS 5.0-6.22 & Windows ME
30593                                   ;LenTab: db 22,14,22,13,13,14 ; PCDOS 7.1
30594
30595                                   ;   Error Function
30596
30597                                   CmdTab:
30598 000050BC 8604                         DB      86h, DEVRD      ; 0 input
30599 000050BE 8605                         DB      86h, DEVRDND    ; 1 input status
30600 000050C0 8708                         DB      87h, DEVWRT     ; 2 output
30601 000050C2 870A                         DB      87h, DEVOST     ; 3 output status
30602 000050C4 8607                         DB      86h, DEVIFL     ; 4 input flush
30603 000050C6 8605                         DB      86h, DEVRDND    ; 5 input status with system WAIT
30604
30605                                   ; Offset 12BEh of IBMDOS.COM (MSDOS 3.3), 1987
30606
30607                                   ;CmdTab:
30608                                   ;   db      86h, 4
30609                                   ;   db      86h, 5
30610                                   ;   db      87h, 8
30611                                   ;   db      87h, 10
30612                                   ;   db      86h, 7
30613                                   ;   db      86h, 5
```

```
30614
30615                                 ;DEV001E:
30616
30617                                 ; 30/04/2019 - Retro DOS v4.0
30618                                 ; DOSCODE:8413h (MSDOS 6.21, MSDOS.SYS)
30619
30620                                 ;Break      <IOFUNC -- DO FUNCTION 1-12 I/O>
30621                                 ;-----------------------------------------------------------------------------
30622                                 ;
30623                                 ; Procedure Name : IOFUNC
30624                                 ;
30625                                 ; Inputs:
30626                                 ;   DS:SI Points to SFT
30627                                 ;   AH is function code
30628                                 ;           = 0 Input
30629                                 ;           = 1 Input Status
30630                                 ;           = 2 Output
30631                                 ;           = 3 Output Status
30632                                 ;           = 4 Flush
30633                                 ;           = 5 Input Status - System WAIT invoked for K09 if no char
30634                                 ;                              present.
30635                                 ;   AL = character if output
30636                                 ; Function:
30637                                 ;   Perform indicated I/O to device or file
30638                                 ; Outputs:
30639                                 ;   AL is character if input
30640                                 ;   If a status call
30641                                 ;           zero set if not ready
30642                                 ;           zero reset if ready (character in AL for input status)
30643                                 ; For regular files:
30644                                 ;   Input Status
30645                                 ;           Gets character but restores position
30646                                 ;           Zero set on EOF
30647                                 ;   Input
30648                                 ;           Gets character advances position
30649                                 ;           Returns ^Z on EOF
30650                                 ;   Output Status
30651                                 ;           Always ready
30652                                 ; AX altered, all other registers preserved
30653                                 ;-----------------------------------------------------------------------------
30654
30655                                 ; 21/11/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
30656                                 ; DOSCODE:83D8h (MSDOS 5.0, MSDOS.SYS)
30657
30658                                 ; 21/02/2024 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMDOS.COM)
30659                                 ; DOSCODE:9175h (PCDOS 7.1, IBMDOS.COM)
30660
30661                                 ; DOSCODE:8413h (MSDOS 6.22, MSDOS.SYS)
30662                                 ; BIOSCODE:861Ah (Windows ME, IO.SYS)
30663
30664                                 IOFUNC:
30665 000050C8 368C16[8C03]             MOV     [SS:IOXAD+2],SS             ; SS override for IOXAD, IOSCNT,
30666                                                                         ; DEVIOBUF
30667 000050CD 36C706[8A03][BC03]        MOV     WORD [SS:IOXAD],DEVIOBUF
30668 000050D4 36C706[8E03]0100          MOV     WORD [SS:IOSCNT],1
30669 000050DB 36A3[BC03]                MOV     WORD [SS:DEVIOBUF],AX
30670                                    ;test   byte [si+6],80h
30671                                    ;TEST   word [SI+SF_ENTRY.sf_flags],sf_isnet ; 8000h
30672 000050DF F6440680                  test    byte [SI+SF_ENTRY.sf_flags+1],(sf_isnet>>8)
30673 000050E3 7403                      JZ      short IOTO22            ;AN000;
30674 000050E5 E9A300                    JMP     IOTOFILE                ;AN000;
30675                                 IOTO22:
30676                                    ;test   word [si+5],80h
30677                                    ;TEST   word [SI+SF_ENTRY.sf_flags],devid_device
30678 000050E8 F6440580                  test    byte [SI+SF_ENTRY.sf_flags],devid_device
30679 000050EC 7503                      JNZ     short IOTO33            ;AN000;
30680 000050EE E99A00                    JMP     IOTOFILE                ;AN000;
30681                                 IOTO33:
30682 000050F1 06                        push    es ; * (MSDOS 6.21)
30683 000050F2 E863B3                    call    save_world
30684 000050F5 8CDA                      MOV     DX,DS
30685 000050F7 8CD3                      MOV     BX,SS
30686 000050F9 8EDB                      MOV     DS,BX
30687 000050FB 8EC3                      MOV     ES,BX
30688 000050FD 31DB                      XOR     BX,BX
30689 000050FF 80FC05                    cmp     ah,5                ; system wait enabled?
30690 00005102 7502                      jnz     short _no_sys_wait
30691                                    ; 21/11/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
30692                                    ; 16/12/2022
30693                                    ;or     bh,04h
30694                                    ;;or    bx,0400H            ; Set bit 10 in status word for driver
30695                                                                ; It is up to device driver to carry out
30696                                                                ; appropriate action.
30697                                    ; 04/07/2024
30698 00005104 B704                      mov     bh,4
30699                                 _no_sys_wait:
30700 00005106 891E[7F03]                MOV     [IOCALL_REQSTAT],BX
30701 0000510A 31DB                      XOR     BX,BX
30702 0000510C 881E[8903]                MOV     [IOMED],BL
30703
30704 00005110 88E3                      MOV     BL,AH                   ; get function
30705 00005112 2E8AA7[B650]              MOV     AH,[cs:BX+LenTab]
30706 00005117 D1E3                      SHL     BX,1
30707 00005119 2E8B8F[BC50]              MOV     CX,[cs:BX+CmdTab]
30708 0000511E BB[7C03]                  MOV     BX,IOCALL ; DOSDATA:037Ch
30709 00005121 8826[7C03]                MOV     [IOCALL_REQLEN],AH
30710 00005125 882E[7E03]                MOV     [IOCALL_REQFUNC],CH
30711
30712 00005129 8EDA                      MOV     DS,DX
30713 0000512B E86401                    CALL    DEVIOCALL
30714 0000512E 368B3E[7F03]              MOV     DI,[SS:IOCALL_REQSTAT]; SS override
30715 00005133 21FF                      and     di,di
30716 00005135 7833                      js      short DevErr
30717                                 OKDevIO:
30718 00005137 8CD0                      MOV     AX,SS
30719 00005139 8ED8                      MOV     DS,AX
30720
30721                                    ;cmp    ch,5
30722 0000513B 80FD05                    CMP     CH,DEVRDND
30723 0000513E 7506                      JNZ     short DNODRD
30724 00005140 A0[8903]                  MOV     AL,[IORCHR]
30725 00005143 A2[BC03]                  MOV     [DEVIOBUF],AL
30726
30727                                 DNODRD:
30728 00005146 8A26[8003]                MOV     AH,[IOCALL_REQSTAT+1]
30729 0000514A F6D4                      NOT     AH                      ; Zero = busy, not zero = ready
30730                                    ;and    ah,2
30731 0000514C 80E402                    AND     AH,STBUI>>8
30732
30733                                 QuickReturn:                            ;AN000; 2/13/KK
30734 0000514F E8EFB2                    call    restore_world
30735 00005152 07                        pop     es ; * (MSDOS 6.21)
30736
30737                                    ; SR;
```

```
                                    ; We return ax = -1 if the user failed on I24. This is the case if
                                    ; IoStatFail = -1 (set after return from the I24)

                                    ; MSDOS 6.0
00005153 9C                         pushf
00005154 36A0[8300]                 mov     al,[ss:IoStatFail]    ;assume fail error
00005158 98                         cbw                           ;sign extend to word

                                ; 21/02/2024
                                %if 0
                                    ; PCDOS 7.1 IBMDOS.COM
                                    ; cbw
                                    inc     ax              ; 21/02/2024 - Erdogan Tan
                                                            ; this may be a BUG
                                                            ; MSDOS 6.22 MSDOS.SYS & Win ME IO.SYS code is
                                                            ;  cbw
                                                            ;  cmp  ax,-1
                                                            ;  jne  short not_fail_ret
                                                            ;  inc  byte [ss:IoStatFail]
                                                            ;  popf
                                                            ;  retn
                                    jnz     short not_fail_ret
                                    dec     ax
                                    jnz     short not_fail_ret ; jns ?
                                %else
                                    ;;cbw
                                    ;cmp    ax,-1           ; (MSDOS 6.22 & Windows ME)
                                    ;jne    short not_fail_ret
                                    ; 27/06/2024
00005159 3CFF                       cmp     al,0FFh ; -1
0000515B 7507                       jne     short not_fail_ret
                                %endif

0000515D 36FE06[8300]               inc     byte [ss:IoStatFail]
00005162 9D                         popf
00005163 C3                         retn

                                not_fail_ret:
00005164 36A1[BC03]                 mov     ax,[ss:DEVIOBUF]      ;ss override
00005168 9D                         popf
00005169 C3                         retn

                                DevErr:
0000516A 88CC                       MOV     AH,CL
0000516C E8B40E                     call    CHARHARD
0000516F 3C01                       CMP     AL,1
00005171 7507                       JNZ     short NO_RETRY
00005173 E8CBB2                     call    restore_world
                                    ; 12/05/2019
00005176 07                         pop     es ; * (MSDOS 6.21)
00005177 E94EFF                     JMP     IOFUNC ; 10/08/2018

                                NO_RETRY:
                                    ; Know user must have wanted Ignore OR Fail. Make sure device shows ready
                                    ; ready so that DOS doesn't get caught in a status loop when user
                                    ; simply wants to ignore the error.
                                    ;
                                    ; SR; If fail wanted by user set ax to special value (ax = -1). This
                                    ; should be checked by the caller on return

                                                                    ; SS override
0000517A 368026[8003]FD             and     byte [SS:IOCALL_REQSTAT+1],0FDh
                                    ;AND    BYTE [SS:IOCALL_REQSTAT+1],~(STBUI>>8)

                                    ; SR;
                                    ; Check if user failed

                                    ; MSDOS 6.0
00005180 3C03                       cmp     al,3
00005182 7505                       jnz     short not_fail
00005184 36FE0E[8300]               dec     byte [ss:IoStatFail]  ;set flag indicating fail on I24
                                not_fail:
00005189 EBAC                       JMP     short OKDevIO

                                IOTOFILE:
0000518B 08E4                       OR      AH,AH
0000518D 7421                       JZ      short IOIN
0000518F FECC                       DEC     AH
00005191 7405                       JZ      short IOIST
00005193 FECC                       DEC     AH
00005195 7411                       JZ      short IOUT
                                IOUT_retn:  ; 18/12/2022
00005197 C3                         retn                          ; NON ZERO FLAG FOR OUTPUT STATUS
                                IOIST:
                                    ;push   word [si+15h]
00005198 FF7415                     PUSH    WORD [SI+SF_ENTRY.sf_position]   ; Save position
                                    ;push   word [si+17h]
0000519B FF7417                     PUSH    WORD [SI+SF_ENTRY.sf_position+2]
0000519E E80F00                     CALL    IOIN
                                    ;pop    word [si+17h]
000051A1 8F4417                     POP     WORD [SI+SF_ENTRY.sf_position+2] ; Restore position
                                    ;pop    word [si+15h]
000051A4 8F4415                     POP     WORD [SI+SF_ENTRY.sf_position]
000051A7 C3                         retn
                                IOUT:
000051A8 E82500                     CALL    SETXADDR
000051AB E8CAEB                     call    DOS_WRITE
                                    ;CALL   RESTXADDR       ; If you change this into a jmp don't
                                    ; 18/12/2022
000051AE EB4F                       jmp     RESTXADDR
                                ;IOUT_retn:
                                    ;retn                   ; come crying to me when things don't
                                                            ; work ARR
                                IOIN:
000051B0 E81D00                     CALL    SETXADDR
                                                                ; SS override for DOS34_FLAG
                                    ;OR     word [SS:DOS34_FLAG],Disable_EOF_I24 ;AN000;
                                    ;or     word [ss:DOS34_FLAG],40h
                                    ; 21/11/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
                                    ; 16/12/2022
000051B3 36800E[1106]40             or      byte [ss:DOS34_FLAG],40h
000051B9 E8BBE9                     CALL    DOS_READ
                                    ;AND    word [SS:DOS34_FLAG],NO_Disable_EOF_I24 ;AN000;
                                    ;and    word [SS:DOS34_FLAG],0FFBFh
                                    ; 21/11/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
                                    ; 16/12/2022
000051BC 368026[1106]BF             and     byte [SS:DOS34_FLAG],0BFh ; 07/12/2022
000051C2 09C9                       OR      CX,CX                   ; Check EOF
000051C4 E83800                     CALL    RESTXADDR
                                                                    ; SS override
000051C7 36A0[BC03]                 MOV     AL,[SS:DEVIOBUF]        ; Get byte from trans addr
000051CB 75CA                       jnz     short IOUT_retn
000051CD B01A                       MOV     AL,1AH                  ; ^Z if no bytes
000051CF C3                         retn
```

```
30862
30863                                   SETXADDR:
30864                                                                   ; SS override
30865 000051D0 368F06[6C03]                 POP     WORD [SS:CALLSCNT]  ; Return address
30866
30867 000051D5 06                           push    es ; * (MSDOS 6.21)
30868
30869 000051D6 E87FB2                       call    save_world
30870                                                                   ; SS override for DMAADD and THISSFT
30871                                       ; 24/09/2023
30872                                       ;PUSH   WORD [SS:DMAADD]    ; Save Disk trans addr
30873                                       ;PUSH   WORD [SS:DMAADD+2]
30874 000051D9 368C1E[A005]                 MOV     [SS:THISSFT+2],DS
30875
30876                                   ; 22/02/2024
30877                                   %if 0
30878                                       push    ss
30879                                       pop     ds
30880
30881                                       ; 24/09/2023
30882                                       push    word [DMAADD]
30883                                       push    word [DMAADD+2]
30884
30885                                       MOV     [THISSFT],SI        ; Finish setting SFT pointer
30886                                       MOV     CX,[IOXAD+2]
30887                                       MOV     [DMAADD+2],CX
30888                                       MOV     CX,[IOXAD]
30889                                       MOV     [DMAADD],CX         ; Set byte trans addr
30890                                   %else
30891                                       ; 22/02/2024
30892                                       ; PCDOS 7.1 IBMDOS.COM
30893
30894 000051DE 36C50E[2C03]                 lds     cx,[ss:DMAADD]      ; Save Disk transfer address
30895 000051E3 51                           push    cx
30896 000051E4 1E                           push    ds
30897 000051E5 36C50E[8A03]                 lds     cx,[ss:IOXAD]       ; Set byte trans address
30898 000051EA 368C1E[2E03]                 mov     [ss:DMAADD+2],ds
30899 000051EF 16                           push    ss
30900 000051F0 1F                           pop     ds
30901 000051F1 890E[2C03]                   mov     [DMAADD],cx
30902 000051F5 8936[9E05]                   mov     [THISSFT],si
30903                                   %endif
30904 000051F9 8B0E[8E03]                   MOV     CX,[IOSCNT]         ; ioscnt specifies length of buffer
30905 000051FD EB10                         JMP     SHORT RESTRET       ; RETURN ADDRESS
30906
30907                                   RESTXADDR:
30908 000051FF 8F06[6C03]                   POP     WORD [CALLSCNT]             ; Return address
30909 00005203 8F06[2E03]                   POP     WORD [DMAADD+2]             ; Restore Disk trans addr
30910 00005207 8F06[2C03]                   POP     WORD [DMAADD]
30911
30912 0000520B E833B2                       call    restore_world
30913
30914 0000520E 07                           pop     es ; * (MSDOS 6.21)
30915                                                                   ; SS override
30916                                   RESTRET:
30917 0000520F 36FF26[6C03]                 JMP     WORD [SS:CALLSCNT]  ; Return address
30918
30919                                   ; DOSCODE:8569h (MSDOS 6.21, MSDOS.SYS)
30920                                   ; 21/11/2022
30921                                   ; DOSCODE:852Eh (MSDOS 5.0, MSDOS.SYS)
30922
30923                                   ; 22/02/2024 - Retro DOS v5.0
30924                                   ; DOSCODE:92CAh (PCDOS 7.1, IBMDOS.COM)
30925
30926                                   ;Break <DEV_OPEN_SFT, DEV_CLOSE_SFT - OPEN or CLOSE A DEVICE>
30927
30928                                   ;--------------------------------------------------------------------------
30929                                   ;** Dev_Open_SFT - Open the Device for an SFT
30930                                   ;
30931                                   ;   Dev_Open_SFT issues an open call to the device associated with
30932                                   ;   the SFT.
30933                                   ;
30934                                   ;   ENTRY   (ES:DI) = SFT
30935                                   ;   EXIT    none
30936                                   ;   USES    all
30937                                   ;--------------------------------------------------------------------------
30938
30939                                   DEV_OPEN_SFT:
30940 00005214 06                           push    es ; * (MSDOS 6.21)
30941 00005215 E840B2                       call    save_world
30942                                       ;mov    al,0Dh
30943 00005218 B00D                         MOV     AL,DEVOPN
30944 0000521A EB06                         JMP     SHORT DO_OPCLS
30945
30946                                   ;--------------------------------------------------------------------------
30947                                   ; Procedure Name : DEV_CLOSE_SFT
30948                                   ;
30949                                   ; Inputs:
30950                                   ;   ES:DI Points to SFT
30951                                   ; Function:
30952                                   ;   Issue a CLOSE call to the correct device
30953                                   ; Outputs:
30954                                   ;   None
30955                                   ; ALL preserved
30956                                   ;--------------------------------------------------------------------------
30957
30958                                   DEV_CLOSE_SFT:
30959 0000521C 06                           push    es ; * (MSDOS 6.21)
30960 0000521D E838B2                       call    save_world
30961                                       ;mov    al,0Eh
30962 00005220 B00E                         MOV     AL,DEVCLS
30963
30964                                       ; Main entry for device open and close. AL contains the function
30965                                       ; requested. Subtlety: if Sharing is NOT loaded then we do NOT issue
30966                                       ; open/close to block devices. This allows networks to function but
30967                                       ; does NOT hang up with bogus change-line code.
30968
30969                                       ;entry  DO_OPCLS
30970                                   DO_OPCLS:
30971                                       ; Is the SFT for the net? If so, no action necessary.
30972
30973                                       ; MSDOS 6.0
30974                                       ;test   word [es:di+5],8000h
30975                                       ;TEST   word [ES:DI+SF_ENTRY.sf_flags],sf_isnet
30976 00005222 26F6450680                   test    byte [es:di+SF_ENTRY.sf_flags+1],(sf_isnet>>8)
30977 00005227 7564                         jnz     short OPCLS_DONE        ; NOP on net SFTs
30978 00005229 30E4                         XOR     AH,AH                   ; Unit
30979                                       ;test   byte [es:di+5],80h
30980 0000522B 26F6450580                   TEST    byte [ES:DI+SF_ENTRY.sf_flags],devid_device
30981                                       ;les    di,[es:di+7]
30982 00005230 26C47D07                     LES     DI,[ES:DI+SF_ENTRY.sf_devptr] ; Get DPB or device
30983 00005234 7511                         JNZ     short GOT_DEV_ADDR
30984
30985                                       ; We are about to call device open/close on a block driver. If no
```

```
                                    ; sharing then just short circuit to done.
                                    ; MSDOS 6.0
                                                            ; SS override
00005236 36803E[0303]01             CMP     byte [ss:fShare],1      ;AN010; /NC or no SHARE
0000523C 764F                       JBE     short OPCLS_DONE        ;AN010; yes

                                ; 22/02/2024
                                %if 0
                                    ; MSDOS 3.3 (& MSDOS 6.0)
                                    ;mov    ah,[es:di+1]
                                    MOV     AH,[ES:DI+DPB.UNIT]     ; (ah) = unit
                                    mov     cl,[es:di]
                                    ;MOV    CL,[ES:DI+DPB.DRIVE]    ; (cl) = drive
                                %else
                                    ; 22/02/2024 - Retro DOS v5.0
                                    ;mov    cx,[es:di+DPB.DRIVE]
0000523E 268B0D                     mov     cx,[es:di]
00005241 88EC                       mov     ah,ch                   ; AH = unit
                                                                    ; CL = drive
                                %endif
                                    ;;les   di,[es:di+12h] ; MSDOS 3.3
                                    ;les    di,[es:di+13h] ; MSDOS 6.0
00005243 26C47D13                   LES     DI,[ES:DI+DPB.DRIVER_ADDR] ; Get device
                                GOT_DEV_ADDR:                           ; ES:DI -> device
                                    ;test   word [es:di+4],800h
                                    ;TEST   word [ES:DI+SYSDEV.ATT],DEVOPCL
00005247 26F6450508                 test    byte [ES:DI+SYSDEV.ATT+1],(DEVOPCL>>8)
0000524C 743F                       JZ      short OPCLS_DONE        ; Device can't
0000524E 06                         PUSH    ES
0000524F 1F                         POP     DS
00005250 89FE                       MOV     SI,DI                   ; DS:SI -> device

                                OPCLS_RETRY:
                                    ;Context ES
00005252 16                         push    ss
00005253 07                         pop     es
                                                            ; DEVCALL is in DOSDATA
00005254 BF[5A03]                   MOV     DI,DEVCALL

00005257 89FB                       MOV     BX,DI
00005259 50                         PUSH    AX
                                    ;mov    al,13
0000525A B00D                       MOV     AL,DOPCLHL
0000525C AA                         STOSB                           ; Length
0000525D 58                         POP     AX

0000525E 86E0                       XCHG    AH,AL
                                    ;STOSB                          ; Unit
                                    ; 22/02/2024 (PCDOS 7.1 IBMDOS.COM)
00005260 AB                         stosw                           ; Unit, Command
00005261 86E0                       XCHG    AH,AL
                                    ;STOSB                          ; Command

00005263 26C7050000                 MOV     WORD [ES:DI],0          ; Status
00005268 50                         PUSH    AX                      ; Save Unit,Command
                                    ;invoke DEVIOCALL2
00005269 E82900                     call    DEVIOCALL2

                                    ;mov    di,[es:bx+3]
0000526C 268B7F03                   MOV     DI,[ES:BX+SRHEAD.REQSTAT]
                                    ;test   di,8000h
                                    ;jz     short OPCLS_DONEP
00005270 21FF                       and     di,di
00005272 7918                       jns     short OPCLS_DONEP       ; No error
                                    ; 21/11/2022
                                    ;test   word [si+4],8000h
                                    ;TEST   word [SI+SYSDEV.ATT],DEVTYP
                                    ;test   word [si+5],80h
00005274 F6440580                   test    byte [SI+SYSDEV.ATT+1],(DEVTYP>>8)
00005278 7404                       JZ      short BLKDEV
0000527A B486                       MOV     AH,86H                  ; Read error in data, Char dev
0000527C EB04                       JMP     SHORT HRDERR
                                BLKDEV:
0000527E 88C8                       MOV     AL,CL                   ; Drive # in AL
00005280 B406                       MOV     AH,6                    ; Read error in data, Blk dev
                                HRDERR:
                                    ;invoke CHARHARD
00005282 E89E0D                     call    CHARHARD
00005285 3C01                       cmp     al,1
00005287 7503                       jne     short OPCLS_DONEP       ; IGNORE or FAIL
                                                                    ;  Note that FAIL is essentually IGNORED
00005289 58                         POP     AX                      ; Get back Unit, Command
0000528A EBC6                       JMP     short OPCLS_RETRY
                                OPCLS_DONEP:
0000528C 58                         POP     AX                      ; Clean stack
                                OPCLS_DONE:
0000528D E8B1B1                     call    restore_world
00005290 07                         pop     es ; * (MSDOS 6.21)
00005291 C3                         retn

                                ; 30/04/2019 - Retro DOS v4.0
                                ; DOSCODE:85EAh (MSDOS 6.21, MSDOS.SYS)

                                ; 22/11/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
                                ; DOSCODE:85AFh (MSDOS 5.0, MSDOS.SYS)

                                ; 22/02/2024 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMDOS.COM)
                                ; DOSCODE:9348h (PCDOS 7.1, IBMDOS.COM)

                                ;Break      <DEVIOCALL, DEVIOCALL2 - CALL A DEVICE>
                                ;-----------------------------------------------------------------------------
                                ;** DevIoCall  - Call Device
                                ;
                                ;   ENTRY   DS:SI Points to device SFT
                                ;           ES:BX Points to request data
                                ;   EXIT    DS:SI -> Device driver
                                ;   USES    DS:SI,AX
                                ;-----------------------------------------------------------------------------
                                ;** DevIoCall2 - Call Device
                                ;
                                ;   ENTRY   DS:SI Points to DPB
                                ;           ES:BX Points to request data
                                ;   EXIT    DS:SI -> Device driver
                                ;   USES    DS:SI,AX
                                ;-----------------------------------------------------------------------------

                                DEVIOCALL:
                                                                    ; SS override for CALLSSEC,
                                    ;lds    si,[si+7]               ; CALLNEWSC, HIGH_SECTOR & CALLDEVAD
00005292 C57407                     LDS     SI,[SI+SF_ENTRY.sf_devptr]

                                    ;entry  DEVIOCALL2
                                DEVIOCALL2:
```

```
                                    ;EnterCrit critDevice
00005295 E888C6                     call    ECritDevice
                                    ; MSDOS 6.0
                                    ;TEST   word [SI+SYSDEV.ATT],DEVTYP ;AN000; >32mb block device ?
                                    ;test   byte [si+5],80h
00005298 F6440580                   test    byte [si+SYSDEV.ATT+1],(DEVTYP>>8)
0000529C 7540                       jnz     short chardev2          ;AN000; >32mb no
                                    ; 16/12/2022
                                    ; 22/11/2022
0000529E 268A4702                   mov     al,[ES:BX+SRHEAD.REQFUNC] ; [es:bx+2]
000052A2 3C04                       cmp     al,DEVRD        ; 4
000052A4 7408                       je      short chkext
000052A6 3C08                       cmp     al,DEVWRT       ; 8
000052A8 7404                       je      short chkext
000052AA 3C09                       cmp     al,DEVWRTV      ; 9
000052AC 7530                       jne     short chardev2
                                    ; 16/12/2022
                                    ; 22/11/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
                                    ;;cmp   byte [es:bx+2],4
                                    ;CMP    byte [ES:BX+SRHEAD.REQFUNC],DEVRD  ;AN000; >32mb read ?
                                    ;JZ     short chkext            ;AN000; >32mb   yes
                                    ;;cmp   byte [es:bx+2],8
                                    ;CMP    byte [ES:BX+SRHEAD.REQFUNC],DEVWRT ;AN000; >32mb write ?
                                    ;JZ     short chkext            ;AN000; >32mb   yes
                                    ;;cmp   byte [es:bx+2],9
                                    ;CMP    byte [ES:BX+SRHEAD.REQFUNC],DEVWRTV
                                    ;                               ;AN000; >32mb write/verify ?
                                    ;JNZ    short chardev2          ;AN000; >32mb no
                                chkext:
                                ; 22/02/2024 - Retro DOS v5.0 (PCDOS 7.1)
                                %if 0
                                    CALL    RW_SC                   ;AN000;LB. use secondary cache if there
                                    JC      short dev_exit          ;AN000;LB. done
                                %endif
                                    ;test   byte [si+4],2
000052AE F6440402                   TEST    byte [SI+SYSDEV.ATT],EXTDRVR ;AN000;>32mb extended driver?
000052B2 741A                       JZ      short chksector                 ;AN000;>32mb   no
000052B4 26800708                   ADD     BYTE [ES:BX],8          ;AN000;>32mb   make length to 30
                                    ;MOV    AX,[SS:CALLSSEC]        ;AN000;>32mb
                                    ;MOV    word [SS:CALLSSEC],-1 ;AN000;>32mb   old sector  =-1
                                    ; 22/02/2024
000052B8 B8FFFF                     mov     ax,-1 ; 0FFFFh
000052BB 368706[6E03]               xchg    ax,[ss:CALLSSEC]
000052C0 36A3[7403]                 MOV     [SS:CALLNEWSC],AX       ;AN000;>32mb   new sector  =
000052C4 36A1[0706]                 MOV     AX,[SS:HIGH_SECTOR]     ;AN000; >32mb  low sector,high sector
000052C8 36A3[7603]                 MOV     [SS:CALLNEWSC+2],AX     ;AN000; >32mb
000052CC EB10                       JMP     short chardev2          ;AN000; >32mb
                                chksector:                          ;AN000; >32mb
000052CE 36833E[0706]00             CMP     word [SS:HIGH_SECTOR],0         ;AN000; >32mb   if >32mb
000052D4 7408                       JZ      short chardev2          ;AN000; >32mb   then fake error
                                    ;mov    word [es:bx+3],8107h
000052D6 26C747030781               MOV     word [ES:BX+SRHEAD.REQSTAT],STERR+STDON+error_I24_not_DOS_disk
                                                                    ;AN000; >32mb
000052DC EB27                       JMP     SHORT dev_exit          ;AN000; >32mb
                                chardev2:                           ;AN000;
                                    ; As above only DS:SI points to device header on entry, and DS:SI is
                                    ; preserved
                                    ;;;
                                    ; 22/02/2024 - Retro DOS v5.0 (PCDOS 7.1 IBMDOS.COM)
000052DE 36FE06[B912]               inc     byte [ss:DEVIO_IN_PROGRESS] ; lock (deviocall in progress)
                                    ;;;
                                    ;mov    ax,[si+6]
000052E3 8B4406                     MOV     AX,[SI+SYSDEV.STRAT]
000052E6 36A3[7803]                 MOV     [SS:CALLDEVAD],AX
000052EA 368C1E[7A03]               MOV     [SS:CALLDEVAD+2],DS
000052EF 36FF1E[7803]               CALL    far [SS:CALLDEVAD]
                                    ;mov    ax,[si+8]
000052F4 8B4408                     MOV     AX,[SI+SYSDEV.INT]
000052F7 36A3[7803]                 MOV     [SS:CALLDEVAD],AX
000052FB 36FF1E[7803]               CALL    far [SS:CALLDEVAD]
                                    ;;;
                                    ; 22/02/2024 - Retro DOS v5.0 (PCDOS 7.1 IBMDOS.COM)
00005300 36FE0E[B912]               dec     byte [ss:DEVIO_IN_PROGRESS] ; unlock (deviocall completed)
                                    ;;;
                                ; 22/02/2024 - Retro DOS v5.0 (PCDOS 7.1)
                                %if 0
                                    ; MSDOS 6.0
                                    CALL    VIRREAD                 ;AN000;LB. move data from SC to buffer
                                    JC      short chardev2          ;AN000;LB. bad sector or exceeds max sec
                                %endif
                                dev_exit:
                                    ;LeaveCrit critDevice
                                    ;call   LCritDevice
                                    ;retn
                                    ; 18/12/2022
00005305 E929C6                     jmp     LCritDevice
                                ; DOSCODE:8669h (MSDOS 6.21, MSDOS.SYS)
                                ; 22/11/2022
                                ; DOSCODE:862Eh (MSDOS 5.0, MSDOS.SYS)
                                ;Break      <SETREAD, SETWRITE -- SET UP HEADER BLOCK>
                                ;-----------------------------------------------------------------------------
                                ;
                                ; Procedure Name : SETREAD, SETWRITE
                                ;
                                ; Inputs:
                                ;   DS:BX = Transfer Address
                                ;   CX = Record Count
                                ;   DX = Starting Record
                                ;   AH = Media Byte
                                ;   AL = Unit Code
                                ; Function:
                                ;   Set up the device call header at DEVCALL
                                ; Output:
                                ;   ES:BX Points to DEVCALL
                                ; No other registers effected
                                ;
```

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

                                SETREAD_XJ:
                                    ;;;
                                    ; 07/02/2024 - Retro DOS v5.0
00005308 89FB                       mov     bx,di
0000530A EB07                       jmp     short SETREAD_X
                                    ;;;

                                SETREAD_XT:
                                    ;;;
                                    ; 07/02/2024 - Retro DOS v5.0
0000530C BB[B603]                   mov     bx,TIMEBUF
0000530F 53                         push    bx
                                SETREAD_XTC:
00005310 B90600                     mov     cx,6
                                    ;;;
                                SETREAD_X:
                                    ;;;
                                    ; 06/02/2024 - Retro DOS v5.0
00005313 31C0                       xor     ax,ax
                                    ;mov    dx,ax ; 0
00005315 99                         cwd
                                    ;;;

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

                                SETREAD:
00005316 57                         PUSH    DI
00005317 51                         PUSH    CX
00005318 50                         PUSH    AX
00005319 B104                       MOV     CL,DEVRD ; mov cl,4
                                SETCALLHEAD:
0000531B B016                       MOV     AL,DRDWRHL ; mov al,16h
0000531D 16                         PUSH    SS
0000531E 07                         POP     ES
                                                                    ; DEVCALL is in DOSDATA
0000531F BF[5A03]                   MOV     DI,DEVCALL

00005322 AA                         STOSB                           ; length
00005323 58                         POP     AX                      ;
00005324 AA                         STOSB                           ; Unit
00005325 50                         PUSH    AX
00005326 88C8                       MOV     AL,CL
00005328 AA                         STOSB                           ; Command code
00005329 31C0                       XOR     AX,AX
0000532B AB                         STOSW                           ; Status
0000532C 83C708                     ADD     DI,8                    ; Skip link fields
0000532F 58                         POP     AX
00005330 86E0                       XCHG    AH,AL
00005332 AA                         STOSB                           ; Media byte
00005333 86C4                       XCHG    AL,AH
00005335 50                         PUSH    AX
00005336 89D8                       MOV     AX,BX
00005338 AB                         STOSW

00005339 8CD8                       MOV     AX,DS
0000533B AB                         STOSW                           ; Transfer addr

0000533C 59                         POP     CX                      ; Real AX
0000533D 58                         POP     AX                      ; Real CX
0000533E AB                         STOSW                           ; Count

0000533F 92                         XCHG    AX,DX                   ; AX=Real DX, DX=real CX, CX=real AX
00005340 AB                         STOSW                           ; Start
00005341 91                         XCHG    AX,CX
00005342 87D1                       XCHG    DX,CX
00005344 5F                         POP     DI
                                                                    ; DEVCALL is in DOSDATA
00005345 BB[5A03]                   MOV     BX,DEVCALL
00005348 C3                         retn

                                    ;entry  SETWRITE
                                SETWRITE:

                                ; Inputs:
                                ;   DS:BX = Transfer Address
                                ;   CX = Record Count
                                ;   DX = Starting Record
                                ;   AH = Media Byte
                                ;   AL = Unit Code
                                ; Function:
                                ;   Set up the device call header at DEVCALL
                                ; Output:
                                ;   ES:BX Points to DEVCALL
                                ; No other registers effected

00005349 57                         PUSH    DI
0000534A 51                         PUSH    CX
0000534B 50                         PUSH    AX
0000534C B108                       MOV     CL,DEVWRT ; mov cl,8
0000534E 36020E[FF02]               ADD     CL,[SS:VERFLG]          ; SS override
00005353 EBC6                       JMP     SHORT SETCALLHEAD

                                ; 22/02/2024 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMDOS.COM)
                                %if 0       ; SC

                                ; 30/04/2019 - Retro DOS v4.0
                                ; DOSCODE:86A8h (MSDOS 6.21, MSDOS.SYS)
                                ; 22/11/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
                                ; DOSCODE:866Dh (MSDOS 5.0, MSDOS.SYS)

                                ;Break      <RW_SC -- Read Write Secondary Cache>
                                ;--------------------------------------------------------------------------
                                ;
                                ; Procedure Name : RW_SC
                                ;
                                ; Inputs:
                                ;    [SC_CACHE_COUNT]= secondary cache count
                                ;    [SC_STATUS]= SC validity status
                                ;    [SEQ_SECTOR]= last sector read
                                ; Function:
                                ;   Read from or write through secondary cache
                                ; Output:
                                ;   ES:BX Points to DEVCALL
                                ;   carry clear, I/O is not done
                                ;               [SC_FLAG]=1 if continuos sectors will be read
                                ;   carry set, I/O is done
                                ;
                                ;--------------------------------------------------------------------------

                                RW_SC:
                                    ; SS override for all variables used.
```

```
31358                                   CMP     word [ss:SC_CACHE_COUNT],0  ;AN000;LB. secondary cache exists?
31359                                   JZ      short scexit4               ;AN000;LB. no, do nothing
31360                                   CMP     word [ss:CALLSCNT],1        ;AN000;LB. sector count = 1 (buffer I/O)
31361                                   JNZ     short scexit4               ;AN000;LB. no, do nothing
31362                                   PUSH    CX                          ;AN000;LB.
31363                                   PUSH    DX                          ;AN000;LB. yes
31364                                   PUSH    DS                          ;AN000;LB. save registers
31365                                   PUSH    SI                          ;AN000;LB.
31366                                   PUSH    ES                          ;AN000;LB.
31367                                   PUSH    DI                          ;AN000;LB.
31368
31369                                   MOV     DX,[ss:CALLSSEC]            ;AN000;LB. starting sector
31370                                   CMP     BYTE [ss:DEVCALL_REQFUNC],DEVRD ;AN000;LB. read ?
31371                                   JZ      short doread                ;AN000;LB. yes
31372                                   CALL    INVALIDATE_SC               ;AN000;LB. invalidate SC
31373                                   JMP     scexit2                     ;AN000;LB. back to normal
31374                                scexit4:                               ;AN000;
31375                                   CLC                                 ;AN000;LB. I/O not done yet
31376                                   retn                                ;AN000;LB.
31377                                doread:                                ;AN000;
31378                                   CALL    SC2BUF                      ;AN000;LB. check if in SC
31379                                   JC      short readSC                ;AN000;LB.
31380                                   MOV     word [ss:DEVCALL_REQSTAT],STDON ;AN000;LB. fake done and ok
31381                                   STC                                 ;AN000;LB. set carry
31382                                   JMP     short saveseq               ;AN000;LB. save seq. sector #
31383                                readSC:                                ;AN000;
31384                                   MOV     AX,[ss:HIGH_SECTOR]         ;AN000;LB. subtract sector num from
31385                                   MOV     CX,[ss:CALLSSEC]            ;AN000;LB. saved sequential sector
31386                                   SUB     CX,[ss:SEQ_SECTOR]          ;AN000;LB. number
31387                                   SBB     AX,[ss:SEQ_SECTOR+2]        ;AN000;LB.
31388                                   ; 24/09/2023
31389                                   ;CMP    AX,0                        ;AN000;LB. greater than 64K
31390                                   JNZ     short saveseq2              ;AN000;LB. yes,save seq. sector #
31391                                chklow:
31392                                   CMP     CX,1                        ;AN000;LB. <= 1
31393                                   JA      short saveseq2              ;AN000;LB. no, not sequential
31394                                   MOV     word [ss:SC_STATUS],-1      ;AN000;LB. presume all SC valid
31395                                   MOV     AX,[ss:SC_CACHE_COUNT]      ;AN000;LB. yes, sequential
31396                                   MOV     [ss:CALLSCNT],AX            ;AN000;LB. read continuous sectors
31397                                readsr:
31398                                   MOV     AX,[ss:CALLXAD+2]           ;AN000;LB. save buffer addr
31399                                   MOV     [ss:TEMP_VAR2],AX           ;AN000;LB. in temp vars
31400                                   MOV     AX,[ss:CALLXAD]                   ;AN000;LB.
31401                                   MOV     [ss:TEMP_VAR],AX            ;AN000;LB.
31402
31403                                   MOV     AX,[ss:SC_CACHE_PTR]        ;AN000;LB. use SC cache addr as
31404                                   MOV     [ss:CALLXAD],AX                   ;AN000;LB. transfer addr
31405                                   MOV     AX,[ss:SC_CACHE_PTR+2]      ;AN000;LB.
31406                                   MOV     [ss:CALLXAD+2],AX           ;AN000;LB.
31407                                   MOV     byte [ss:SC_FLAG],1         ;AN000;LB. flag it for later;
31408                                   MOV     AL,[ss:SC_DRIVE]            ;AN000;LB. current drive
31409                                   MOV     [ss:CurSC_DRIVE],AL         ;AN000;LB. set current drive
31410                                   MOV     AX,[ss:CALLSSEC]            ;AN000;LB. current sector
31411                                   MOV     [ss:CurSC_SECTOR],AX        ;AN000;LB. set current sector
31412                                   MOV     AX,[ss:HIGH_SECTOR]         ;AN000;LB.
31413                                   MOV     [ss:CurSC_SECTOR+2],AX      ;AN000;LB.
31414                                saveseq2:                              ;AN000;
31415                                   CLC                                 ;AN000;LB. clear carry
31416                                saveseq:                               ;AN000;
31417                                   MOV     AX,[ss:HIGH_SECTOR]         ;AN000;LB. save current sector #
31418                                   MOV     [ss:SEQ_SECTOR+2],AX        ;AN000;LB. for access mode ref.
31419                                   MOV     AX,[ss:CALLSSEC]            ;AN000;LB.
31420                                   MOV     [ss:SEQ_SECTOR],AX          ;AN000;LB.
31421                                   JMP     short scexit                ;AN000;LB.
31422                                scexit2:                               ;AN000;LB.
31423                                   CLC                                 ;AN000;LB. clear carry
31424                                scexit:                                ;AN000;
31425                                   POP     DI                          ;AN000;LB.
31426                                   POP     ES                          ;AN000;LB. restore registers
31427                                   POP     SI                          ;AN000;LB.
31428                                   POP     DS                          ;AN000;LB.
31429                                   POP     DX                          ;AN000;LB.
31430                                   POP     CX                          ;AN000;LB.
31431                                   retn                                ;AN000;LB.
31432
31433                                ;Break     <IN_SC -- check if in secondary cache>
31434                                ;--------------------------------------------------------------------------
31435                                ;
31436                                ; Procedure Name : IN_SC
31437                                ;
31438                                ; Inputs:  [SC_DRIVE]= requesting drive
31439                                ;       [CURSC_DRIVE]= current SC drive
31440                                ;       [CURSC_SECTOR]= starting scetor # of SC
31441                                ;       [SC_CACHE_COUNT]= SC count
31442                                ;       [HIGH_SECTOR]:DX= sector number
31443                                ; Function:
31444                                ;   Check if the sector is in secondary cache
31445                                ; Output:
31446                                ;   carry clear, in SC
31447                                ;      CX= the index in the secondary cache
31448                                ;   carry set, not in SC
31449                                ;
31450                                ;--------------------------------------------------------------------------
31451
31452                                   ; 22/11/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
31453                                IN_SC:
31454                                   ; SS override for all variables used
31455                                   MOV     AL,[ss:SC_DRIVE]            ;AN000;;LB. current drive
31456                                   CMP     AL,[ss:CurSC_DRIVE]         ;AN000;;LB. same as SC drive
31457                                   JNZ     short outrange2                   ;AN000;;LB. no
31458                                   MOV     AX,[ss:HIGH_SECTOR]         ;AN000;;LB. subtract sector num from
31459                                   MOV     CX,DX                       ;AN000;;LB. secondary starting sector
31460                                   SUB     CX,[ss:CurSC_SECTOR]          ;AN000;;LB. number
31461                                   SBB     AX,[ss:CurSC_SECTOR+2]        ;AN000;;LB.
31462                                   ; 24/09/2023
31463                                   ;CMP    AX,0                        ;AN000;;LB. greater than 64K
31464                                   JNZ     short outrange2                   ;AN000;;LB. yes
31465                                   CMP     CX,[ss:SC_CACHE_COUNT]      ;AN000;;LB. greater than SC count
31466                                   JAE     short outrange2                   ;AN000;;LB. yes
31467                                   CLC                                 ;AN000;;LB. clear carry
31468                                   ;JMP    short inexit                ;AN000;;LB. in SC
31469                                   ; 16/12/2022
31470                                   retn    ; 30/04/2019
31471                                   ; 22/11/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
31472                                   ;jmp    short inexit
31473
31474                                outrange2:                             ;AN000;;LB. set carry
31475                                   STC                                 ;AN000;;LB.
31476                                inexit:                                ;AN000;;LB.
31477                                   retn                                ;AN000;;LB.
31478
31479                                ;Break     <INVALIDATE_SC - invalide secondary cache>
31480                                ;--------------------------------------------------------------------------
31481                                ;
```

```
                        ; Procedure Name : Invalidate_Sc
                        ;
                        ; Inputs:  [SC_DRIVE]= requesting drive
                        ;       [CURSC_DRIVE]= current SC drive
                        ;       [CURSC_SECTOR]= starting scetor # of SC
                        ;       [SC_CACHE_COUNT]= SC count
                        ;       [SC_STATUS]= SC status word
                        ;       [HIGH_SECTOR]:DX= sector number
                        ;
                        ; Function:
                        ;   invalidate secondary cache if in there
                        ; Output:
                        ;   [SC_STATUS] is updated
                        ;-----------------------------------------------------------------------------

                        INVALIDATE_SC:
                            ; SS override for all variables used

                            CALL    IN_SC                       ;AN000;;LB. in secondary cache
                            JC      short outrange              ;AN000;;LB. no
                            MOV     AX,1                        ;AN000;;LB. invalidate the sector
                            SHL     AX,CL                       ;AN000;;LB. in the secondary cache
                            NOT     AX                          ;AN000;;LB.
                            AND     [ss:SC_STATUS],AX           ;AN000;;LB. save the status
                        outrange:                               ;AN000;;LB.
                            retn                                ;AN000;;LB.

                        ; DOSCODE:87A5h (MSDOS 6.21, MSDOS.SYS)
                        ; 22/11/2022
                        ; DOSCODE:876Ah (MSDOS 5.0, MSDOS.SYS)

                        ;Break      <VIRREAD- virtually read data into buffer>
                        ;-----------------------------------------------------------------------------
                        ;
                        ; Procedure Name : SC_FLAG
                        ;
                        ; Inputs:  SC_FLAG = 0, no sectors were read into SC
                        ;               1, continuous sectors were read into SC
                        ; Function:
                        ;      Move data from SC to buffer
                        ; Output:
                        ;    carry clear, data is moved to buffer
                        ;    carry set, bad sector or exceeds maximum sector
                        ;      SC_FLAG =0
                        ;      CALLSCNT=1
                        ;      SC_STATUS= -1 if succeeded
                        ;
                        ;                 0 if failed
                        ;-----------------------------------------------------------------------------

                        VIRREAD:
                            ; SS override for all variables used

                            CMP     byte [ss:SC_FLAG],0         ;AN000;;LB. from SC fill
                            JZ      short sc2end                ;AN000;;LB. no
                            MOV     AX,[ss:TEMP_VAR2]           ;AN000;;LB. restore buffer addr
                            MOV     [ss:CALLXAD+2],AX           ;AN000;;LB.
                            MOV     AX,[ss:TEMP_VAR]            ;AN000;;LB.
                            MOV     [ss:CALLXAD],AX                     ;AN000;;LB.
                            MOV     byte [ss:SC_FLAG],0         ;AN000;;LB. reset sc_flag
                            MOV     word [ss:CALLSCNT],1        ;AN000;;LB. one sector transferred

                            ;TEST   word [SS:DEVCALL_REQSTAT],STERR ;AN000;;LB. error?
                            test    byte [ss:DEVCALL_REQSTAT+1],(STERR>>8) ; 80h
                            JNZ     short scerror               ;AN000;;LB. yes
                            PUSH    DS                          ;AN000;;LB.
                            PUSH    SI                          ;AN000;;LB.
                            PUSH    ES                          ;AN000;;LB.
                            PUSH    DI                          ;AN000;;LB.
                            PUSH    DX                          ;AN000;;LB.
                            PUSH    CX                          ;AN000;;LB.
                            XOR     CX,CX                       ;AN000;;LB. we want first sector in SC
                            CALL    SC2BUF2                     ;AN000;;LB. move data from SC to buf
                            POP     CX
                            POP     DX                          ;AN000;;LB.
                            POP     DI                          ;AN000;;LB.
                            POP     ES                          ;AN000;;LB.
                            POP     SI                          ;AN000;;LB.
                            POP     DS                          ;AN000;;LB.
                            JMP     SHORT sc2end                ;AN000;;LB. return
                        scerror:                                ;AN000;
                            MOV     word [ss:CALLSCNT],1        ;AN000;;LB. reset sector count to 1
                            MOV     word [ss:SC_STATUS],0       ;AN000;;LB. invalidate all SC sectors
                            MOV     byte [ss:CurSC_DRIVE],-1    ;AN000;;LB. invalidate drive
                            STC                                 ;AN000;;LB. carry set
                            retn                                ;AN000;;LB.
                        sc2end:                                 ;AN000;
                            CLC                                 ;AN000;;LB. carry clear
                            retn                                ;AN000;;LB.

                        ; 30/04/2019 - Retro  DOS v4.0
                        ; DOSCODE:87FDh (MSDOS 6.21, MSDOS.SYS)
                        ; 22/11/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
                        ; DOSCODE:87C2h (MSDOS 5.0, MSDOS.SYS)

                        ;Break      <SC2BUF- move data from SC to buffer>
                        ;-----------------------------------------------------------------------------
                        ;
                        ; Procedure Name : SC2BUF
                        ;
                        ; Inputs:  [SC_STATUS] = SC validity status
                        ;       [SC_SECTOR_SIZE] = request sector size
                        ;       [SC_CACHE_PTR] = pointer to SC
                        ; Function:
                        ;      Move data from SC to buffer
                        ; Output:
                        ;       carry clear, in SC  and data is moved
                        ;       carry set, not in SC and data is not moved
                        ;-----------------------------------------------------------------------------

                        SC2BUF:
                            ; SS override for all variables used
                            CALL    IN_SC                       ;AN000;LB. in secondary cache
                            ;JC     short noSC                  ;AN000;LB. no
                            ; 24/09/2023
                            jc      short sexit
                            MOV     AX,1                        ;AN000;LB. check if valid sector
                            SHL     AX,CL                       ;AN000;LB. in the secondary cache
                            TEST    [ss:SC_STATUS],AX           ;AN000;LB.
                            JZ      short noSC                  ;AN000;LB. invalid
                        ;entry SC2BUF2
                        SC2BUF2:                                ;AN000;
                            ;MOV    AX,CX                       ;AN000;LB. times index with
                            ;MUL    word [ss:SC_SECTOR_SIZE]    ;AN000;LB. sector size
```

```
                                    ; 24/09/2023
                                    mov     ax,[ss:SC_SECTOR_SIZE]
                                    xchg    ax,cx ; cx = [ss:SC_SECTOR_SIZE]
                                    mul     cx
                                    ADD     AX,[ss:SC_CACHE_PTR]        ;AN000;LB. add SC starting addr
                                    ADC     DX,[ss:SC_CACHE_PTR+2]      ;AN000;LB.
                                    MOV     DS,DX                       ;AN000;LB. DS:SI-> SC sector addr
                                    MOV     SI,AX                       ;AN000;LB.
                                    MOV     ES,[ss:CALLXAD+2]                   ;AN000;LB. ES:DI-> buffer addr
                                    MOV     DI,[ss:CALLXAD]                     ;AN000;LB.
                                    ; 24/09/2023
                                    ;MOV    CX,[ss:SC_SECTOR_SIZE]      ;AN000;LB. count= sector size
                                    SHR     CX,1                        ;AN000;LB. may use DWORD move for 386
                                ;entry MOVWORDS
                                MOVWORDS:                               ;AN000;
                                    CMP     byte [ss:DDMOVE],0          ;AN000;LB. 386 ?
                                    JZ      short nodd                  ;AN000;LB. no
                                    SHR     CX,1                        ;AN000;LB. words/2
                                    DB      66H                         ;AN000;LB. use double word move
                                nodd:
                                    REP     MOVSW                       ;AN000;LB. move to buffer
                                    CLC                                 ;AN000;LB. clear carry
                                    retn                                ;AN000;LB. exit
                                noSC:                                       ;AN000;
                                    STC                                 ;AN000;LB. set carry
                                sexit:                                      ;AN000;
                                    retn                                ;AN000;LB.

                                %endif      ; SC

                                ;============================================================================
                                ; MKNODE.ASM, MSDOS 6.0, 1991
                                ;============================================================================
                                ; 29/07/2018 - Retro DOS v3.0
                                ; 19/05/2019 - Retro DOS v4.0
                                ; 22/02/2024 - Retro DOS v5.0

                                ;   TITLE   MKNODE - Node maker
                                ;   NAME    MKNODE

                                ;** MKNODE.ASM
                                ;----------------------------------------------------------------------------
                                ;   Low level routines for making a new local file system node
                                ;   and filling in an SFT from a directory entry
                                ;
                                ;   BUILDDIR
                                ;   SETDOTENT
                                ;   MakeNode
                                ;   NEWENTRY
                                ;   FREEENT
                                ;   NEWDIR
                                ;   DOOPEN
                                ;   RENAME_MAKE
                                ;   CHECK_VIRT_OPEN
                                ;
                                ;   Revision history:
                                ;
                                ;    AN000  version 4.0  Jan. 1988
                                ;    A004   PTM 3680  --- Make SFT NAME field offset same as 3.30

                                ;Break   <BUILDDIR,NEWDIR -- ALLOCATE DIRECTORIES>
                                ;----------------------------------------------------------------------------
                                ;
                                ; Procedure Name : BUILDDIR,NEWDIR
                                ;
                                ; Inputs:
                                ;       ES:BP Points to DPB
                                ;       [THISSFT] Set if using NEWDIR entry point
                                ;               (used by ALLOCATE)
                                ;       [LASTENT] current last valid entry number in directory if no free
                                ;               entries
                                ;       [DIRSTART] Points to first cluster of dir (0 means root)
                                ; Function:
                                ;       Grow directory if no free entries and not root
                                ; Outputs:
                                ;       CARRY SET IF FAILURE
                                ;       ELSE
                                ;          AX entry number of new entry
                                ;          If a new dir [DIRSTART],[CLUSFAC],[CLUSNUM],[DIRSEC] set
                                ;               AX = first entry of new dir
                                ;       GETENT should be called to set [LASTENT]
                                ;
                                ;----------------------------------------------------------------------------

                                ; 19/05/2019 - Retro DOS v4.0
                                ; DOSCODE:8845h (MSDOS 6.21, MSDOS.SYS)
                                ; 22/11/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
                                ; DOSCODE:880Ah (MSDOS 6.21, MSDOS.SYS)

                                ; 24/09/2023 - Retro DOS v4.2 (Modified MSDOS 6.21 MSDOS.SYS)
                                ; DOSCODE:8845h (MSDOS 6.22, MSDOS.SYS)

                                ; 23/02/2024 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMDOS.COM)
                                ; DOSCODE:940Ah (PCDOS 7.1, IBMDOS.COM)

                                ; (Windows ME IO.SYS - BIOSCODE:890Ch)

                                BUILDDIR:
                                    ; 29/07/2018 - Retro DOS v3.0
                                    ; IBMDOS.COM (MSDOS 3.3, 1987) - Offset 4E66h

 00005355 A1[D805]                  MOV     AX,[ENTFREE]
 00005358 83F8FF                    CMP     AX,-1 ; 0FFFFh
                                    ;JZ     short CHECK_IF_ROOT
                                    ;CLC
                                    ;retn
                                    ; 24/09/2023
 0000535B 750E                      jne     short builddir_cmc_retn  ; cf=1 (will be 0)

                                CHECK_IF_ROOT:
                                    ; 23/02/2024 (PCDOS 7.1 IBMDOS.COM)
                                    ;;;
 0000535D 833E[DC0A]00              cmp     word [DIRSTART_HW],0
 00005362 7509                      jnz     short NEWDIR
                                    ;;;
 00005364 833E[C205]00              CMP     word [DIRSTART],0
 00005369 7502                      JNZ     short NEWDIR
                                    ;STC                            ; Can't grow root
                                    ; 24/09/2023
                                    ; [DIRSTART]=0, cf=0, zf=1 (cf will be 1 after cmc instruction)
                                builddir_cmc_retn:
                                    ; 24/09/2023
 0000536B F5                        cmc     ; cf=1 <-> cf=0
                                builddir_retn:
```

```
31730 0000536C C3                       retn
31731
31732                                   ;entry   NEWDIR
31733                                 NEWDIR:
31734                                   ; 23/02/2024
31735                                   ;;;
31736 0000536D 8B1E[DC0A]               mov     bx,[DIRSTART_HW]
31737 00005371 891E[E80A]               mov     [CLUSTNUM_HW],bx
31738 00005375 09DB                     or      bx,bx
31739                                   ;;;
31740 00005377 8B1E[C205]               MOV     BX,[DIRSTART]
31741                                   ;;;
31742 0000537B 7504                     jnz     short NEWDIR2 ; 23/02/2024
31743                                   ;;;
31744 0000537D 09DB                     OR      BX,BX
31745 0000537F 7405                     JZ      short NULLDIR
31746                                 NEWDIR2:
31747 00005381 E8B708                   call    GETEOF
31748 00005384 72E6                     jc      short builddir_retn   ; Screw up
31749                                 NULLDIR:
31750                                   ;MOV   CX,1
31751                                   ; 23/02/2024
31752                                   ;;;
31753 00005386 31C9                     xor     cx,cx
31754 00005388 890E[F00A]               mov     [CCOUNT_HW],cx ; 0
31755 0000538C 41                       inc     cx ; 1
31756                                   ;;;
31757 0000538D E84906                   call    ALLOCATE
31758 00005390 72DA                     jc      short builddir_retn
31759 00005392 8B16[C205]               MOV     DX,[DIRSTART]
31760                                   ; 23/02/2024
31761                                   ;;;
31762 00005396 3B16[DC0A]               cmp     dx,[DIRSTART_HW]
31763 0000539A 7519                     jnz     short ADDINGDIR
31764                                   ;;;
31765 0000539C 09D2                     OR      DX,DX
31766 0000539E 7515                     JNZ     short ADDINGDIR
31767                                   ; 23/02/2024
31768                                   ;;;
31769 000053A0 FF36[E80A]               push    word [CLUSTNUM_HW]
31770 000053A4 8F06[EE0A]               pop     word [ROOTCLUST_HW]
31771                                   ;;;
31772 000053A8 E8B4F5                   call    SETDIRSRCH
31773 000053AB 72BF                     jc      short builddir_retn
31774 000053AD C706[4803]FFFF           MOV     word [LASTENT],-1
31775 000053B3 EB3F                     JMP     SHORT GOTDIRREC
31776                                 ADDINGDIR:
31777 000053B5 53                       PUSH    BX
31778                                   ; 23/02/2024
31779                                   ;;;
31780 000053B6 FF36[E80A]               push    word [CLUSTNUM_HW]
31781                                   ;
31782 000053BA 8B1E[DE0A]               mov     bx,[CLUSNUM_HW]
31783 000053BE 891E[E80A]               mov     [CLUSTNUM_HW],bx
31784                                   ;;;
31785 000053C2 8B1E[BC05]               MOV     BX,[CLUSNUM]
31786 000053C6 E8EF0E                   call    IsEOF
31787                                   ;;;
31788 000053C9 8F06[E80A]               pop     word [CLUSTNUM_HW] ; 23/02/2024
31789                                   ;;;
31790 000053CD 5B                       POP     BX
31791 000053CE 721D                     JB      short NOTFIRSTGROW
31792                                 ;;;; 10/17/86 update CLUSNUM in the fastopen cache
31793 000053D0 891E[BC05]               MOV     [CLUSNUM],BX
31794                                   ; 24/09/2023
31795                                   ;PUSH   CX ; (not necessary)
31796 000053D4 50                       PUSH    AX
31797 000053D5 55                       PUSH    BP
31798                                   ; 23/02/2024
31799                                   ;;;
31800 000053D6 A1[E80A]                 mov     ax,[CLUSTNUM_HW]
31801 000053D9 A3[DE0A]                 mov     [CLUSNUM_HW],ax
31802                                   ;;;
31803 000053DC B401                     MOV     AH,1                    ; CLUSNUM update
31804                                   ; 15/12/2022
31805 000053DE 268A5600                 mov     dl,[ES:BP] ; 09/09/2018
31806                                   ; 22/11/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
31807                                   ;;mov   dl,[es:bp+0]
31808                                   ;MOV    DL,[ES:BP+DPB.DRIVE]  ; drive #
31809 000053E2 8B0E[C205]               MOV     CX,[DIRSTART]           ; first cluster #
31810 000053E6 89DD                     MOV     BP,BX                   ; CLUSNUM
31811 000053E8 E87CD9                   call    FastOpen_Update
31812 000053EB 5D                       POP     BP
31813 000053EC 58                       POP     AX
31814                                   ; 24/09/2023
31815                                   ;POP    CX
31816
31817                                 ;;;; 10/17/86 update CLUSNUM in the fastopen cache
31818                                 NOTFIRSTGROW:
31819 000053ED 89DA                     MOV     DX,BX
31820 000053EF 30DB                     XOR     BL,BL
31821 000053F1 E8A305                   call    FIGREC
31822                                 GOTDIRREC:
31823                                   ;mov    cl,[es:bp+4]
31824 000053F4 268A4E04                 MOV     CL,[ES:BP+DPB.CLUSTER_MASK]
31825                                   ;INC    CL
31826                                   ; 27/06/2024
31827 000053F8 41                       inc     cx
31828 000053F9 30ED                     XOR     CH,CH
31829                                 ZERODIR:
31830 000053FB 51                       PUSH    CX
31831                                   ; 22/09/2023
31832                                   ;;mov   byte [ALLOWED],18h
31833                                   ;MOV    byte [ALLOWED],Allowed_FAIL+Allowed_RETRY ; *
31834 000053FC B0FF                     MOV     AL,0FFH
31835                                   ;call   GETBUFFR
31836 000053FE E82A15                   call    GETBUFFRD ; *
31837 00005401 7302                     JNC     short GET_SSIZE
31838 00005403 59                       POP     CX
31839 00005404 C3                       retn
31840
31841                                 GET_SSIZE:
31842                                   ;mov    cx,[es:bp+2]
31843 00005405 268B4E02                 MOV     CX,[ES:BP+DPB.SECTOR_SIZE]
31844 00005409 06                       PUSH    ES
31845 0000540A C43E[E205]               LES     DI,[CURBUF]
31846                                   ;or     byte [es:di+5],4
31847 0000540E 26804D0504               OR      byte [ES:DI+BUFFINFO.buf_flags],buf_isDIR
31848 00005413 57                       PUSH    DI
31849                                   ;;;add  di,16   ; MSDOS 3.3
31850                                   ;;add   di,20   ; MSDOS 6.0
31851                                   ;add    di,24   ; PCDOS 7.1 ; 23/02/2024
31852 00005414 83C718                   ADD     DI,BUFINSIZ
31853 00005417 31C0                     XOR     AX,AX
```

```
00005419 D1E9                       SHR     CX,1
0000541B F3AB                       REP     STOSW
0000541D 7301                       JNC     short EVENZ
0000541F AA                         STOSB
                                EVENZ:
00005420 5F                         POP     DI

                                ; 23/02/2024
                                %if 0
                                    ; MSDOS 6.0
                                    TEST    byte [ES:DI+BUFFINFO.buf_flags],buf_dirty
                                                                    ;LB. if already dirty          ;AN000;
                                    JNZ     short yesdirty7         ;LB.  don't increment dirty count ;AN000;
                                    call    INC_DIRTY_COUNT         ;LB.                          ;AN000;

                                    ;or     byte [es:di+5],40h
                                    OR      byte [ES:DI+BUFFINFO.buf_flags],buf_dirty
                                %else
                                    ; 23/02/2024 - Retro DOS v5.0 (PCDOS 7.1 IBMDOS.COM)
00005421 E89A17                     call    SET_BUF_DIRTY
                                %endif

                                yesdirty7:
00005424 07                         POP     ES
00005425 59                         POP     CX

                                    ; 19/05/2019 - Retro DOS v4.0

                                    ; MSDOS 3.3
                                    ;INC    DX

                                    ; MSDOS 6.0
                                    ; 24/09/2023
                                    ;add    dx,1
                                    ;;adc   word [HIGH_SECTOR],0
                                    ;; 24/09/2023
                                    ;; ax=0
                                    ;adc    [HIGH_SECTOR],ax ; 0
                                    ; 24/09/2023
00005426 42                         inc     dx
00005427 7504                       jnz     short loop_zerodir
00005429 FF06[0706]                 inc     word [HIGH_SECTOR]
                                loop_zerodir:
0000542D E2CC                       LOOP    ZERODIR

0000542F A1[4803]                   MOV     AX,[LASTENT]
00005432 40                         INC     AX
                                    ; 24/09/2023
                                    ; cf=0
                                    ;CLC
00005433 C3                         retn

                                ;--------------------------------------------------------------------------
                                ;
                                ; Procedure Name : SETDOTENT
                                ;
                                ; set up a . or .. directory entry for a directory.
                                ;
                                ;   Inputs:     ES:DI point to the beginning of a directory entry.
                                ;               AX contains ". " or ".."
                                ;               DX contains first cluster of entry
                                ;
                                ;--------------------------------------------------------------------------

                                    ; 23/02/2024 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMDOS.COM)

                                SETDOTENT:
                                ;   Fill in name field
00005434 AB                         STOSW
00005435 B90400                     MOV     CX,4
00005438 B82020                     MOV     AX,"  " ; 2020h
0000543B F3AB                       REP     STOSW
0000543D AA                         STOSB

                                ;   Set up attribute
                                    ;mov    al, 10h
0000543E B010                       MOV     AL,attr_directory
00005440 AA                         STOSB

                                ;   Initialize time and date of creation
                                    ;ADD    DI,10
                                    ; 23/04/2024 - Retro DOS v5.0 (PCDOS 7.1 IBMDOS.COM)
                                    ;;;
00005441 83C708                     add     di,8
00005444 A1[F20A]                   mov     ax,[CLUSTERS_HW]
00005447 AB                         stosw           ; Set up first cluster field (hw)
                                    ;;;
00005448 8B36[9E05]                 MOV     SI,[THISSFT]
                                    ;mov    ax,[si+0Dh]
0000544C 8B440D                     MOV     AX,[SI+SF_ENTRY.sf_time]
0000544F AB                         STOSW
                                    ;mov    ax,[si+0Fh]
00005450 8B440F                     MOV     AX,[SI+SF_ENTRY.sf_date]
00005453 AB                         STOSW

                                ;   Set up first cluster field (lw)
00005454 89D0                       MOV     AX,DX
00005456 AB                         STOSW

                                ;   0 file size
                                    ;XOR    AX,AX
00005457 91                         xchg    ax,cx ; 23/02/2024
00005458 AB                         STOSW
00005459 AB                         STOSW
0000545A C3                         retn

                                ;Break   <MAKENODE -- CREATE A NEW NODE>
                                ;--------------------------------------------------------------------------
                                ;
                                ; Procedure Name : MakeNode
                                ;
                                ; Inputs:
                                ;       AL - attribute to create
                                ;       AH = 0 if it is ok to truncate a file already by this name
                                ;   AH != 0 if truncation not allowed (prexisting file is an error)
                                ;               (AH ignored on dirs and devices)
                                ;
                                ;        NOTE: When making a DIR or volume ID, AH need not be set since
                                ;              a name already existant is ALWAYS an error in these cases.
                                ;
                                ;       [WFP_START] Points to WFP string ("d:/" must be first 3 chars, NUL
                                ;               terminated)
                                ;       [CURR_DIR_END] Points to end of Current dir part of string
                                ;               ( = -1 if current dir not involved, else
```

```
                                ;               Points to first char after last "/" of current dir part)
                                ;       [THISCDS] Points to CDS being used
                                ;       [THISSFT] Points to an empty SFT. EXCEPT sf_mode filled in.
                                ; Function:
                                ;       Make a new node
                                ; Outputs:
                                ;       Sets EXTERR_LOCUS = errLOC_Disk or errLOC_Unk via GetPathNoset
                                ;       CARRY SET IF ERROR
                                ;          AX = 1 A node by this name exists and is a directory
                                ;          AX = 2 A new node could not be created
                                ;          AX = 3 A node by this name exists and is a disk file
                                ;               (AH was NZ on input)
                                ;          AX = 4 Bad Path
                                ;               SI return from GetPath maintained
                                ;          AX = 5 Attribute mismatch
                                ;          AX = 6 Sharing Violation
                                ;               (INT 24 generated ALWAYS since create is always compat mode
                                ;          AX = 7 file not found for Extended Open (not exists and fails)
                                ;       ELSE
                                ;          AX = 0 Disk Node
                                ;          AX = 3 Device Node (error in some cases)
                                ;          [DIRSTART],[DIRSEC],[CLUSFAC],[CLUSNUM] set to directory
                                ;               containing new node.
                                ;          [CURBUF+2]:BX Points to entry
                                ;          [CURBUF+2]:SI Points to entry.dir_first
                                ;          [THISSFT] is filled in
                                ;               sf_mode = unchanged.
                                ;          Attribute byte in entry is input AL
                                ; DS preserved, others destroyed
                                ;
                                ;--------------------------------------------------------------------------

                                ; 19/05/2019 - Retro DOS v4.0
                                ; DOSCODE:8925h (MSDOS 6.21, MSDOS.SYS)

                                ; 22/11/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
                                ; DOSCODE:88EAh (MSDOS 5.0, MSDOS.SYS)

                                ; 23/02/2024 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMDOS.COM)
                                ; DOSCODE:951Ah (PCDOS 7.1, IBMDOS.COM)

                                MakeNode:
                                    ;mov    word [CREATING],0E5FFh
0000545B C706[7E05]FFE5             MOV     WORD [CREATING],DIRFREE*256 + 0FFh ; Creating, not DEL *.*
00005461 50                         PUSH    AX              ; Save AH value
00005462 C606[4C03]00               MOV     byte [NoSetDir],0
00005467 A2[6D05]                   MOV     [SATTRIB],AL
0000546A E8ACF6                     call    GetPathNoSet
0000546D 88CA                       MOV     DL,CL           ; Save CL info
                                    ;MOV    CX,AX           ; Device ID to CH
                                    ; 23/02/2024
0000546F 91                         xchg    ax,cx
00005470 58                         POP     AX              ; Get back AH
00005471 732D                       JNC     short make_exists ; File existed
00005473 7505                       JNZ     short make_err_4 ; Path bad
00005475 80FA80                     CMP     DL,80h          ; Check "CL" return from GETPATH
00005478 7405                       JZ      short make_type       ; Name simply not found, and no metas
                                make_err_4:
0000547A B004                       MOV     AL,4            ; case 1 bad path
                                make_err_ret:
                                    ;XOR    AH,AH
                                    ; 23/02/2024
0000547C 98                         cbw
0000547D F9                         STC
                                ;make_retn: ; 22/11/2022
0000547E C3                         retn

                                    ;entry  RENAME_MAKE     ; Used by DOS_RENAME to "copy" a node
                                RENAME_MAKE:
                                make_type:
                                ;Extended Open hooks
                                    ; MSDOS 6.0
                                    ;TESTB  EXTOPEN_ON,EXT_OPEN_ON;FT. from extended open              ;AN000;
0000547F F606[F605]01               test    byte [EXTOPEN_ON],EXT_OPEN_ON ; 1
00005484 7411                       JZ      short make_type2      ;FT. no                      ;AN000;
00005486 800E[F605]04               OR      byte [EXTOPEN_ON],EXT_FILE_NOT_EXISTS ; 4
                                                                  ;FT. set for extended open ;AN000;
                                    ;TESTB  EXTOPEN_FLAG,0F0H     ;FT. not exists and fails    ;AN000;
0000548B F606[F405]F0               test    byte [EXTOPEN_FLAG],0F0h
00005490 7505                       JNZ     short make_type2      ;FT. no                      ;AN000;
00005492 F9                         STC                           ;FT. set carry               ;AN000;
00005493 B80700                     MOV    AX,7                   ;FT. file not found          ;AN000;
                                    ; 22/11/2022
                                make_retn:
                                    ;return
00005496 C3                         retn                          ;FT.                         ;AN000;

                                ;   Extended Open hooks

                                make_type2:
00005497 C43E[9E05]                 LES     DI,[THISSFT]
0000549B 31C0                       XOR     AX,AX           ; nothing exists Disk Node
0000549D F9                         STC                     ; Not found
0000549E EB59                       JMP     short make_new

                                ; The node exists. It may be either a device, directory or file:
                                ;   Zero set => directory
                                ;   High bit of CH on => device
                                ;   else => file

                                make_exists:
000054A0 7447                       JZ      short make_exists_dir
000054A2 B003                       MOV     AL,3            ; file exists type 3  (error or device node)
                                    ;test   byte [ATTRIB],18h
000054A4 F606[6B05]18               TEST    byte [ATTRIB],attr_volume_id+attr_directory
000054A9 753A                       JNZ     short make_err_ret_5
                                                            ; Cannot already exist as Disk or Device Node
                                                            ;  if making DIR or Volume ID
000054AB 08ED                       OR      CH,CH
000054AD 781A                       JS      short make_share ; No further checks on attributes if device
000054AF 08E4                       OR      AH,AH
000054B1 75C9                       JNZ     short make_err_ret ; truncating NOT OK (AL = 3)
000054B3 51                         PUSH    CX              ; Save device ID
000054B4 8E06[E405]                 MOV     ES,[CURBUF+2]
                                    ;mov    ch,[es:bx+0Bh]
000054B8 268A6F0B                   MOV     CH,[ES:BX+dir_entry.dir_attr] ; Get file attributes
                                    ;test   ch,1
000054BC F6C501                     test    CH,attr_read_only
000054BF 7523                       JNZ     short make_err_ret_5P ; Cannot create on read only files
000054C1 E803F9                     call    MatchAttributes
000054C4 59                         POP     CX              ; Devid back in CH
000054C5 751E                       JNZ     short make_err_ret_5 ; Attributes not ok
000054C7 30C0                       XOR     AL,AL           ; AL = 0, Disk Node
                                make_share:
```

```
                                    ;XOR    AH,AH
                                    ; 23/02/2024
000054C9 98                         cbw
000054CA 50                         PUSH    AX              ; Save Disk or Device node
000054CB 51                         PUSH    CX              ; Save Device ID
000054CC 88EC                       MOV     AH,CH           ; Device ID to AH
000054CE E83201                     CALL    DOOPEN          ; Fill in SFT for share check
000054D1 C43E[9E05]                 LES     DI,[THISSFT]
000054D5 56                         push    si
000054D6 53                         push    bx              ; Save CURBUF pointers
000054D7 E8AF2F                     call    ShareEnter
000054DA 734E                       jnc     short MakeEndShare

                                ; User failed request.
000054DC 5B                         pop     bx
000054DD 5E                         pop     si
000054DE 59                         pop     cx
000054DF 58                         pop     ax

                                Make_Share_ret:
000054E0 B006                       MOV     AL,6
000054E2 EB98                       JMP     short make_err_ret

                                make_err_ret_5P:
000054E4 59                         POP     CX              ; Get back device ID
                                make_err_ret_5:
000054E5 B005                       MOV      AL,5           ; Attribute mismatch
                                        ; 22/11/2022
000054E7 EB93                       JMP     short make_err_ret

                                make_exists_dir:
000054E9 B001                       MOV     AL,1            ; exists as directory, always an error
                                    ; 22/11/2022
000054EB EB8F                       JMP     short make_err_ret

                                make_save:
000054ED 50                         PUSH    AX              ; Save whether Disk or File
000054EE 89C8                       MOV     AX,CX           ; Device ID to AH
000054F0 E86800                     CALL    NEWENTRY
000054F3 58                         POP     AX              ; 0 if Disk, 3 if File
000054F4 73A0                       jnc     short make_retn
000054F6 B002                       MOV     AL,2            ; create failed case 2
                                make_save_retn:
000054F8 C3                         retn

                                make_new:
000054F9 E8F1FF                     call    make_save
000054FC 72FA                       jc      short make_save_retn  ; case 2 fail
                                    ;test   byte [ATTRIB],10h
000054FE F606[6B05]10               test    BYTE [ATTRIB],attr_directory
00005503 75F3                       jnz     short make_save_retn  ; Don't "open" directories,
                                                                  ; so don't tell the sharer about them
00005505 50                         push    ax
00005506 53                         push    bx
00005507 56                         push    si
00005508 E87E2F                     call    ShareEnter
0000550B 5E                         pop     si
0000550C 5B                         pop     bx
0000550D 58                         pop     ax
0000550E 73E8                       jnc     short make_save_retn

                                ; We get here by having the user FAIL a share problem. Typically a failure of
                                ; this nature is an out-of-space or an internal error. We clean up as best as
                                ; possible: delete the newly created directory entry and return share_error.

00005510 50                         PUSH    AX
00005511 C43E[E205]                 LES     DI,[CURBUF]
                                    ;mov    byte [es:bx],0E5h
00005515 26C607E5                   MOV     BYTE [ES:BX],DIRFREE  ; nuke newly created entry.

                                ; 23/02/2024
                                %if 0
                                    ; MSDOS 6.0
                                    ;test   byte [es:di+5],40h
                                    TEST    byte [ES:DI+BUFFINFO.buf_flags],buf_dirty
                                                                  ;LB. if already dirty          ;AN000;
                                    JNZ     short yesdirty8       ;LB.  don't increment dirty count ;AN000;
                                    ; 22/11/2022
                                    call    INC_DIRTY_COUNT       ;LB.                          ;AN000;
                                    ;or     byte [es:di+5],40h
                                    OR      byte [ES:DI+BUFFINFO.buf_flags],buf_dirty ; flag buffer as dirty
                                yesdirty8:
                                %else
                                    ; 23/02/2024 - Retro DOS v5.0 (PCDOS 7.1 IBMDOS.COM)
00005519 E8A216                     call    SET_BUF_DIRTY
                                %endif
0000551C C42E[8A05]                 LES     BP,[THISDPB]
                                    ; 15/12/2022
00005520 268A4600                   mov     al,[ES:BP]
                                    ; 22/11/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
                                    ;;mov   al,[es:bp+0]
                                    ;MOV    AL,[ES:BP+DPB.DRIVE]  ; get drive for flush
00005524 E87115                     call    FLUSHBUF              ; write out buffer.
00005527 58                         POP     AX
00005528 EBB6                       jmp     short Make_Share_ret

                                ; We have found an existing file. We have also entered it into the share set.
                                ; At this point we need to call newentry to correctly address the problem of
                                ; getting rid of old data (create an existing file) or creating a new
                                ; directory entry (create a new file). Unfortunately, this operation may
                                ; result in an INT 24 that the user doesn't return from, thus locking the file
                                ; irretrievably into the share set. The correct solution is for us to LEAVE
                                ; the share set now, do the operation and then reassert the share access.
                                ;
                                ; We are allowed to do this! There is no window! After all, we are in
                                ; critDisk here and for someone else to get in, they must enter critDisk also.

                                    ; 22/11/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
                                    ; DOSCODE:89C8h (MSDOS 5.0, MSDOS.SYS)

                                MakeEndShare:
0000552A C43E[9E05]                 LES     DI,[THISSFT]          ; grab SFT
0000552E 31C0                       XOR     AX,AX
00005530 E8AFC3                     call    ECritSFT
00005533 268705                     xchg    AX,[ES:DI]
                                    ;XCHG   AX,[ES:DI+SF_ENTRY.sf_ref_count]
00005536 50                         push    ax
00005537 57                         push    di
00005538 06                         push    es
00005539 9C                         PUSHF
0000553A E8472F                     call    ShareEnd              ; remove sharing
0000553D 9D                         POPF
0000553E 07                         pop     es
0000553F 5F                         pop     di
```

```
32226 00005540 268F05                   pop     word [ES:DI]
32227                                   ;pop    word [ES:DI+SF_ENTRY.sf_ref_count]
32228 00005543 E8C9C3                   call    LCritSFT
32229                                   ; 22/11/2022
32230                                   ; DOSCODE:89E4h (MSDOS 5.0, MSDOS.SYS)
32231 00005546 5B                       pop     bx
32232 00005547 5E                       pop     si
32233 00005548 59                       pop     cx
32234 00005549 58                       pop     ax
32235 0000554A E8A0FF                   CALL    make_save
32236
32237                                 ; If the user failed, we do not reenter into the sharing set.
32238
32239 0000554D 72A9                     jc      short make_save_retn  ; bye if error
32240 0000554F 50                       push    ax
32241 00005550 53                       push    bx
32242 00005551 56                       push    si
32243 00005552 9C                       PUSHF
32244 00005553 E8332F                   call    ShareEnter
32245 00005556 9D                       POPF
32246 00005557 5E                       pop     si
32247 00005558 5B                       pop     bx
32248 00005559 58                       pop     ax
32249
32250                                 ; If Share_check fails, then we have an internal ERROR!!!!!
32251
32252                                 makeendshare_retn:
32253 0000555A C3                       retn
32254
32255                                 ;--------------------------------------------------------------------------
32256                                 ;
32257                                 ; Procedure Name : NEWENTRY
32258                                 ;
32259                                 ; Inputs:
32260                                 ;       [THISSFT] set
32261                                 ;       [THISDPB] set
32262                                 ;       [LASTENT] current last valid entry number in directory if no free
32263                                 ;                 entries
32264                                 ;       [VOLID] set if a volume ID was found during search
32265                                 ;     Attrib Contains attributes for new file
32266                                 ;       [DIRSTART] Points to first cluster of dir (0 means root)
32267                                 ;       CARRY FLAG INDICATES STATUS OF SEARCH FOR FILE
32268                                 ;                 NC means file existed (device)
32269                                 ;                 C  means file did not exist
32270                                 ;       AH = Device ID byte
32271                                 ;       If FILE
32272                                 ;       [CURBUF+2]:BX points to start of directory entry
32273                                 ;       [CURBUF+2]:SI points to dir_first of directory entry
32274                                 ;       If device
32275                                 ;       DS:BX points to start of "fake" directory entry
32276                                 ;       DS:SI points to dir_first of "fake" directory entry
32277                                 ;                 (has DWORD pointer to device header)
32278                                 ; Function:
32279                                 ;       Make a new directory entry
32280                                 ;       If an old one existed it is truncated first
32281                                 ; Outputs:
32282                                 ;       Carry set if error
32283                                 ;                 Can't grow dir, atts didn't match, attempt to make 2nd
32284                                 ;                 vol ID, user FAILed to I 24
32285                                 ;       else
32286                                 ;                 outputs of DOOPEN
32287                                 ; DS, BX, SI preserved (meaning on SI BX, not value), others destroyed
32288                                 ;
32289                                 ;--------------------------------------------------------------------------
32290
32291                                     ; 22/11/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
32292                                     ; DOSCODE:89F9h (MSDOS 5.0, MSDOS.SYS)
32293
32294                                     ; 23/02/2024 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMDOS.COM)
32295                                     ; DOSCODE:961Ah (PCDOS 7.1, IBMDOS.COM)
32296
32297                                     ; (MSDOS 6.22 MSDOS.SYS - DOSCODE:8A34h)
32298                                     ; (Windows ME IO.SYS - BIOSCODE:8B7Dh)
32299
32300                                 NEWENTRY:
32301 0000555B C42E[8A05]               LES     BP,[THISDPB]
32302 0000555F 7315                     JNC     short EXISTENT
32303 00005561 803E[4A03]00             CMP     byte [FAILERR],0
32304                                   ;STC
32305                                   ;jnz    short makeendshare_retn       ; User FAILed, node might exist
32306                                   ; 24/09/2023
32307 00005566 750C                     jnz     short ERRRET3
32308 00005568 E8EAFD                   CALL    BUILDDIR        ; Try to build dir
32309 0000556B 72ED                     jc      short makeendshare_retn       ; Failed
32310 0000556D E832F3                   call    GETENT          ; Point at that free entry
32311 00005570 72E8                     jc      short makeendshare_retn       ; Failed
32312 00005572 EB0E                     JMP     SHORT FREESPOT
32313
32314                                 ERRRET3:
32315 00005574 F9                       STC
32316                                 newentry_retn:
32317 00005575 C3                       retn
32318
32319                                 EXISTENT:
32320 00005576 08E4                     OR      AH,AH           ; Check if file is I/O device
32321 00005578 7903                     JNS     short NOT_DEV1
32322 0000557A E98600                   JMP     DOOPEN          ; If so, proceed with open
32323
32324                                 NOT_DEV1:
32325 0000557D E84D01                   call    FREEENT ; Free cluster chain
32326 00005580 72F3                     jc      short newentry_retn ; Failed
32327                                 FREESPOT:
32328                                   ;test   byte [ATTRIB],8
32329 00005582 F606[6B05]08             test    BYTE [ATTRIB],attr_volume_id
32330 00005587 7407                     JZ      short NOTVOLID
32331 00005589 803E[7B05]00             CMP     BYTE [VOLID],0
32332 0000558E 75E4                     JNZ     short ERRRET3  ; Can't create a second volume ID
32333                                 NOTVOLID:
32334 00005590 8E06[E405]               MOV     ES,[CURBUF+2]
32335 00005594 89DF                     MOV     DI,BX
32336
32337 00005596 BE[4B05]                 MOV     SI,NAME1
32338
32339 00005599 B90500                   MOV     CX,5
32340 0000559C F3A5                     REP     MOVSW
32341 0000559E A4                       MOVSB                   ; Move name into dir entry
32342 0000559F A0[6B05]                 MOV     AL,[ATTRIB]
32343 000055A2 AA                       STOSB                   ; Attributes
32344
32345                                 ;; File Tagging for Create DOS 4.00
32346 000055A3 B105                     MOV     CL,5            ;FT. assume normal FBUGBUG   ;AN000;
32347                                 ;; File Tagging for Create DOS 4.00
32348
32349 000055A5 31C0                     XOR     AX,AX
```

```
32350 000055A7 F3AB                     REP     STOSW           ; Zero pad
32351 000055A9 E8B4B5                   call    DATE16
32352 000055AC 92                       XCHG    AX,DX
32353 000055AD AB                       STOSW                   ; dir_time
32354 000055AE 92                       XCHG    AX,DX
32355                                   ; 23/02/2024 (PCDOS 7.1 IBMDOS.COM)
32356                                   ;;;
32357 000055AF 268945FA                 mov     [es:di-6],ax    ; last access date
32358                                   ;;;
32359 000055B3 AB                       STOSW                   ; dir_date
32360 000055B4 31C0                     XOR     AX,AX
32361 000055B6 57                       PUSH    DI              ; Correct SI input value
32362                                                           ; (recomputed for new buffer)
32363 000055B7 AB                       STOSW                   ; Zero dir_first and size
32364 000055B8 AB                       STOSW
32365 000055B9 AB                       STOSW
32366                                 updnxt:
32367 000055BA 8B36[E205]               MOV     SI,[CURBUF]
32368
32369                                   ; 19/05/2019 - Retro DOS v4.0
32370
32371                                   ; MSDOS 6.0
32372 000055BE 26F6440540              TEST    byte [ES:SI+BUFFINFO.buf_flags],buf_dirty
32373                                                   ;LB. if already dirty           ;AN000;
32374 000055C3 7508                     JNZ     short yesdirty9       ;LB.  don't increment dirty count ;AN000;
32375 000055C5 E80216                   call    INC_DIRTY_COUNT       ;LB.                              ;AN000;
32376
32377                                   ;or     byte [es:si+5],40h
32378 000055C8 26804C0540               OR      byte [ES:SI+BUFFINFO.buf_flags],buf_dirty
32379                                 yesdirty9:
32380 000055CD C42E[8A05]               LES     BP,[THISDPB]
32381                                   ; 15/12/2022
32382 000055D1 268A4600                 MOV     AL,[ES:BP]
32383                                   ; 22/11/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
32384                                   ;;mov   al,[es:bp+0]
32385                                   ;MOV    AL,[ES:BP+DPB.DRIVE] ; Sets AH value again (in AL)
32386 000055D5 50                       PUSH    AX
32387 000055D6 53                       PUSH    BX
32388
32389                                 ; If we have a file, we need to increment the open ref. count so that
32390                                 ; we have some protection against invalid media changes if an Int 24
32391                                 ; error occurs.
32392                                 ; Do nothing for a device.
32393
32394 000055D7 06                         push    es
32395 000055D8 57                         push    di
32396 000055D9 C43E[9E05]                 LES     DI,[THISSFT]
32397                                     ;test   word [es:di+5],80h
32398                                     ;TEST   word [ES:DI+SF_ENTRY.sf_flags],devid_device
32399 000055DD 26F6450580                 test    byte [ES:DI+SF_ENTRY.sf_flags],devid_device
32400 000055E2 7512                       jnz     short GotADevice
32401 000055E4 1E                         push    ds
32402 000055E5 53                         push    bx
32403 000055E6 C51E[8A05]                 LDS     BX,[THISDPB]
32404                                     ;mov    [es:di+7],bx
32405 000055EA 26895D07                   MOV     [ES:DI+SF_ENTRY.sf_devptr],BX
32406 000055EE 8CDB                       MOV     BX,DS
32407                                     ;mov    [es:di+9],bx
32408 000055F0 26895D09                   MOV     [ES:DI+SF_ENTRY.sf_devptr+2],BX
32409 000055F4 5B                         pop     bx
32410 000055F5 1F                         pop     ds ; need to use DS for segment later on
32411
32412                                 ; 23/02/2024 Retro DOS v5.0 (PCDOS 7.1 IBMDOS.COM)
32413                                 %if 0
32414                                     call    DEV_OPEN_SFT   ; increment ref. count
32415                                     mov     byte [VIRTUAL_OPEN],1; set flag
32416                                 %endif
32417
32418                                 GotADevice:
32419 000055F6 5F                         pop     di
32420 000055F7 07                         pop     es
32421
32422 000055F8 E89D14                     call    FLUSHBUF
32423
32424                                 ; 23/02/2024 Retro DOS v5.0 (PCDOS 7.1 IBMDOS.COM)
32425                                 %if 0
32426                                     Call    CHECK_VIRT_OPEN       ; decrement ref. count;AN000;
32427                                 %endif
32428 000055FB 5B                         POP     BX
32429 000055FC 58                         POP     AX
32430 000055FD 5E                         POP     SI              ; Get SI input back
32431 000055FE 88C4                       MOV     AH,AL           ; Get I/O driver number back
32432 00005600 7301                       jnc     short DOOPEN
32433 00005602 C3                         retn                    ; Failed
32434
32435                                 ;NOTE FALL THROUGH
32436
32437                                 ; DOSCODE:8AE4h (MSDOS 6.21, MSDOS.SYS)
32438
32439                                 ; 25/11/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
32440                                 ; DOSCODE:8AA9h (MSDOS 5.0, MSDOS.SYS)
32441
32442                                 ; DOOPEN
32443                                 ;-----------------------------------------------------------------------------
32444                                 ;
32445                                 ; Inputs:
32446                                 ;       [THISDPB] points to DPB if file
32447                                 ;       [THISSFT] points to SFT being used
32448                                 ;       AH = Device ID byte
32449                                 ;       If FILE
32450                                 ;       [CURBUF+2]:BX points to start of directory entry
32451                                 ;       [CURBUF+2]:SI points to dir_first of directory entry
32452                                 ;       If device
32453                                 ;       DS:BX points to start of "fake" directory entry
32454                                 ;       DS:SI points to dir_first of "fake" directory entry
32455                                 ;                 (has DWORD pointer to device header)
32456                                 ; Function:
32457                                 ;       Fill in SFT from dir entry
32458                                 ; Outputs:
32459                                 ;       CARRY CLEAR
32460                                 ;       sf_ref_count and sf_mode fields not altered
32461                                 ;       sf_flags high byte = 0
32462                                 ;       sf_flags low byte = AH except
32463                                 ;       sf_flags Bit 6 set (not dirty or not EOF)
32464                                 ;       sf_attr sf_date sf_time sf_name set from entry
32465                                 ;       sf_position = 0
32466                                 ;       If device
32467                                 ;       sf_devptr = dword at dir_first (pointer to device header)
32468                                 ;       sf_size = 0
32469                                 ;       If file
32470                                 ;       sf_firclus sf_size set from entry
32471                                 ;       sf_devptr = [THISDPB]
32472                                 ;       sf_cluspos = 0
32473                                 ;       sf_lstclus = sf_firclus
```

```
                                ;     sf_dirsec sf_dirpos set
                                ; DS,SI,BX preserved, others destroyed
                                ;
                                ;--------------------------------------------------------------------------

                                    ; 23/02/2024 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMDOS.COM)
                                    ; DOSCODE:96C3h (PCDOS 7.1, IBMDOS.COM)

                                    ; (MSDOS 6.22 MSDOS.SYS - DOSCODE:8AE4h)
                                    ; (Windows ME IO.SYS - BIOSCODE:8C4Ch)

                                    ;entry  DOOPEN
                                DOOPEN:
                                ;   Generate and store attribute

00005603 88E6                       MOV     DH,AH           ; AH to different place
                                    ; 23/02/2024 (PCDOS 7.1 IBMDOS.COM)
                                    ;;;
00005605 30D2                       xor     dl,dl ; 0
00005607 08E4                       or      ah,ah
00005609 780D                       js      short DEV_SFT0
0000560B C43E[8A05]                 les     di,[THISDPB]
                                    ;cmp    word [es:di+0Fh],0
0000560F 26837D0F00                 cmp     word [es:di+DPB.FAT_SIZE],0
00005614 7402                       jz      short DEV_SFT0 ; FAT32
00005616 FEC2                       inc     dl ; 1
                                DEV_SFT0:
                                    ;;;
00005618 C43E[9E05]                 LES     DI,[THISSFT]
                                    ;add    di,4
0000561C 83C704                     ADD     DI,SF_ENTRY.sf_attr ; Skip ref_count and mode fields
                                    ; 24/09/2023
0000561F 31C0                       xor     ax,ax
                                    ;XOR    AL,AL           ; Assume it's a device, devices have an
                                                            ;  attribute of 0 (for R/O testing etc).
00005621 08F6                       OR      DH,DH           ; See if our assumption good.
00005623 7807                       JS      short DEV_SFT1 ; If device DS=DOSGROUP
00005625 8E1E[E405]                 MOV     DS,[CURBUF+2]
                                    ;mov    al,[BX+0Bh]
00005629 8A470B                     MOV     AL,[BX+dir_entry.dir_attr]
                                                            ; If file, get attrib from dir entry
                                DEV_SFT1:
0000562C AA                         STOSB                   ; sf_attr, ES:DI -> sf_flags

                                ;   Generate and store flags word

                                    ; 24/09/2023
                                    ;XOR    AX,AX
                                    ; ah=0
0000562D 88F0                       MOV     AL,DH
                                    ;or     al,40h
0000562F 0C40                       OR      AL,devid_file_clean
00005631 AB                         STOSW                   ; sf_flags, ES:DI -> sf_devptr

                                ;   Generate and store device pointer

00005632 1E                         PUSH    DS
                                    ;lds    ax,[bx+1Ah]
00005633 C5471A                     LDS     AX,[BX+dir_entry.dir_first] ; Assume device
00005636 08F6                       OR      DH,DH
00005638 7805                       JS      short DEV_SFT2

                                ;hkn; SS override
0000563A 36C506[8A05]               LDS     AX,[SS:THISDPB]       ; Was file
                                DEV_SFT2:
0000563F AB                         STOSW                   ; store offset
00005640 8CD8                       MOV     AX,DS
00005642 1F                         POP     DS
00005643 AB                         STOSW                   ; store segment
                                                            ; ES:DI -> sf_firclus

                                ;   Generate pointer to, generate and store first cluster
                                ;   (irrelevant for devices)

00005644 56                         PUSH    SI              ; Save pointer to dir_first

                                ; 23/02/2024
                                %if 0

                                    MOVSW                   ; dir_first -> sf_firclus
                                                            ; DS:SI -> dir_size_l, ES:DI -> sf_time

                                ;   Copy time/date of last modification

                                    ;sub    si,6
                                    SUB     SI,dir_entry.dir_size_l - dir_entry.dir_time
                                %else
                                    ; 23/02/2024 - Retro DOS v5.0
                                    ; (PCDOS 7.1 IBMDOS.COM)
                                    ;;;
00005645 83C720                     add     di,32           ; SF_ENTRY.sf_chain ; first cluster (32 bit) !?
00005648 A5                         movsw                   ; first cluster, lw
                                    ;add    si,0FFF8h
00005649 83C6F8                     add     si,-8
0000564C AD                         lodsw   ; 27/06/2024    ; first cluster, hw
                                    ;
0000564D 08D2                       or      dl,dl
0000564F 7402                       jz      short FILE_SFT0 ; FAT32
00005651 31C0                       xor     ax,ax ; 0       ; clear hw of first cluster (invalid)
                                FILE_SFT0:
00005653 AB                         stosw
                                    ;add    di,0FFDEh
00005654 83C7DE                     add     di,-34          ; SF_ENTRY.sf_time
                                    ;;;
                                %endif
                                                            ; DS:SI->dir_time
00005657 A5                         MOVSW                   ; dir_time -> sf_time
                                                            ; DS:SI -> dir_date, ES:DI -> sf_date
00005658 A5                         MOVSW                   ; dir_date -> sf_date
                                                            ; DS:SI -> dir_first, ES:DI -> sf_size

                                ;   Generate and store file size (0 for devices)

00005659 AD                         LODSW                   ; skip dir_first, DS:SI -> dir_size_l
0000565A AD                         LODSW                   ; dir_size_l in AX, DS:SI -> dir_size_h
                                    ;MOV    CX,AX           ; dir_size_l in CX
                                    ; 23/02/2024
0000565B 91                         xchg    ax,cx
0000565C AD                         LODSW                   ; dir_size_h (size AX:CX), DS:SI -> ????
0000565D 08F6                       OR      DH,DH
0000565F 7904                       JNS     short FILE_SFT1
00005661 31C0                       XOR     AX,AX
00005663 89C1                       MOV     CX,AX           ; Devices are open ended
                                FILE_SFT1:
```

```
32598 00005665 91                       XCHG    AX,CX
32599 00005666 AB                       STOSW                   ; Low word of sf_size
32600 00005667 91                       XCHG    AX,CX
32601 00005668 AB                       STOSW                   ; High word of sf_size
32602                                                           ; ES:DI -> sf_position
32603                                 ; Initialize position to 0
32604
32605 00005669 31C0                     XOR     AX,AX
32606 0000566B AB                       STOSW
32607 0000566C AB                       STOSW                   ; sf_position
32608                                                           ; ES:DI -> sf_cluspos
32609
32610                                 ; Generate cluster optimizations for files
32611
32612 0000566D 08F6                     OR      DH,DH
32613 0000566F 784E                     JS      short DEV_SFT3
32614 00005671 AB                       STOSW                   ; sf_cluspos ; 19h
32615
32616                                 ; 23/02/2024
32617                                 %if 0
32618                                     ;mov    ax,[bx+1Ah]
32619                                     MOV     AX,[BX+dir_entry.dir_first]
32620                                     ; 19/05/2019
32621                                     ; MSDOS 3.3
32622                                     ;STOSW                  ; sf_lstclus ; 1Bh
32623                                     ; MSDOS 6.0
32624                                     PUSH    DI              ;AN004; save dirsec offset
32625                                     ;sub    di,1Bh
32626                                     SUB     DI,SF_ENTRY.sf_dirsec ;AN004; es:di -> SFT
32627                                     ;mov    [es:di+35h],ax
32628                                     MOV     [ES:DI+SF_ENTRY.sf_lstclus],AX       ;AN004; save it
32629                                     POP     DI              ;AN004; restore dirsec offset
32630                                 %else
32631                                     ; 23/02/2024 - Retro DOS v5.0
32632                                     ; (PCDOS 7.1 IBMDOS.COM)
32633                                     ;;;
32634                                     ;add    di,0FFF0h
32635 00005672 83C7F0                   add     di,-16
32636 00005675 AB                       stosw                   ; sf_cluspos_h (sf_firclus in MSDOS 5.0-6.22)
32637                                     ;add    di,SF_ENTRY.sf_dirsec
32638 00005676 83C70E                   add     di,14           ; sf_dirsec ; 27
32639 00005679 268B4510                 mov     ax,[es:di+10h] ; [ES:DI+SF_ENTRY.sf_chain] ; 43
32640                                                           ; first cluster (32 bit)
32641 0000567D 2689451A                 mov     [es:di+1Ah],ax
32642                                   ;mov    [es:di+SF_ENTRY.sf_lstclus-SF_ENTRY.sf_dirsec],ax ; 53
32643 00005681 268B4512                 mov     ax,[es:di+12h] ; [ES:DI+SF_ENTRY.sf_chain+2] ; 45
32644 00005685 2689451C                 mov     [es:di+1Ch],ax
32645                                   ;mov    [es:di+SF_ENTRY.sf_lstclus+2-SF_ENTRY.sf_dirsec],ax ; 55
32646                                   ;;;
32647                                 %endif
32648
32649                                 ; DOS 3.3  FastOpen  6/13/86
32650
32651 00005689 1E                       PUSH    DS
32652
32653                                 ;hkn; SS is DOSDATA
32654 0000568A 16                       push    ss
32655 0000568B 1F                       pop     ds
32656                                   ;test   byte [FastOpenFlg],4
32657 0000568C F606[4611]04             TEST    byte [FastOpenFlg],Special_Fill_Set
32658 00005691 7411                     JZ      short Not_FastOpen
32659
32660                                 ;hkn; FastOpen_Ext_Info is in DOSDATA
32661 00005693 BE[4711]                 MOV     SI,FastOpen_Ext_Info
32662
32663                                   ;mov    ax,[si+1]
32664 00005696 8B4401                   MOV     AX,[SI+FEI.dirsec]
32665 00005699 AB                       STOSW                   ; sf_dirsec
32666                                   ; MSDOS 6.0
32667                                   ;mov    ax,[si+3]
32668 0000569A 8B4403                   MOV     AX,[SI+FEI.dirsec+2]
32669                                   ;;; changed for >32mb
32670 0000569D AB                       STOSW                   ; sf_dirsec
32671                                   ; 19/08//2018
32672 0000569E 8A04                     mov     al,[SI]
32673                                   ;MOV    AL,[SI+FEI.dirpos] ; mov al,[SI+0]
32674 000056A0 AA                       STOSB                   ; sf_dirpos
32675 000056A1 1F                       POP     DS
32676                                   ;JMP    short Next_Name
32677                                   ; 24/09/2023
32678 000056A2 EB1E                     jmp     short FILE_SFT2       ; cf=0 (after 'test' instruction)
32679
32680                                 ; DOS 3.3  FastOpen  6/13/86
32681
32682                                 Not_FastOpen:
32683                                     ;POP    DS              ; normal path
32684
32685                                 ;hkn; SS override
32686                                     ;MOV    SI,[SS:CURBUF] ; DS:SI->buffer header
32687                                     ; 16/12/2022
32688                                     ; 28/07/2019
32689 000056A4 8B36[E205]               mov     si,[CURBUF]
32690 000056A8 1F                       pop     ds
32691                                     ; 07/12/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
32692                                     ;pop    ds
32693                                     ;mov    si,[ss:CURBUF]
32694
32695                                     ;mov    ax,[si+6]
32696 000056A9 8B4406                   MOV     AX,[SI+BUFFINFO.buf_sector]   ;F.C. >32mb ;AN000;
32697 000056AC AB                       STOSW                   ; sf_dirsec    ;F.C. >32mb ;AN000;
32698                                   ; 19/05/2019
32699                                   ; MSDOS 6.0
32700                                   ;mov    ax,[si+8]
32701 000056AD 8B4408                   MOV     AX,[SI+BUFFINFO.buf_sector+2] ;F.C. >32mb ;AN000;
32702 000056B0 AB                       STOSW                   ; sf_dirsec    ;F.C. >32mb ;AN000;
32703
32704 000056B1 89D8                     MOV     AX,BX
32705                                   ;;;add  si,16   ; MSDOS 3.3
32706                                   ;;add   si,20   ; MSDOS 6.0
32707                                   ;add    si,24   ; PCDOS 7.1 ; 23/02/2024
32708 000056B3 83C618                   ADD     SI,BUFINSIZ    ; DS:SI-> start of data in buffer
32709 000056B6 29F0                     SUB     AX,SI          ; AX = BX relative to start of sector
32710                                   ;mov    cl,32
32711 000056B8 B120                     MOV     CL,dir_entry.size
32712 000056BA F6F1                     DIV     CL
32713 000056BC AA                       STOSB                   ; sf_dirpos
32714                                 Next_Name:
32715 000056BD EB03                     JMP     SHORT FILE_SFT2
32716
32717                                   ; 24/09/2023
32718                                   ; cf=0 (after 'or' instruction)
32719                                 DEV_SFT3:
32720                                     ;add    di,7
32721 000056BF 83C707                   ADD     DI,SF_ENTRY.sf_name-SF_ENTRY.sf_cluspos
```

```
                                FILE_SFT2:

                                ; Copy in the object's name

000056C2 89DE                       MOV     SI,BX           ; DS:SI points to dir_name
000056C4 B90B00                     MOV     CX,11
000056C7 F3A4                       REP     MOVSB           ; sf_name
000056C9 5E                         POP     SI              ; recover DS:SI -> dir_first

                                ;hkn; SS is DOSDATA
000056CA 16                         push    ss
000056CB 1F                         pop     ds
                                    ; 24/09/2023
                                    ; cf=0
                                    ;CLC
000056CC C3                         retn

                                ;--------------------------------------------------------------------------
                                ;
                                ; Procedure Name : FREEENT
                                ;
                                ; Inputs:
                                ;       ES:BP -> DPB
                                ;       [CURBUF] Set
                                ;       [CURBUF+2]:BX points to directory entry
                                ;       [CURBUF+2]:SI points to above dir_first
                                ; Function:
                                ;       Free the cluster chain for the entry if present
                                ; Outputs:
                                ;       Carry set if error (currently user FAILed to I 24)
                                ;       (NOTE dir_firclus and dir_size_l/h are wrong)
                                ; DS BX SI ES BP preserved (BX,SI in meaning, not value) others destroyed
                                ;--------------------------------------------------------------------------

                                    ; 25/11/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
                                    ; 24/02/2024 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMDOS.COM)

                                FREEENT:
000056CD 1E                         PUSH    DS
000056CE C53E[E205]                 LDS     DI,[CURBUF]
000056D2 8B0C                       MOV     CX,[SI]         ; Get pointer to clusters
                                    ; 24/02/2023 (PCDOS 7.1 IBMDOS.COM)
                                    ;;;
000056D4 8B44FA                     mov     ax,[si-6]       ; hw of first cluster (dir_first)
                                    ;;;
                                    ; 19/05/2019 - Retro DOS v4.0
                                    ; MSDOS 6.0
                                    ;mov    dx,[di+8] ; 24/02/2024
000056D7 8B5508                     MOV     DX,[DI+BUFFINFO.buf_sector+2] ;F.C. >32mb  ;AN000;
                                ;hkn; SS override
000056DA 368916[0706]               MOV     [SS:HIGH_SECTOR],DX         ;F.C. >32mb  ;AN000;
                                    ;mov    dx,[di+6] ; 24/02/2024
000056DF 8B5506                     MOV     DX,[DI+BUFFINFO.buf_sector]
000056E2 1F                         POP     DS

                                    ; 24/02/2023 (PCDOS 7.1 IBMDOS.COM)
                                    ;;;
                                    ;cmp    word [bp+0Fh],0
000056E3 26837E0F00                 cmp     word [es:bp+DPB.FAT_SIZE],0
000056E8 7515                       jnz     short freeent2 ; not FAT32
                                    ;cmp    ax,0
000056EA 09C0                       or      ax,ax
000056EC 7505                       jnz     short freeent1
                                    ;
000056EE 83F902                     cmp     cx,2
000056F1 7242                       jb      short RET1      ; Was 0 length file (or mucked Firclus if AX:CX=1)
                                freeent1:
                                    ;cmp    ax,[es:bp+2Fh]
000056F3 263B462F                   cmp     ax,[es:bp+DPB.LAST_CLUSTER+2]
000056F7 7511                       jne     short freeent3
                                    ;cmp    cx,[es:bp+2Dh]
000056F9 263B4E2D                   cmp     cx,[es:bp+DPB.LAST_CLUSTER]
000056FD EB0B                       jmp     short freeent3
                                freeent2:
000056FF 31C0                       xor     ax,ax
                                    ;;;

00005701 83F902                     CMP     CX,2
00005704 722F                       JB      short RET1      ; Was 0 length file (or mucked Firclus if CX=1)

                                    ;cmp    cx,[es:bp+0Dh]
00005706 263B4E0D                   CMP     CX,[ES:BP+DPB.MAX_CLUSTER]
                                freeent3:   ; 24/02/2024
                                    ;JA     short RET1      ; Treat like zero length file (firclus mucked)
0000570A 7718                       ja      short freeent_retn ; 24/02/2024
0000570C 29FB                       SUB     BX,DI
0000570E 53                         PUSH    BX              ; Save offset
0000570F FF36[0706]                 PUSH    word [HIGH_SECTOR] ;F.C. >32mb       ;AN000;
00005713 52                         PUSH    DX              ; Save sector number
00005714 89CB                       MOV     BX,CX
                                    ; 24/02/2023 (PCDOS 7.1 IBMDOS.COM)
                                    ;;;
00005716 A3[E80A]                   mov     [CLUSTNUM_HW],ax
                                    ;;;
00005719 E8B404                     call    RELEASE         ; Free any data allocated
0000571C 5A                         POP     DX
0000571D 8F06[0706]                 POP     word [HIGH_SECTOR] ;F.C. >32mb       ;AN000;
00005721 7302                       JNC     short GET_BUF_BACK
00005723 5B                         POP     BX
                                freeent_retn:
00005724 C3                         retn                    ; Screw up

                                GET_BUF_BACK:
                                    ; 22/09/2023
                                    ;;mov   byte [ALLOWED],18h
                                    ;MOV    byte [ALLOWED],Allowed_RETRY+Allowed_FAIL ; *
                                    ;XOR    AL,AL ; *
                                    ;call   GETBUFFR        ; Get sector back
00005725 E80112                     call    GETBUFFER ; *   ; pre read

00005728 5B                         POP     BX              ; Get offset back
00005729 72F9                       jc      short freeent_retn
0000572B E850E8                     call    SET_BUF_AS_DIR
0000572E 031E[E205]                 ADD     BX,[CURBUF]     ; Correct it for new buffer

                                    ;MOV    SI,BX
                                    ;;add   si,1Ah
                                    ;ADD    SI,dir_entry.dir_first; Get corrected SI
                                    ; 24/02/2024 (PCDOS 7.1 IBMDOS.COM)
                                    ;lea    si,[bx+1Ah]
00005732 8D771A                     lea     si,[bx+dir_entry.dir_first]
                                RET1:
00005735 F8                         CLC
00005736 C3                         retn
```

```
                                ; 23/02/2024 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMDOS.COM)
                                %if 0

                                ;--------------------------------------------------------------------------
                                ;
                                ; Procedure Name : CHECK_VIRT_OPEN
                                ;
                                ; CHECK_VIRT_OPEN checks to see if we had performed a "virtual open" (by
                                ; examining the flag [VIRTUAL_OPEN] to see if it is 1). If we did, then
                                ; it calls Dev_Close_SFT to decrement the ref. count. It also resets the
                                ; flag [VIRTUAL_OPEN].
                                ; No registers affected (including flags).
                                ; On input, [THISSFT] points to current SFT.
                                ;
                                ;--------------------------------------------------------------------------

                                CHECK_VIRT_OPEN:
                                    PUSH    AX
                                    lahf                    ; preserve flags
                                    CMP     byte [VIRTUAL_OPEN],0
                                    JZ      short ALL_CLOSED
                                    mov     byte [VIRTUAL_OPEN],0 ; reset flag
                                    push    es
                                    push    di
                                    LES     DI,[THISSFT]
                                    call    DEV_CLOSE_SFT
                                    pop     di
                                    pop     es

                                ALL_CLOSED:
                                    sahf                    ; restore flags
                                    POP     AX
                                    retn

                                %endif

                                ;==========================================================================
                                ; ROM.ASM, MSDOS 6.0, 1991
                                ;==========================================================================
                                ; 29/07/2018 - Retro DOS v3.0
                                ; 20/05/2019 - Retro DOS v4.0
                                ; 10/02/2024 - Retro DOS v5.0

                                ;   TITLE   ROM - Miscellaneous routines
                                ;   NAME    ROM

                                ;** Misc Low level routines for doing simple FCB computations, Cache
                                ;       reads and writes, I/O optimization, and FAT allocation/deallocation
                                ;
                                ;   SKPCLP
                                ;   FNDCLUS
                                ;   BUFSEC
                                ;   BUFRD
                                ;   BUFWRT
                                ;   NEXTSEC
                                ;   OPTIMIZE
                                ;   FIGREC
                                ;   ALLOCATE
                                ;   RESTFATBYT
                                ;   RELEASE
                                ;   RELBLKS
                                ;   GETEOF
                                ;
                                ;   Modification history:
                                ;
                                ;           Created: ARR 30 March 1983
                                ;                M039: DB 10/25/90 - Disk read/write optimization.

                                ;Break   <FNDCLUS -- Skip over allocation units>
                                ;--------------------------------------------------------------------------
                                ;
                                ; Procedure Name : FNDCLUS
                                ;
                                ; Inputs:
                                ;       CX = No. of clusters to skip
                                ;       ES:BP = Base of drive parameters
                                ;       [THISSFT] point to SFT
                                ; Outputs:
                                ;       BX = Last cluster skipped to
                                ;       CX = No. of clusters remaining (0 unless EOF)
                                ;       DX = Position of last cluster
                                ;       Carry set if error (currently user FAILed to I 24)
                                ; DI destroyed. No other registers affected.
                                ;--------------------------------------------------------------------------

                                    ;;;
                                    ; 10/02/2024 - Retro DOS v5.0
                                    ; (PCDOS 7.1 IBMDOS.COM)
                                FNDCLUS_X:
00005737 8B0E[BC05]                 mov     cx,[CLUSNUM]
0000573B 8B16[DE0A]                 mov     dx,[CLUSNUM_HW]
                                    ;;;

                                ; 20/05/2019 - Retro DOS v4.0
                                ; DOSCODE:8BF2h (MSDOS 6.21, MSDOS.SYS)
                                ; 25/11/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
                                ; DOSCODE:8BB7h (MSDOS 5.0, MSDOS.SYS)

                                ; 25/02/2024
                                ; 10/02/2024 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMDOS.COM)
                                ; DOSCODE:9801h (PCDOS 7.1 IBMDOS.COM)

                                FNDCLUS:
0000573F 06                         PUSH    ES
00005740 C43E[9E05]                     LES     DI,[THISSFT]            ; setup addressability to SFT

                                    ;;;
                                    ; 10/02/2024 (PCDOS 7.1 IBMDOS.COM)
00005744 8916[F80A]                 mov     [CLSKIP_HW],dx
                                    ;mov    bx,[es:di+37h]
00005748 268B5D37                   mov     bx,[es:di+SF_ENTRY.sf_lstclus+2] ; 24/02/2024
0000574C 891E[E80A]                 mov     [CLUSTNUM_HW],bx
00005750 09DB                       or      bx,bx ; *
00005752 268B550B                   mov     dx,[es:di+0Bh] ; [ES:DI+SF_ENTRY.sf_firclus] ; MSDOS 5.0-6.22
                                        ;mov       dx,[es:di+SF_ENTRY.sf_cluspos_hw] ; PCDOS 7.1
00005756 8916[E20A]                 mov     [LASTPOS_HW],dx
                                    ;;;

                                    ;;mov   bx,[es:di+1Bh] ; MSDOS 3.3
                                    ;mov    bx,[es:di+35h] ; MSDOS 6.0
0000575A 268B5D35                   MOV     BX,[ES:DI+SF_ENTRY.sf_lstclus]
                                    ;mov    dx,[es:di+19h]
0000575E 268B5519                       MOV     DX,[ES:DI+SF_ENTRY.sf_cluspos]
```

```
                                        ; 10/02/2024
                                        ;OR     BX,BX
                                        ;JZ     short NOCLUS
                                        ;;;
                                        ; 10/02/2024 (PCDOS 7.1 IBMDOS.COM)
00005762 7506                           jnz     short fndclus_1 ; *
00005764 09DB                           or      bx,bx
00005766 7502                           jnz     short fndclus_1
00005768 EB6A                           jmp     NOCLUS

                                    fndclus_1:
                                        ;;;

                                    ; 10/02/2024
                                    %if 0
                                            SUB CX,DX
                                            JNB short FINDIT
                                        ADD     CX,DX
                                        XOR     DX,DX
                                        ;mov    bx,[es:di+0Bh]
                                        MOV     BX,[ES:DI+SF_ENTRY.sf_firclus]
                                    FINDIT:
                                            POP ES
                                        JCXZ    RET9
                                    %else
                                        ;;;
                                        ; 10/02/2024 (PCDOS 7.1 IBMDOS.COM)
0000576A 29D1                           sub     cx,dx
0000576C 50                             push    ax
0000576D A1[E20A]                       mov     ax,[LASTPOS_HW]
00005770 1906[F80A]                     sbb     [CLSKIP_HW],ax
00005774 58                             pop     ax
00005775 731C                           jnb     short FINDIT
00005777 8B1E[E20A]                     mov     bx,[LASTPOS_HW]
0000577B 01D1                           add     cx,dx
0000577D 111E[F80A]                     adc     [CLSKIP_HW],bx
00005781 31D2                           xor     dx,dx
00005783 8916[E20A]                     mov     [LASTPOS_HW],dx ; 0
00005787 268B5D2D                       mov     bx,[es:di+2Dh] ; [ES:DI+SF_ENTRY.sf_chain+2] ; MSDOS 5.0-6.22
                                                                ; [ES:DI+SF_ENTRY.sf_fcluster+2] ; PCDOS 7.1
0000578B 891E[E80A]                     mov     [CLUSTNUM_HW],bx
0000578F 268B5D2B                       mov     bx,[es:di+2Bh] ; [ES:DI+SF_ENTRY.sf_chain] ; MSDOS 5.0-6.22
                                                                ; [ES:DI+SF_ENTRY.sf_fcluster] ; PCDOS 7.1
                                    FINDIT:
00005793 07                             pop     es
00005794 3B0E[F80A]                     cmp     cx,[CLSKIP_HW]
00005798 7502                           jnz     short SKPCLP
0000579A E337                           jcxz    RET9    ; cf=0
                                        ;;;
                                    %endif

                                        ;entry  SKPCLP
                                    SKPCLP:

                                    ; 10/02/2024
                                    %if 0
                                        call    UNPACK
                                            jc  short fndclus_retn    ; retc

                                        ; 09/09/2018

                                        ; MSDOS 3.3
                                        ;push   bx
                                        ;mov    bx,di
                                        ;call   IsEOF
                                        ;pop    bx
                                        ;jae    short RET9

                                        ; 20/05/2019 - Retro DOS v4.0

                                        ; MSDOS 6.0
                                        xchg    bx,di
                                        call    IsEOF
                                        xchg    bx,di
                                        jae     short RET9

                                            XCHG    BX,DI
                                            INC     DX

                                        LOOP    SKPCLP                  ; RMFS
                                    %else
                                        ;;;
                                        ; 10/02/2024 (PCDOS 7.1 IBMDOS.COM)

                                        ;push   word [LASTPOS_HW]
                                        ;push   dx
                                        ;push   word [CLSKIP_HW]
                                        ;push   cx

0000579C E8420B                         call    UNPACK

                                        ;pop    cx
                                        ;pop    word [CLSKIP_HW]
                                        ;pop    dx
                                        ;pop    word [LASTPOS_HW]

0000579F 7242                           jc      short fndclus_retn

000057A1 FF36[E80A]                     push    word [CLUSTNUM_HW]
000057A5 53                             push    bx
000057A6 89FB                           mov     bx,di
000057A8 8B3E[EC0A]                     mov     di,[CCONTENT_HW]
000057AC 893E[E80A]                     mov     [CLUSTNUM_HW],di
000057B0 E8050B                         call    IsEOF
000057B3 7206                           jc      short SKPCLP2
000057B5 5B                             pop     bx
000057B6 8F06[E80A]                     pop     word [CLUSTNUM_HW]
                                        ;jmp    short RET9
                                        ; 25/02/2024
000057BA C3                             retn

                                    SKPCLP2:
000057BB 83C404                         add     sp,4
000057BE 42                             inc     dx
000057BF 7504                           jnz     short SKPCLP3
000057C1 FF06[E20A]                     inc     word [LASTPOS_HW]
                                    SKPCLP3:
000057C5 83E901                         sub     cx,1
000057C8 831E[F80A]00                   sbb     word [CLSKIP_HW],0
000057CD 75CD                           jnz     short SKPCLP          ; loop
000057CF 09C9                           or      cx,cx
000057D1 75C9                           jnz     short SKPCLP          ; loop
                                        ;;;
                                    %endif
```

```
33094
33095                                   RET9:
33096                                       ;CLC
33097                                       ; 25/02/2024
33098                                       ; cf=0
33099 000057D3 C3                           retn
33100
33101                                   NOCLUS:
33102 000057D4 07                           POP ES
33103
33104                                   ; 10/02/2024
33105                                   %if 0
33106                                           INC CX
33107                                           DEC DX
33108                                   %else
33109                                       ;;;
33110                                       ; 10/02/2024 (PCDOS 7.1 IBMDOS.COM)
33111                                       ;push   di
33112                                       ;xor    di,di ; 0
33113                                       ;add    cx,1
33114                                       ;adc    [CLSKIP_HW],di ; CLSKIP_HW:BX = Last cluster skipped to
33115                                       ;sub    dx,1
33116                                       ;sbb    [LASTPOS_HW],di       ; LASTPOS_HW:DX = Position of last cluster
33117                                       ;pop    di
33118                                       ;;;
33119                                       ; 25/02/2024 - Retro DOS v5.0
33120 000057D5 4A                           dec     dx
33121 000057D6 7904                         jns     short noclus1
33122 000057D8 FF0E[E20A]                   dec     word [LASTPOS_HW]
33123                                   noclus1:
33124 000057DC 41                           inc     cx
33125 000057DD 7504                         jnz     short fndclus_retn
33126 000057DF FF06[F80A]                   inc     word [CLSKIP_HW]
33127                                   %endif
33128                                       ; 25/02/2024
33129                                       ; cf=0
33130                                           ;CLC
33131
33132                                   fndclus_retn:
33133 000057E3 C3                              retn
33134
33135                                   ;Break  <BUFSEC -- BUFFER A SECTOR AND SET UP A TRANSFER>
33136                                   ;--------------------------------------------------------------------------
33137                                   ;
33138                                   ; Procedure Name : BUFSEC
33139                                   ;
33140                                   ; Inputs:
33141                                   ;       AH = priority of buffer
33142                                   ;       AL = 0 if buffer must be read, 1 if no pre-read needed
33143                                   ;       ES:BP = Base of drive parameters
33144                                   ;       [CLUSNUM] = Physical cluster number
33145                                   ;       [SECCLUSPOS] = Sector position of transfer within cluster
33146                                   ;       [BYTCNT1] = Size of transfer
33147                                   ; Function:
33148                                   ;       Insure specified sector is in buffer, flushing buffer before
33149                                   ;       read if necessary.
33150                                   ; Outputs:
33151                                   ;       ES:DI = Pointer to buffer
33152                                   ;       SI = Pointer to transfer address
33153                                   ;       CX = Number of bytes
33154                                   ;       [NEXTADD] updated
33155                                   ;       [TRANS] set to indicate a transfer will occur
33156                                   ;       Carry set if error (user FAILed to I 24)
33157                                   ;--------------------------------------------------------------------------
33158
33159                                       ; 25/02/2024 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMDOS.COM)
33160                                       ; PCDOS 7.1 IBMDOS.COM - DOSCODE:98C2h
33161
33162                                       ; (MSDOS 5.0 MSDOS.SYS - DOSCODE:8BF1h) - Retro DOS v4.0 (& v4.1)
33163                                       ; (MSDOS 6.22 MSDOS.SYS - DOSCODE:8C2Ch) - Retro DOS v4.2
33164                                       ; (Windows ME IO.SYS - BIOSCODE:0BE33h)
33165
33166                                   BUFSEC:
33167                                       ; 25/02/2024
33168                                       ;;;
33169                                       ;push   word [CLUSTNUM_HW]
33170 000057E4 8B16[DE0A]                  mov     dx,[CLUSNUM_HW]
33171                                       ;mov    [CLUSTNUM_HW],dx
33172 000057E8 8716[E80A]                  xchg    dx,[CLUSTNUM_HW]
33173 000057EC 52                          push    dx      ; [CLUSTNUM_HW]
33174                                       ;;;
33175 000057ED 8B16[BC05]                      MOV     DX,[CLUSNUM]
33176 000057F1 8A1E[7305]                      MOV     BL,[SECCLUSPOS]
33177                                       ;mov    byte [ALLOWED],38h
33178 000057F5 C606[4B03]38                    MOV     byte [ALLOWED],Allowed_FAIL+Allowed_RETRY+Allowed_IGNORE
33179 000057FA E89A01                          CALL    FIGREC
33180 000057FD E83011                      call    GETBUFFR
33181                                       ; 25/02/2024
33182                                       ;;;
33183 00005800 8F06[E80A]                  pop     word [CLUSTNUM_HW]
33184                                       ;;;
33185 00005804 72DD                            jc  short fndclus_retn
33186
33187 00005806 C606[7405]01                    MOV     BYTE [TRANS],1    ; A transfer is taking place
33188 0000580B 8B36[B805]                      MOV     SI,[NEXTADD]
33189 0000580F 89F7                            MOV     DI,SI
33190 00005811 8B0E[D205]                      MOV     CX,[BYTCNT1]
33191 00005815 01CF                            ADD     DI,CX
33192 00005817 893E[B805]                      MOV     [NEXTADD],DI
33193 0000581B C43E[E205]                      LES     DI,[CURBUF]
33194                                       ;or     byte [es:di+5],8
33195 0000581F 26804D0508                      OR      byte [ES:DI+BUFFINFO.buf_flags],buf_isDATA
33196                                       ;;;lea  di,[di+16] ; MSDOS 3.3
33197                                       ;;lea   di,[di+20] ; MSDOS 6.0
33198                                           ;lea       di,[di+24] ; PCDOS 7.1  ; 25/02/2024
33199 00005824 8D7D18                      LEA     DI,[DI+BUFINSIZ]        ; Point to buffer
33200 00005827 033E[CC05]                      ADD     DI,[BYTSECPOS]
33201 0000582B F8                              CLC ; (not necessary!?) ; 25/02/2024
33202 0000582C C3                              retn
33203
33204                                   ;Break   <BUFRD, BUFWRT -- PERFORM BUFFERED READ AND WRITE>
33205
33206                                   ;--------------------------------------------------------------------------
33207                                   ;
33208                                   ; Procedure Name : BUFRD
33209                                   ;
33210                                   ; Do a partial sector read via one of the system buffers
33211                                   ; ES:BP Points to DPB
33212                                   ; Carry set if error (currently user FAILed to I 24)
33213                                   ;
33214                                   ; DS - set to DOSDATA
33215                                   ;
33216                                   ;--------------------------------------------------------------------------
33217
```

```
                                    ; 25/02/2024 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMDOS.COM)
                                    ; 25/11/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
                                    ; 20/05/2019 - Retro DOS v4.0
                                BUFRD:
0000582D 06                             PUSH       ES
0000582E 31C0                           xor ax,ax                   ; pre-read sector
00005830 E8B1FF                        CALL    BUFSEC
00005833 7303                           JNC short BUF_OK ; ds=ss

                                BUF_IO_FAIL:                                ; this label used by BUFWRT also
00005835 07                             POP ES
00005836 EB2D                           JMP     SHORT RBUFPLACED ; ds=ss

                                BUF_OK:
00005838 8CC3                           MOV     BX,ES
0000583A 8E06[2E03]                     MOV     ES,[DMAADD+2]
0000583E 8EDB                           MOV     DS,BX
00005840 87FE                       XCHG    DI,SI
00005842 D1E9                           SHR     CX,1
                                ;M039
                                    ; MSDOS 3.3
                                    ;JNC    short EVENRD
                                    ;MOVSB
                                ;EVENRD:
                                    ;REP     MOVSW

                                ;   CX = # of whole WORDs ; CF=1 if odd # of bytes.
                                ;       DS:SI-> Source within Buffer.
                                ;       ES:DI-> Destination within Transfer memory block.

                                    ; MSDOS 6.0
00005844 F3A5                       rep     movsw                   ;Copy Buffer to Transfer memory.
                                    ;adc    cx,0                      ;CX=1 if odd # of bytes, else CX=0.
                                    ;rep    movsb                     ;Copy last byte.
                                    ; 16/12/2022
00005846 7301                       jnc     short EVENRD ; **** 20/05/2019
00005848 A4                         movsb ; ****
                                    ; 25/11/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
                                    ;adc    cx,0
                                    ;rep    movsb
                                ;M039
                                EVENRD: ; ****
00005849 07                             POP ES
                                ;hkn; SS override
0000584A 36C53E[E205]                   LDS     DI,[SS:CURBUF]
                                    ;;;lea  bx,[di+16]
                                    ;;lea   bx,[di+20] ; MSDOS 6.0
                                    ;lea    bx,[di+24] ; PCDOS 7.1; 25/02/2024
0000584F 8D5D18                     LEA     BX,[DI+BUFINSIZ]
00005852 29DE                           SUB     SI,BX                   ; Position in buffer
00005854 E87B10                         call        PLACEBUF
                                    ;cmp    si,[es:bp+2]
00005857 263B7602                   CMP     SI,[ES:BP+DPB.SECTOR_SIZE] ; Read Last byte?
0000585B 7205                           JB  short RBUFPLACEDC ; ds<>ss ; No, leave buf where it is
                                ;M039
                                    ; MSDOS 3.3
                                    ;call   PLACEHEAD               ; Make it prime candidate for chucking
                                                                        ;  even though it is MRU.
                                        ; MSDOS 6.0
0000585D 36893E[6D00]               MOV     [ss:BufferQueue],DI    ; Make it prime candidate for
                                ;M039                                       ; chucking even though it is MRU.

                                RBUFPLACEDC:
00005862 F8                             CLC
                                ;RBUFPLACED:
00005863 16                         push    ss
00005864 1F                         pop     ds
                                RBUFPLACED: ; 25/02/2024 (ds=ss)
00005865 C3                             retn

                                ;--------------------------------------------------------------------------
                                ;
                                ; Procedure : BUFWRT
                                ;
                                ; Do a partial sector write via one of the system buffers
                                ; ES:BP Points to DPB
                                ; Carry set if error (currently user FAILed to I 24)
                                ;
                                ; DS - set to DOSDATA
                                ;
                                ;--------------------------------------------------------------------------

                                    ; 25/11/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
                                    ; 20/05/2019 - Retro DOS v4.0
                                BUFWRT:
                                    ;MOV    AX,[SECPOS]
                                    ; MSDOS 6.0
                                    ;ADD    AX,1                    ; Set for next sector
                                    ;MOV    [SECPOS],AX             ;F.C. >32mb    ;AN000;
                                    ;ADC    word [SECPOS+2],0       ;F.C. >32mb    ;AN000;
                                    ; 24/09/2023
00005866 FF06[C405]                 inc     word [SECPOS]
0000586A 7504                       jnz     short bufw_secpos
0000586C FF06[C605]                 inc     word [SECPOS+2]
                                bufw_secpos:
00005870 A1[C605]                   MOV     AX,[SECPOS+2]           ;F.C. >32mb    ;AN000;
00005873 3B06[CA05]                 CMP     AX,[VALSEC+2]           ;F.C. >32mb    ;AN000;
00005877 B001                       MOV     AL,1                    ;F.C. >32mb    ;AN000;
00005879 770F                       JA      short NOREAD            ;F.C. >32mb    ;AN000;
0000587B 720B                       JB      short _doread           ;F.C. >32mb    ;AN000;
0000587D A1[C405]                   MOV     AX,[SECPOS]             ;F.C. >32mb    ;AN000;

                                    ; MSDOS 3.3
                                    ;INC    AX
                                    ;MOV    [SECPOS],AX ; 09/09/2018

                                    ; 20/05/2019
                                    ; MSDOS 3.3 & MSDOS 6.0
00005880 3B06[C805]                 CMP     AX,[VALSEC]             ; Has sector been written before?
00005884 B001                       MOV     AL,1
00005886 7702                       JA      short NOREAD            ; Skip preread if SECPOS>VALSEC
                                _doread:
00005888 30C0                       XOR     AL,AL
                                NOREAD:
0000588A 06                         PUSH    ES
0000588B E856FF                     CALL    BUFSEC
0000588E 72A5                       JC      short BUF_IO_FAIL
00005890 8E1E[2E03]                 MOV     DS,[DMAADD+2]
00005894 D1E9                       SHR     CX,1
                                ;M039
                                    ; MSDOS 3.3
                                    ;JNC    short EVENWRT ; 09/09/2018
                                    ;MOVSB
                                ;EVENWRT:
```

```
                                    ;REP    MOVSW
                                
                                ;   CX = # of whole WORDs; CF=1 if odd # of bytes.
                                ;   DS:SI-> Source within Transfer memory block.
                                ;   ES:DI-> Destination within Buffer.
                                
                                    ; MSDOS 6.0
00005896 F3A5                       rep     movsw                   ;Copy Transfer memory to Buffer.
                                    ;adc    cx,0                    ;CX=1 if odd # of bytes, else CX=0.
                                    ;rep    movsb                   ;Copy last byte.
                                    ; 16/12/2022
00005898 7301                       jnc     short EVENWRT ; **** 20/05/2019
0000589A A4                         movsb ; ****
                                    ; 25/11/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
                                    ;adc    cx,0
                                    ;rep    movsb
                                ;M039
                                EVENWRT: ; ****
0000589B 07                         POP     ES
                                
                                ;hkn; SS override
0000589C 36C51E[E205]               LDS     BX,[SS:CURBUF]
                                
                                    ; MSDOS 6.0
000058A1 F6470540                   TEST    byte [BX+BUFFINFO.buf_flags],buf_dirty
                                                                    ;LB. if already dirty          ;AN000;
000058A5 7507                       JNZ     short yesdirty10        ;LB. don't increment dirty count ;AN000;
000058A7 E82013                     call    INC_DIRTY_COUNT                 ;LB.                            ;AN000;
                                
                                    ;or     byte [bx+5],40h
000058AA 804F0540                   OR      byte [BX+BUFFINFO.buf_flags],buf_dirty
                                yesdirty10:
                                    ;;;lea  si,[bx+16]
                                    ;;lea   si,[bx+20] ; MSDOS 6.0
                                    ;lea    si,[bx+24] ; PCDOS 7.1; 25/02/2024
000058AE 8D7718                     LEA     SI,[BX+BUFINSIZ]
000058B1 29F7                       SUB     DI,SI                   ; Position in buffer
                                ;M039
                                    ; MSDOS 3.3
                                    ;MOV    SI,DI
                                    ;MOV    DI,BX
                                    ;call   PLACEBUF
                                    ;;cmp   si,[es:bp+2]
                                    ;CMP    SI,[ES:BP+DPB.SECTOR_SIZE] ; Written last byte?
                                    ;JB     short WBUFPLACED        ; No, leave buf where it is
                                    ;call   PLACEHEAD               ; Make it prime candidate for chucking
                                                                    ;  even though it is MRU.
                                    ; 10/02/2024
000058B3 16                         push    ss
000058B4 1F                         pop     ds
                                
                                    ; MSDOS 6.0
                                    ;cmp    di,[es:bp+2]
000058B5 263B7E02                   CMP     di,[ES:BP+DPB.SECTOR_SIZE] ; Written last byte?
000058B9 7204                       JB      short WBUFPLACED        ; No, leave buf where it is
                                
                                    ; 10/02/2024
                                    ;MOV    [ss:BufferQueue],BX     ; Make it prime candidate for
                                                                    ; chucking even though it is MRU.
000058BB 891E[6D00]                 mov     [BufferQueue],bx
                                ;M039
                                
                                WBUFPLACED:
000058BF F8                         CLC
                                    ; 10/02/2024
                                    ;push   ss
                                    ;pop    ds
000058C0 C3                         retn
                                
                                ;Break   <NEXTSEC -- Compute next sector to read or write>
                                ;--------------------------------------------------------------------------
                                ;
                                ; Procedure Name : NEXTSEC
                                ;
                                ; Compute the next sector to read or write
                                ; ES:BP Points to DPB
                                ;
                                ;--------------------------------------------------------------------------
                                
                                    ; 29/02/2024
                                    ; 26/02/2024 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMDOS.COM)
                                
                                NEXTSEC:
000058C1 F606[7405]FF               test    byte [TRANS],0FFh ; -1
                                    ;JZ     short CLRET
                                    ; 29/02/2024
000058C6 743D                       jz      short CLRET2
                                
000058C8 A0[7305]                   MOV     AL,[SECCLUSPOS]
000058CB FEC0                       INC     AL
                                    ;cmp    al,[es:bp+4]
000058CD 263A4604                   CMP     AL,[ES:BP+DPB.CLUSTER_MASK]
000058D1 762E                       JBE     short SAVPOS
                                
                                    ; 26/02/2024 (PCDOS 7.1 IBMDOS.COM)
                                    ;;;
000058D3 8B1E[DE0A]                 mov      bx,[CLUSNUM_HW]
000058D7 891E[E80A]                 mov     [CLUSTNUM_HW],bx
                                    ;;;
                                
000058DB 8B1E[BC05]                 MOV     BX,[CLUSNUM]
000058DF E8D609                     call    IsEOF
000058E2 7322                       JAE     short NONEXT
                                
000058E4 E8FA09                     call    UNPACK
                                    ;JC     short NONEXT
                                    ; 26/02/2024
000058E7 721E                       jc      short NONEXT2
                                clusgot:
                                    ; 26/02/2024 - Retro DOS v5.0
                                    ;;;
                                    ;push   di
                                    ;mov    di,[CCONTENT_HW]
                                    ;mov    [CLUSNUM_HW],di
                                    ;pop    di
000058E9 FF36[EC0A]                 push    word [CCONTENT_HW]
000058ED 8F06[DE0A]                 pop     word [CLUSNUM_HW]
                                    ;;;
000058F1 893E[BC05]                 MOV     [CLUSNUM],DI
000058F5 FF06[BA05]                 INC     word [LASTPOS]
                                    ; 26/02/2024 - Retro DOS v5.0
                                    ;;;
000058F9 7504                       jnz     short clusgot1
000058FB FF06[E20A]                 inc     word [LASTPOS_HW]
```

```
                                clusgot1:
                                    ;;;
000058FF B000                       MOV     AL,0
                                SAVPOS:
00005901 A2[7305]                   MOV     [SECCLUSPOS],AL
                                CLRET:
00005904 F8                         CLC
                                CLRET2:             ; 29/02/2024
00005905 C3                         retn
                                NONEXT:
00005906 F9                         STC
                                NONEXT2:    ; 26/02/2024
00005907 C3                         retn

                                ;Break      <OPTIMIZE -- DO A USER DISK REQUEST WELL>
                                ;--------------------------------------------------------------------------
                                ;
                                ; Procedure Name : OPTIMIZE
                                ;
                                ; Inputs:
                                ;     BX = Physical cluster
                                ;     CX = No. of records
                                ;     DL = sector within cluster
                                ;     ES:BP = Base of drive parameters
                                ;     [NEXTADD] = transfer address
                                ; Outputs:
                                ;     AX = No. of records remaining
                                ;     BX = Transfer address
                                ;     CX = No. or records to be transferred
                                ;     DX = Physical sector address (LOW)
                                ;     [HIGH_SECTOR] = Physical sector address (HIGH)
                                ;     DI = Next cluster
                                ;     [CLUSNUM] = Last cluster accessed
                                ;     [NEXTADD] updated
                                ;     Carry set if error (currently user FAILed to I 24)
                                ; ES:BP unchanged. Note that segment of transfer not set.
                                ;
                                ;--------------------------------------------------------------------------

                                    ; 25/11/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
                                    ; 27/02/2024 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMDOS.COM)

                                OPTIMIZE:
00005908 52                         PUSH    DX
                                    ; 27/02/2024 (PCDOS 7.1 IBMDOS.COM)
                                    ;;;
00005909 FF36[E80A]                 push    word [CLUSTNUM_HW]
                                    ;;;
0000590D 53                         PUSH    BX
                                    ;mov    al,[es:bp+4]
0000590E 268A4604                   MOV     AL,[ES:BP+DPB.CLUSTER_MASK]
00005912 FEC0                       INC     AL              ; Number of sectors per cluster
00005914 88C4                       MOV     AH,AL
00005916 28D0                       SUB     AL,DL           ; AL = Num of sectors left in first cluster
00005918 89CA                       MOV     DX,CX
                                    ;MOV    CX,0
                                    ; 25/11/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
                                    ; 16/12/2022
0000591A 31C9                       xor     cx,cx   ; sub cx,cx
                                OPTCLUS:
                                ; AL has number of sectors available in current cluster
                                ; AH has number of sectors available in next cluster
                                ; BX has current physical cluster
                                ; CX has number of sequential sectors found so far
                                ; DX has number of sectors left to transfer
                                ; ES:BP Points to DPB
                                ; ES:SI has FAT pointer

                                do_norm3:
0000591C E8C209                     call    UNPACK
0000591F 7263                       JC      short OP_ERR   ; ds=ss ; 27/02/2024
                                ;clusgot2:
00005921 00C1                       ADD     CL,AL
00005923 80D500                     ADC     CH,0
00005926 39D1                       CMP     CX,DX
00005928 735F                       JAE     short BLKDON
0000592A 88E0                       MOV     AL,AH
                                    ;INC    BX
                                    ; 27/02/2024 - Retro DOS v5.0 (PCDOS 7.1 IBMDOS.COM)
                                    ;;;
                                    ; 28/06/2024
0000592C 57                         push    di
                                    ;xor    di,di
                                    ;add    bx,1
                                    ;adc    [CLUSTNUM_HW],di
                                    ;
0000592D 43                         inc     bx
0000592E 7504                       jnz     short clusgot2
00005930 FF06[E80A]                 inc     word [CLUSTNUM_HW]
                                clusgot2:
00005934 8B3E[EC0A]                 mov     di,[CCONTENT_HW]
00005938 3B3E[E80A]                 cmp     di,[CLUSTNUM_HW]
                                    ; 28/06/2024
0000593C 5F                         pop     di
0000593D 7504                       jne     short clusgot3
0000593F 39DF                       cmp     di,bx
00005941 74D9                       je      short OPTCLUS
                                clusgot3:
                                    ;;;
                                    ;CMP    DI,BX
                                    ;JZ     short OPTCLUS
                                    ;DEC    BX
                                    ; 27/02/2024 - Retro DOS v5.0 (PCDOS 7.1 IBMDOS.COM)
                                    ;;;
                                    ;sub    bx,1
                                    ;sbb    word [CLUSTNUM_HW],0
                                    ;
00005943 4B                         dec     bx
00005944 7904                       jns     short FINCLUS
00005946 FF0E[E80A]                 dec     word [CLUSTNUM_HW]
                                    ;;;
                                FINCLUS:
                                    ;MOV    [CLUSNUM],BX   ; Last cluster accessed
                                    ; 27/02/2024 - Retro DOS v5.0 (PCDOS 7.1 IBMDOS.COM)
                                    ;;;
                                    ;push   bx
                                    ;add    [LASTPOS],bx
                                    ;mov    bx,[CLUSTNUM_HW]
                                    ;adc    [LASTPOS_HW],bx
                                    ;mov    [CLUSNUM_HW],bx
                                    ;pop    word [CLUSNUM]
                                    ;
0000594A 891E[BC05]                 mov     [CLUSNUM],bx
0000594E 011E[BA05]                 add     [LASTPOS],bx
```

```
33590 00005952 A1[E80A]                 mov     ax,[CLUSTNUM_HW]
33591 00005955 1106[E20A]               adc     [LASTPOS_HW],ax
33592 00005959 A3[DE0A]                 mov     [CLUSNUM_HW],ax
33593                                   ;;;
33594 0000595C 29CA                     SUB     DX,CX           ; Number of sectors still needed
33595 0000595E 52                       PUSH    DX
33596 0000595F 89C8                     MOV     AX,CX
33597                                   ;mul    word[ES:BP+2]
33598 00005961 26F76602                 MUL     word [ES:BP+DPB.SECTOR_SIZE]
33599                                                   ; Number of sectors times sector size
33600 00005965 8B36[B805]               MOV     SI,[NEXTADD]
33601 00005969 01F0                     ADD     AX,SI           ; Adjust by size of transfer
33602 0000596B A3[B805]                 MOV     [NEXTADD],AX
33603 0000596E 58                       POP     AX              ; Number of sectors still needed
33604 0000596F 5A                       POP     DX              ; Starting cluster
33605                                   ; 27/02/2024 (PCDOS 7.1 IBMDOS.COM)
33606                                   ;SUB    BX,DX           ; Number of new clusters accessed
33607                                   ;ADD    [LASTPOS],BX
33608                                   ;;;
33609 00005970 5B                       pop     bx              ; Starting cluster, hw
33610 00005971 891E[E80A]               mov     [CLUSTNUM_HW],bx
33611 00005975 2916[BA05]               sub     [LASTPOS],dx
33612 00005979 191E[E20A]               sbb     [LASTPOS_HW],bx
33613                                   ;;;
33614 0000597D 5B                       POP     BX              ; BL = sector position within cluster
33615 0000597E E81600                   call    FIGREC
33616 00005981 89F3                     MOV     BX,SI
33617                                   ; 24/09/2023
33618                                   ; cf=0 (at the return of FIGREC)
33619                                   ;CLC
33620 00005983 C3                       retn
33621                                 OP_ERR:
33622                                   ;ADD    SP,4
33623                                   ; 27/02/2024 (PCDOS 7.1 IBMDOS.COM)
33624                                   ;;;
33625 00005984 83C406                   add     sp,6
33626                                   ;;;
33627 00005987 F9                       STC
33628 00005988 C3                       retn
33629                                 BLKDON:
33630 00005989 29D1                     SUB     CX,DX           ; Number of sectors in cluster we don't want
33631 0000598B 28CC                     SUB     AH,CL           ; Number of sectors in cluster we accepted
33632 0000598D FECC                     DEC     AH              ; Adjust to mean position within cluster
33633 0000598F 8826[7305]               MOV     [SECCLUSPOS],AH
33634 00005993 89D1                     MOV     CX,DX           ; Anyway, make the total equal to the request
33635 00005995 EBB3                     JMP     SHORT FINCLUS
33636
33637                                 ;Break      <FIGREC -- Figure sector in allocation unit>
33638                                 ;-----------------------------------------------------------------------------
33639                                 ;
33640                                 ; Procedure Name : FIGREC
33641                                 ;
33642                                 ; Inputs:
33643                                 ;     DX = Physical cluster number
33644                                 ;     BL = Sector position within cluster
33645                                 ;     ES:BP = Base of drive parameters
33646                                 ; Outputs:
33647                                 ;     DX = physical sector number (LOW)
33648                                 ;     [HIGH_SECTOR] Physical sector address (HIGH)
33649                                 ; No other registers affected.
33650                                 ;
33651                                 ;-----------------------------------------------------------------------------
33652
33653                                     ; 10/06/2019
33654                                     ; 20/05/2019 - Retro DOS v4.0
33655                                     ; DOSCODE:8D96h (MSDOS 6.21, MSDOS.SYS)
33656                                     ; 25/11/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
33657                                     ; DOSCODE:8D5Bh (MSDOS 5.0, MSDOS.SYS)
33658
33659                                     ; 27/02/2024 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMDOS.COM)
33660                                     ; DOSCODE:9A89h (PCDOS 7.1, IBMDOS.COM)
33661                                 FIGREC:
33662 00005997 51                       PUSH    CX
33663                                   ; 27/02/2024
33664                                   ;;;
33665 00005998 50                       push    ax
33666 00005999 31C9                     xor     cx,cx
33667                                   ;;;
33668                                   ;mov    cl,[es:bp+5]
33669 0000599B 268A4E05                 MOV     CL,[ES:BP+DPB.CLUSTER_SHIFT]
33670                                   ;DEC    DX
33671                                   ;DEC    DX
33672                                   ; 27/02/2024
33673                                   ;;;
33674                                   ;mov    ax,[ss:CLUSTNUM_HW]
33675                                   ; ds = ss ; Retro DOS v5.0
33676 0000599F A1[E80A]                 mov     ax,[CLUSTNUM_HW]
33677 000059A2 83EA02                   sub     dx,2
33678 000059A5 83D800                   sbb     ax,0
33679 000059A8 E307                     jcxz    noshift
33680                                   ;;;
33681
33682                                 ; 27/02/2024
33683                                 %if 0
33684                                     ; MSDOS 3.3
33685                                     ;SHL    DX,CL
33686
33687                                 ;hkn; SS override HIGH_SECTOR
33688                                     ; MSDOS 6.0
33689                                     MOV     word [SS:HIGH_SECTOR],0             ;F.C. >32mb
33690                                     ; 24/09/2023
33691                                     xor     ch,ch                       ;F.C. >32mb
33692                                     OR      CL,CL                       ;F.C. >32mb
33693                                     JZ      short noshift               ;F.C. >32mb
33694                                     ; 27/02/2024
33695                                     ;XOR    CH,CH                       ;F.C. >32mb
33696                                 rotleft:                                ;F.C. >32mb
33697                                     CLC                                 ;F.C. >32mb
33698                                     RCL     DX,1                        ;F.C. >32mb
33699                                     ; 10/06/2019
33700                                     RCL     word [ss:HIGH_SECTOR],1             ;F.C. >32mb
33701                                     LOOP    rotleft                     ;F.C. >32mb
33702                                 %else
33703                                     ; 27/02/2024 - Retro DOS v5.0
33704                                     ; (PCDOS 7.1 IBMDOS.COM)
33705                                 rotleft:
33706 000059AA F8                       clc
33707 000059AB D1D2                     rcl     dx,1
33708 000059AD D1D0                     rcl     ax,1
33709 000059AF E2F9                     loop    rotleft
33710                                 %endif
33711
33712                                 noshift:
33713                                     ; MSDOS 3.3 & MSDOS 6.0
```

```
000059B1 08DA                       OR      DL,BL

                                    ; 27/02/2024 - Retro DOS v5.0
                                    ;;;
                                    ;cmp    word [es:bp+0Fh],0
000059B3 26394E0F                   cmp     word [es:bp+DPB.FAT_SIZE],cx ; 0
000059B7 750A                       jnz     short noshift1  ; not FAT32
                                    ;add    dx,[es:bp+29h]
000059B9 26035629                   add     dx,[es:bp+DPB.FCLUS_FSECTOR]
                                    ;adc    ax,[es:bp+2Bh]
000059BD 2613462B                   adc     ax,[es:bp+DPB.FCLUS_FSECTOR+2]
000059C1 EB06                       jmp     short noshift2
                                noshift1:
                                    ;;;

                                    ;add    dx,[es:bp+0Bh]
000059C3 2603560B                   ADD     DX,[ES:BP+DPB.FIRST_SECTOR]
                                    ; MSDOS 6.0
                                    ; 10/06/2019
                                    ;ADC    word [ss:HIGH_SECTOR],0             ;F.C. >32mb
                                    ; 24/09/2023
                                    ; cx=0
                                    ;ADC    word [ss:HIGH_SECTOR],cx ; 0
                                    ; 27/02/2024 - Retro DOS v5.0
                                    ;;;
000059C7 11C8                       adc     ax,cx ; adc ax,0
                                noshift2:
                                    ;mov    [ss:HIGH_SECTOR],ax
000059C9 A3[0706]                   mov     [HIGH_SECTOR],ax
                                    ;
000059CC 58                         pop     ax
                                    ;;;

                                    ; MSDOS 3.3 & MSDOS 6.0
000059CD 59                         POP     CX
                                figrec_retn:
000059CE C3                         retn

                                ; 20/05/2019 - Retro DOS v4.0
                                ; DOSCODE:8DC2h (MSDOS 6.21, MSDOS.SYS)

                                ; 30/07/2018 - Retro DOS v3.0
                                ; IBMDOS.COM (MSDOS 3.3, 1987) - Offset

                                ;Break   <ALLOCATE -- Assign disk space>
                                ;--------------------------------------------------------------------------
                                ;
                                ; Procedure Name : ALLOCATE - Allocate Disk Space
                                ;
                                ;   ALLOCATE is called to allocate disk clusters. The new clusters are
                                ;   FAT-chained onto the end of the existing file.
                                ;
                                ;   The DPB contains the cluster # of the last free cluster allocated
                                ;   (dpb_next_free). We start at this cluster and scan towards higher
                                ;   numbered clusters, looking for the necessary free blocks.
                                ;
                                ;   Once again, fancy terminology gets in the way of correct coding. When
                                ;   using next_free, start scanning AT THAT POINT and not the one following it.
                                ;   This fixes the boundary condition bug when only free = next_free = 2.
                                ;
                                ;       If we get to the end of the disk without satisfaction:
                                ;
                                ;           if (dpb_next_free == 2) then we've scanned the whole disk.
                                ;               return (insufficient_disk_space)
                                ;           ELSE
                                ;               dpb_next_free = 2; start scan over from the beginning.
                                ;
                                ;   Note that there is no multitasking interlock. There is no race when
                                ;   examining the entrys in an in-core FAT block since there will be no
                                ;   context switch. When UNPACK context switches while waiting for a FAT read
                                ;   we are done with any in-core FAT blocks, so again there is no race. The
                                ;   only special concern is that V2 and V3 MSDOS left the last allocated
                                ;   cluster as "00"; marking it EOF only when the entire alloc request was
                                ;   satisfied. We can't allow another activation to think this cluster is
                                ;   free, so we give it a special temporary mark to show that it is, indeed,
                                ;   allocated.
                                ;
                                ;   Note that when we run out of space this algorithem will scan from
                                ;   dpb_next_free to the end, then scan from cluster 2 through the end,
                                ;   redundantly scanning the later part of the disk. This only happens when
                                ;   we run out of space, so sue me.
                                ;
                                ;---+----+----+----+----+----+----+----+----+----+----+----+----+----+----;
                                ;           C  A  V  E  A  T     P  A  T  T  E  R  S  O  N               ;
                                ;                                                                        ;
                                ;   The use of FATBYT and RESTFATBYT is somewhat mysterious. Here is the
                                ;   explanation:
                                ;
                                ;   In the NUL file case (sf_firclus currently 0) ALLOCATE is called with
                                ;   entry BX = 0. What needs to be done in this case is to stuff the cluster
                                ;   number of the first cluster allocated in sf_firclus when the ALLOCATE is
                                ;   complete. THIS VALUE IS SAVED TEMPORARILY IN CLUSTER 0, HENCE THE CURRENT
                                ;   VALUE IN CLUSTER 0 MUST BE SAVED AND RESTORED. This is a side effect of
                                ;   the fact that PACK and UNPACK don't treat requests for clusters 0 and 1 as
                                ;   errors. This "stuff" is done by the call to PACK which is right before
                                ;   the
                                ;           LOOP   findfre          ; alloc more if needed
                                ;   instruction when the first cluster is allocated to the nul file. The
                                ;   value is recalled from cluster 0 and stored at sf_firclus at ads4:
                                ;
                                ;   This method is obviously useless (because it is non-reentrant) for
                                ;   multitasking, and will have to be changed. Storing the required value on
                                ;   the stack is recommended. Setting sf_firclus at the PACK of cluster 0
                                ;   (instead of actually doing the PACK) is BAD because it doesn't handle
                                ;   problems with INT 24 well.
                                ;
                                ;           C  A  V  E  A  T     P  A  T  T  E  R  S  O  N               ;
                                ;---+----+----+----+----+----+----+----+----+----+----+----+----+----+----;
                                ;                                                                        ;
                                ;       ENTRY   BX = Last cluster of file (0 if null file)
                                ;               CX = No. of clusters to allocate
                                ;               ES:BP = Base of drive parameters
                                ;               [THISSFT] = Points to SFT
                                ;
                                ;       EXIT    'C' set if insufficient space
                                ;                 [FAILERR] can be tested to see the reason for failure
                                ;                 CX = max. no. of clusters that could be added to file
                                ;               'C' clear if space allocated
                                ;                 BX = First cluster allocated
                                ;                 FAT is fully updated
                                ;                 sf_FIRCLUS field of SFT set if file was null
                                ;
                                ;       USES    ALL but SI, BP
```

```
                                ;callmagic  proc near
                                ;       push    ds                              ;push segment of routine
                                ;       push    Offset MagicPatch               ;push offset for routine
                                ;       retf                                    ;simulate jmp far
                                ;                                               ;far return address is on
                                ;                                               ;stack, so far return from
                                ;                                               ;call will return this routine
                                ;callmagic  endp

                                ; 27/02/2024 - Retro DOS v5.0
                                ;PCDOS 7.1 IBMDOS.COM - DOSCODE:9AC5h

                                ; (MSDOS 6.22 MSDOS.SYS - DOSCODE:8DC2h)
                                ; (Windows ME IO.SYS - BIOSCODE:0C078h)

                                ; 25/09/2023
                                %if 1       ; 27/02/2024
                                callmagic:
000059CF 1E                         push    ds
000059D0 36FF36[0906]               push    word [ss:OffsetMagicPatch]
000059D5 CB                         retf
                                %endif

                                ; 10/02/2024 - Retro DOS v5.0
                                %if 1
                                ALLOCATE_X:
000059D6 A3[F00A]                   mov     [CCOUNT_HW],ax
                                    ;jmp    short ALLOCATE
                                %endif

                                    ; 27/02/2024 - Retro DOS v5.0
                                    ; PCDOS 7.1 IBMDOS.COM - DOSCODE:9ACFh

                                    ; (MSDOS 6.22 MSDOS.SYS - DOSCODE:8DC9h)
                                    ; (Windows ME IO.SYS - BIOSCODE:0C07Fh)

                                ALLOCATE:
                                    ; 10/09/2018
                                ;BEGIN MAGICDRV MODIFICATIONS
                                ;
                                ;7/5/92 scottq
                                ;
                                ;This is the disk compression patch location which allows
                                ;the disk compression software to fail allocations if the
                                ;FAT would allows allocation, but the free space for compressed
                                ;data would not.
                                ;
                                ;;; call    far ptr MAGICPATCH
                                ;;; We cannot do a far call since we cannot have fix-ups[romdos,hidos],
                                ;;; but we do know the segment and offset of the routine
                                ;;; so simulate a far call to dosdata:magicpatch
                                ;;; note dosassume above, so DS -> dosdata

                                    ; MSDOS 6.0
                                       ;clc                              ;clear carry so we fall through
                                    ;                               ;if no patch is present
                                    ;push  cs                       ;push segment for far return
                                        ;call      callmagic               ;this is a near call
                                        ;jnc       short Regular_Allocate_Path
                                    ;jmp   Disk_Full_Return

                                ; 27/02/2024 - Retro DOS v5.0
                                ; DOSCODE:9ACFh (PCDOS 7.1, IBMDOS.COM)

                                ; 25/09/2023
                                %if 1       ; 27/02/2024 - Retro DOS v5.0
000059D9 F8                         clc             ; COUNT_HW:CX = Number of clusters to allocate
000059DA 0E                         push    cs
000059DB E8F1FF                     call    callmagic
000059DE 7303                       jnc     short Regular_Allocate_Path
000059E0 E9CA01                     jmp     Disk_Full_Return
                                Regular_Allocate_Path:
                                %endif

                                    ; 20/05/2019 - Retro DOS v4.0
                                ;END MAGICDRV MODIFICATIONS

                                    ; 25/11/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
                                    ; DOSCODE:8D87h (MSDOS 5.0, MSDOS.SYS)

                                    ; 27/02/2024 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMDOS.COM)
                                    ; DOSCODE:9AD6h (PCDOS 7.1, IBMDOS.COM)

000059E3 53                             PUSH    BX                      ; save (bx)
000059E4 31DB                           XOR     BX,BX

                                    ; 27/02/2024
                                    ;;;
000059E6 FF36[E80A]                 push    word [CLUSTNUM_HW]
000059EA 891E[E80A]                 mov     [CLUSTNUM_HW],bx ; cluster 0
                                    ;;;

000059EE E8F008                     call    UNPACK
000059F1 893E[9605]                     MOV     [FATBYT],DI             ; save correct cluster 0 value

                                    ; 27/02/2024
                                    ;;;             ; CCONTENT_HW:DI = content of FAT (for cluster 0)
000059F5 8B1E[EC0A]                 mov     bx,[CCONTENT_HW]
000059F9 891E[E40A]                 mov     [FATBYT_HW],bx
000059FD 8F06[E80A]                 pop     word [CLUSTNUM_HW]
                                    ;;;

00005A01 5B                             POP     BX
00005A02 72CA                           jc  short figrec_retn     ; abort if error [INTERR?]

                                ; 27/02/2024
                                %if 0
                                        PUSH    CX
                                        PUSH    BX

                                        MOV     DX,BX
                                    ;;mov   bx,[es:bp+1Ch]  ; MSDOS 3.3
                                    ;mov    bx,[es:bp+1Dh] ; MSDOS 6.0
                                        mov     bx,[ES:BP+DPB.NEXT_FREE]
                                        cmp     bx,2
                                        ja  short FINDFRE

                                ;   couldn't find enough free space beyond dpb_next_free, or dpb_next_free is
                                ;   <2 or >dpb_max_clus. Reset it and restart the scan

                                ads1:
                                    ;;mov   word [es:bp+1Ch],2 ; MSDOS 3.3
                                    ;mov    word [es:bp+1Dh],2 ; MSDOS 6.0
                                        mov     word [ES:BP+DPB.NEXT_FREE],2
```

```
33962                                           mov     bx,1                    ; Counter next instruction so first
33963                                                                           ;       cluster examined is 2
33964                                   %else
33965                                       ; 27/02/2024 - Retro DOS v5.0
33966                                       ; (PCDOS 7.1 IBMDOS.COM)
33967                                       ;;;
33968 00005A04 FF36[F00A]                   push    word [CCOUNT_HW]
33969 00005A08 51                           push    cx
33970 00005A09 FF36[E80A]                   push    word [CLUSTNUM_HW]
33971 00005A0D 53                           push    bx
33972
33973 00005A0E 89DA                         mov     dx,bx
33974
33975 00005A10 31DB                         xor     bx,bx ; 0
33976                                       ;cmp    [es:bp+0Fh],bx ; 0
33977 00005A12 26395E0F                     cmp     [es:bp+DPB.FAT_SIZE],bx ; 0
33978 00005A16 871E[E80A]                   xchg    bx,[CLUSTNUM_HW]
33979 00005A1A 891E[EA0A]                   mov     [CLUSDATA_HW],bx
33980                                       ;mov    bx,[es:bp+1Dh]
33981 00005A1E 268B5E1D                     mov     bx,[es:bp+DPB.NEXT_FREE]
33982 00005A22 7510                         jnz     short ads1              ; not FAT32
33983
33984                                       ;mov    bx,[es:bp+3Bh]
33985 00005A24 268B5E3B                     mov     bx,[es:bp+DPB.FAT32_NXTFREE+2]
33986 00005A28 891E[E80A]                   mov     [CLUSTNUM_HW],bx
33987                                       ;cmp    bx,0
33988 00005A2C 09DB                         or      bx,bx
33989                                       ;mov    bx,[es:bp+39h]
33990 00005A2E 268B5E39                     mov     bx,[es:bp+DPB.FAT32_NXTFREE]
33991 00005A32 7526                         jnz     short FINDFRE
33992                                   ads1:
33993 00005A34 83FB02                       cmp     bx,2
33994 00005A37 7721                         ja      short FINDFRE
33995                                   ads2:
33996 00005A39 31DB                         xor     bx,bx ; 0
33997                                       ;cmp    [es:bp+0Fh],bx
33998 00005A3B 26395E0F                     cmp     [es:bp+DPB.FAT_SIZE],bx ; 0
33999 00005A3F 750A                         jnz     short ads3              ; not FAT32
34000
34001                                       ;mov    word [es:bp+39h],2
34002 00005A41 26C746390200                 mov     word [es:bp+DPB.FAT32_NXTFREE],2
34003                                       ;mov    [es:bp+3Bh],bx
34004 00005A47 26895E3B                     mov     [es:bp+DPB.FAT32_NXTFREE+2],bx ; 0
34005                                   ads3:
34006 00005A4B 891E[E80A]                   mov     [CLUSTNUM_HW],bx ; 0
34007                                       ;mov    word [es:bp+1Dh],2
34008 00005A4F 26C7461D0200                 mov     word [es:bp+DPB.NEXT_FREE],2
34009 00005A55 43                           inc     bx ; 1
34010                                       ;or     [es:bp+18h],bl
34011 00005A56 26085E18                     or      [es:bp+DPB.FIRST_ACCESS],bl ; 1
34012                                       ;;;
34013                                   %endif
34014
34015                                   ;   Scanning both forwards and backwards for a free cluster
34016                                   ;
34017                                   ;       (BX) = forwards scan pointer
34018                                   ;       (CX) = clusters remaining to be allocated
34019                                   ;       (DX) = current last cluster in file
34020                                   ;       (TOS) = last cluster of file
34021
34022                                   FINDFRE:
34023                                   %if 0
34024                                           INC     BX
34025                                       ;cmp    bx,[es:bp+0Dh]
34026                                           CMP BX,[ES:BP+DPB.MAX_CLUSTER]
34027                                       ja      short ads7      ; at end of disk
34028                                           call        UNPACK          ; check out this cluster
34029                                           jc  short ads4      ; FAT error             [INTERR?]
34030                                           jnz short FINDFRE   ; not free, keep on truckin
34031
34032                                   ;   Have found a free cluster. Chain it to the file
34033                                   ;
34034                                   ;       (BX) = found free cluster #
34035                                   ;       (DX) = current last cluster in file
34036
34037                                       ;;mov   [es:bp+1Ch],bx
34038                                       ;mov    [es:bp+1Dh],bx ; MSDOS 6.0
34039                                           mov [ES:BP+DPB.NEXT_FREE],bx ; next time start search here
34040                                           xchg    ax,dx           ; save (dx) in ax
34041                                           mov     dx,1            ; mark this free guy as "1"
34042                                       call    PACK            ; set special "temporary" mark
34043                                           jc  short ads4      ; FAT error             [INTERR?]
34044                                       ;;cmp   word [es:bp+1Eh],-1
34045                                           ;cmp        word [es:bp+1Fh],-1 ; MSDOS 6.0
34046                                       CMP     word [ES:BP+DPB.FREE_CNT],-1 ; Free count valid?
34047                                           JZ  short NO_ALLOC ; No
34048                                       ;;dec   word [es:bp+1Eh]
34049                                           ;dec        word [es:bp+1Fh] ; MSDOS 6.0
34050                                           DEC     word [ES:BP+DPB.FREE_CNT] ; Reduce free count by 1
34051                                   NO_ALLOC:
34052                                           xchg    ax,dx           ; (dx) = current last cluster in file
34053                                           XCHG    BX,DX
34054                                           MOV     AX,DX
34055                                       call    PACK            ; link free cluster onto file
34056                                                                   ;  CAVEAT.. On Nul file, first pass stuffs
34057                                                                   ;    cluster 0 with FIRCLUS value.
34058                                           jc  short ads4      ; FAT error [INTERR?]
34059                                           xchg    BX,AX           ; (BX) = last one we looked at
34060                                           mov     dx,bx           ; (dx) = current end of file
34061                                           LOOP    FINDFRE         ; alloc more if needed
34062                                   %else
34063                                       ; 27/02/2024 - Retro DOS v5.0
34064                                       ; (PCDOS 7.1 IBMDOS.COM)
34065                                       ;;;
34066                                       ;add    bx,1
34067                                       ;adc    word [CLUSTNUM_HW],0
34068 00005A5A 43                           inc     bx
34069 00005A5B 7504                         jnz     short FINDFRE2
34070 00005A5D FF06[E80A]                   inc     word [CLUSTNUM_HW]
34071                                   FINDFRE2:
34072                                       ;cmp    word [es:bp+0Fh],0
34073 00005A61 26837E0F00                   cmp     word [es:bp+DPB.FAT_SIZE],0
34074 00005A66 7512                         jnz     short ads4 ; not FAT32
34075                                       ; FAT32
34076 00005A68 53                           push    bx
34077 00005A69 8B1E[E80A]                   mov     bx,[CLUSTNUM_HW]
34078                                       ;cmp    bx,[es:bp+2Fh]
34079 00005A6D 263B5E2F                     cmp     bx,[es:bp+DPB.LAST_CLUSTER+2]
34080 00005A71 5B                           pop     bx
34081 00005A72 750A                         jne     short ads5
34082                                       ;cmp    bx,[es:bp+2Dh]
34083 00005A74 263B5E2D                     cmp     bx,[es:bp+DPB.LAST_CLUSTER]
34084 00005A78 EB04                         jmp     short ads5
34085                                   ads4:
```

```
                                    ;cmp    bx,[es:bp+0Dh]
00005A7A 263B5E0D                   cmp     bx,[es:bp+DPB.MAX_CLUSTER]
                                ads5:
00005A7E 7603                       jbe     short ads6 ; jna short ads6
00005A80 E9ED00                     jmp     ads15
                                ads6:
                                    ; 27/02/2024 - Retro DOS v5.0
                                    ;push   cx
                                    ;push   word [CCOUNT_HW]
                                    ;push   bx
                                    ;push   word [CLUSTNUM_HW]
                                    ;push   dx
                                    ;push   word [CLUSDATA_HW]

00005A83 E85B08                     call    UNPACK

                                    ; 27/02/2024 - Retro DOS v5.0
                                    ;pop    word [CLUSDATA_HW]
                                    ;pop    dx
                                    ;pop    word [CLUSTNUM_HW]
                                    ;pop    bx
                                    ;pop    word [CCOUNT_HW]
                                    ;pop    cx

00005A86 7303                       jnc     short ads7

00005A88 E9A200                     jmp     ads13
                                ads7:
00005A8B 75CD                       jnz     short FINDFRE

                                    ;mov    [es:bp+1Dh],bx
00005A8D 26895E1D                   mov     [es:bp+DPB.NEXT_FREE],bx

                                    ;cmp    word [es:bp+0Fh],0
00005A91 26837E0F00                 cmp     word [es:bp+DPB.FAT_SIZE],0
00005A96 750E                       jnz     short ads8 ; not FAT32
                                    ; FAT32
00005A98 53                         push    bx
00005A99 8B1E[E80A]                 mov     bx,[CLUSTNUM_HW]
                                    ;mov    [es:bp+3Bh],bx
00005A9D 26895E3B                   mov     [es:bp+DPB.FAT32_NXTFREE+2],bx
00005AA1 5B                         pop     bx
                                    ;mov    [es:bp+39h],bx
00005AA2 26895E39                   mov     [es:bp+DPB.FAT32_NXTFREE],bx
                                ads8:
                                    ;or     byte [es:bp+18h],1
00005AA6 26804E1801                 or      byte [es:bp+DPB.FIRST_ACCESS],1

                                    ; 27/02/2024 - Retro DOS v5.0
                                    ;push   cx
                                    ;push   word [CCOUNT_HW]

00005AAB 53                         push    bx
00005AAC FF36[E80A]                 push    word [CLUSTNUM_HW]

00005AB0 52                         push    dx
00005AB1 FF36[EA0A]                 push    word [CLUSDATA_HW]

00005AB5 31D2                       xor     dx,dx   ; 0
00005AB7 8916[EA0A]                 mov     [CLUSDATA_HW],dx ; 0
00005ABB 42                         inc     dx      ; 1     ; mark this free guy as "1"
00005ABC E8D108                     call    PACK            ; set special "temporary" mark

00005ABF 8F06[E80A]                 pop     word [CLUSTNUM_HW]
00005AC3 5B                         pop     bx
00005AC4 8F06[EA0A]                 pop     word [CLUSDATA_HW]
00005AC8 5A                         pop     dx

                                    ; 27/02/2024 - Retro DOS v5.0
                                    ;pop    word [CCOUNT_HW]
                                    ;pop    cx

00005AC9 7262                       jc      short ads13

00005ACB 50                         push    ax
00005ACC 31C0                       xor     ax,ax ; 0
                                    ;cmp    [es:bp+0Fh],ax ; 0
00005ACE 2639460F                   cmp     [es:bp+DPB.FAT_SIZE],ax ; 0
00005AD2 7517                       jnz     short ads10 ; not FAT32
                                    ; FAT32
00005AD4 48                         dec     ax ; 0FFFFh ; -1
                                    ;cmp    [es:bp+21h],ax
00005AD5 26394621                   cmp     [es:bp+DPB.FREE_CNT+2],ax ; 0FFFFh
00005AD9 7506                       jnz     short ads9
                                    ;cmp    [es:bp+1Fh],ax
00005ADB 2639461F                   cmp     [es:bp+DPB.FREE_CNT],ax ; 0FFFFh
                                ;ads9:
00005ADF 7415                       jz      short NO_ALLOC ; Free count not valid
                                ads9:
                                    ; Reduce free count by 1
                                    ;add    [es:bp+1Fh],ax
00005AE1 2601461F                   add     [es:bp+DPB.FREE_CNT],ax ; -1
                                    ;adc    [es:bp+21h],ax
00005AE5 26114621                   adc     [es:bp+DPB.FREE_CNT+2],ax ; -1
00005AE9 EB0B                       jmp     short NO_ALLOC
                                ads10:
00005AEB 48                         dec     ax ; 0FFFFh ; -1
                                    ;cmp    [es:bp+1Fh],ax
00005AEC 2639461F                   cmp     [es:bp+DPB.FREE_CNT],ax ; 0FFFFh
00005AF0 7404                       jz      short NO_ALLOC ; Free count not valid
                                    ;dec    word [es:bp+1Fh]
00005AF2 26FF4E1F                   dec     word [es:bp+DPB.FREE_CNT] ; Reduce free count by 1
                                NO_ALLOC:
00005AF6 58                         pop     ax

                                    ; 27/02/2024 - Retro DOS v5.0
                                    ;push   cx
                                    ;push   word [CCOUNT_HW]

00005AF7 52                         push    dx
00005AF8 FF36[EA0A]                 push    word [CLUSDATA_HW]

00005AFC E89108                     call    PACK

00005AFF 8F06[E80A]                 pop     word [CLUSTNUM_HW]
00005B03 5B                         pop     bx

                                    ; 27/02/2024 - Retro DOS v5.0
                                    ;pop    word [CCOUNT_HW]
                                    ;pop    cx

00005B04 7227                       jc      short ads13

00005B06 8B16[E80A]                 mov     dx,[CLUSTNUM_HW]
```

```
34210 00005B0A 8916[EA0A]                   mov     [CLUSDATA_HW],dx
34211 00005B0E 89DA                         mov     dx,bx
34212
34213 00005B10 83E901                       sub     cx,1
34214 00005B13 831E[F00A]00                 sbb     word [CCOUNT_HW],0
34215
34216 00005B18 3B0E[F00A]                   cmp     cx,[CCOUNT_HW]
34217 00005B1C 7502                         jne     short ads11
34218 00005B1E E303                         jcxz    ads12
34219                                   ads11:
34220 00005B20 E937FF                       jmp     FINDFRE
34221                                       ;;;
34222                                   %endif
34223
34224
34225                                   ;   We've successfully extended the file. Clean up and exit
34226                                   ;
34227                                   ;       (BX) = last cluster in file
34228
34229                                   ads12:              ; 27/02/2024
34230 00005B23 BAFFFF                           MOV     DX,0FFFFH
34231                                       ;;;
34232 00005B26 8916[EA0A]                   mov     [CLUSDATA_HW],dx
34233                                       ;;;
34234 00005B2A E86308                       call    PACK            ; mark last cluster EOF
34235
34236                                   ;   Note that FAT errors jump here to clean the stack and exit. This saves us
34237                                   ;   2 whole bytes. Hope its worth it...
34238                                   ;
34239                                   ;       'C' set if error
34240                                   ;       calling (BX) and (CX) pushed on stack
34241
34242                                   ; 27/02/2024
34243                                   %if 0
34244
34245                                   ads4:
34246                                       POP     BX
34247                                           POP     CX              ; Don't need this stuff since we're successful
34248                                           jc  short figrec_retn
34249                                           call       UNPACK           ; Get first cluster allocated for return
34250                                                                     ; CAVEAT... In nul file case, UNPACKs cluster 0.
34251                                           jc  short figrec_retn
34252                                       call    RESTFATBYT      ; Restore correct cluster 0 value
34253                                           jc  short figrec_retn
34254                                           XCHG    BX,DI           ; (DI) = last cluster in file upon our entry
34255                                           OR      DI,DI           ; clear 'C'
34256                                       jnz     short figrec_retn ; we were extending an existing file
34257                                   %else
34258                                       ; 27/02/2024 - Retro DOS v5.0
34259                                       ; (PCDOS 7.1 IBMDOS.COM)
34260                                       ;;;
34261                                   ads13:
34262 00005B2D 5B                           pop     bx
34263 00005B2E 8F06[E80A]                   pop     word [CLUSTNUM_HW]
34264 00005B32 59                           pop     cx
34265 00005B33 8F06[F00A]                   pop     word [CCOUNT_HW]
34266 00005B37 7301                         jnc     short ads14
34267                                   ads_ret:
34268 00005B39 C3                           retn
34269                                   ads14:
34270 00005B3A E8A407                       call    UNPACK          ; Get first cluster allocated for return
34271 00005B3D 72FA                         jc      short ads_ret
34272 00005B3F E87200                       call     RESTFATBYT      ; Restore correct cluster 0 value
34273 00005B42 72F5                         jc      short ads_ret
34274 00005B44 50                           push    ax
34275 00005B45 A1[EC0A]                     mov     ax,[CCONTENT_HW]
34276 00005B48 8706[E80A]                   xchg    ax,[CLUSTNUM_HW]
34277 00005B4C 87DF                         xchg    bx,di           ; CLUSTNUM_HW:DI = last cluster in file upon our entry
34278 00005B4E 09F8                         or      ax,di
34279 00005B50 58                           pop     ax
34280 00005B51 75E6                         jnz     short ads_ret  ; we were extending an existing file
34281                                       ;;;
34282                                   %endif
34283
34284                                   ;   We were doing the first allocation for a new file. Update the SFT cluster
34285                                   ;   info
34286                                   dofastk:
34287                                   ; 27/02/2024
34288                                   %if 0
34289                                       ; 20/05/2019
34290                                       ; MSDOS 6.0
34291                                       ;push   dx ; * MSDOS 6.0
34292                                       ;;mov   dl,[es:bp+0]
34293                                       ;;MOV   DL,[ES:BP+DPB.DRIVE]  ; get drive #
34294                                       ;mov    dl,[es:bp]
34295
34296                                       ; 25/11/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
34297                                       ; DOSCODE:8DF9h (MSDOS 5.0, MSDOS.SYS)
34298
34299                                       ; 16/12/2022
34300                                       ;push   dx ; *
34301                                       ;mov    dl,[ES:BP+DPB.DRIVE]
34302                                       ; 15/12/2022
34303                                       ;mov    dl,[es:bp]
34304
34305                                       ; MSDOS 3.3 & MSDOS 6.0
34306                                       PUSH    ES
34307                                       LES      DI,[THISSFT]
34308                                       ;mov    [es:di+0Bh],bx
34309                                       MOV      [ES:DI+SF_ENTRY.sf_firclus],BX
34310                                       ;;mov   [es:di+1Bh],bx ; MSDOS 3.3
34311                                       ;mov    [es:di+35h],bx ; MSDOS 6.0
34312                                       MOV      [ES:DI+SF_ENTRY.sf_lstclus],BX
34313                                       POP     ES
34314                                       ;retn
34315
34316                                       ;pop    dx ; * MSDOS 6.0
34317
34318                                       ; 16/12/2022
34319                                       ; 25/11/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
34320                                       ;pop    dx ; *
34321                                   %else
34322                                       ; 27/02/2024 - Retro DOS v5.0
34323                                       ; (PCDOS 7.1 IBMDOS.COM)
34324                                       ;;;
34325 00005B53 52                           push    dx
34326 00005B54 06                           push    es
34327 00005B55 C43E[9E05]                   les     di,[THISSFT]
34328 00005B59 8B16[E80A]                   mov     dx,[CLUSTNUM_HW]
34329 00005B5D 26895D2B                     mov     [es:di+2Bh],bx ; [es:di+SF_ENTRYT.sf_chain]
34330                                                                ; 32 bit first cluster, lw
34331 00005B61 2689552D                     mov     [es:di+2Dh],dx ; [es:di+SF_ENTRYT.sf_chain+2]
34332                                                                ; 32 bit first cluster, hw
34333 00005B65 26895D35                     mov     [es:di+35h],bx ; [es:di+SF_ENTRYT.sf_lstclus]
```

```
                                                        ; 32 bit last cluster, lw
00005B69 26895537                   mov     [es:di+37h],dx ; [es:di+SF_ENTRYT.sf_lstclus+2]
                                                        ; 32 bit last cluster, hw
00005B6D 07                         pop     es
00005B6E 5A                         pop     dx
                                    ;;;
                                %endif

00005B6F C3                         retn

                                ;** we're at the end of the disk, and not satisfied. See if we've scanned ALL
                                ;   of the disk...

                                ; 27/02/2024
                                %if 0
                                ads7:
                                    ;cmp    word [es:bp+1Dh],2
                                    cmp     word [ES:BP+DPB.NEXT_FREE],2
                                    jnz     short ads1      ; start scan from front of disk
                                %else
                                    ; 27/02/2024 - Retro DOS v5.0 (PCDOS 7.1 IBMDOS.COM)
                                    ;;;
                                ads15:
                                    ;cmp    word [es:bp+0Fh],0
00005B70 26837E0F00                 cmp     word [es:bp+DPB.FAT_SIZE],0
00005B75 7407                       jz      short ads16 ; FAT32
                                    ; FAT
                                    ;cmp    word [es:bp+1Dh],2
00005B77 26837E1D02                 cmp     word [es:bp+DPB.NEXT_FREE],2
00005B7C EB0C                       jmp     short ads17
                                ads16:
                                    ;cmp    word [es:bp+3Bh],0
00005B7E 26837E3B00                 cmp     word [es:bp+DPB.FAT32_NXTFREE+2],0
                                    ;jnz    short ads17
00005B83 7507                       jnz     short ads19 ; 27/02/2024
                                    ;cmp    word [es:bp+39h],2
00005B85 26837E3902                 cmp     word [es:bp+DPB.FAT32_NXTFREE],2
                                ads17:
00005B8A 7403                       jz      short ads18
                                ads19:
00005B8C E9AAFE                     jmp     ads2            ; start scan from front of disk
                                    ;;;
                                %endif

                                ;   Sorry, we've gone over the whole disk, with insufficient luck. Lets give
                                ;   the space back to the free list and tell the caller how much he could have
                                ;   had.  We have to make sure we remove the "special mark" we put on the last
                                ;   cluster we were able to allocate, so it doesn't become orphaned.
                                ;
                                ;       (CX) = clusters remaining to be allocated
                                ;       (TOS) = last cluster of file (before call to ALLOCATE)
                                ;       (TOS+1) = # of clusters wanted to allocate

                                ads18:              ; 27/02/2024
00005B8F 5B                             POP     BX              ; (BX) = last cluster of file
00005B90 BAFFFF                         MOV     DX,0FFFFH

                                    ; 27/02/2024 - Retro DOS v5.0 (PCDOS 7.1 IBMDOS.COM)
                                    ;;;
00005B93 8916[EA0A]                 mov     [CLUSDATA_HW],dx
00005B97 8F06[E80A]                 pop     word [CLUSTNUM_HW]
                                    ;;;

00005B9B E83800                     call    RELBLKS         ; give back any clusters just alloced
00005B9E 58                             POP     AX              ; No. of clusters requested
                                                                ; Don't "retc". We are setting Carry anyway,
                                                                ;   Alloc failed, so proceed with return CX
                                                                ;   setup.
                                    ;;;     ; 27/02/2024
00005B9F 8F06[F20A]                 pop     word [CLUSTERS_HW]
                                    ;;;

00005BA3 29C8                           SUB     AX,CX           ; AX=No. of clusters allocated

                                    ;;;     ; 27/02/2024
00005BA5 831E[F20A]00               sbb     word [CLUSTERS_HW],0
                                    ;;;

00005BAA E80700                     call    RESTFATBYT      ; Don't "retc". We are setting Carry anyway,
                                                                ;   Alloc failed.
                                Disk_Full_Return:               ;label added for magic patch 8-6-92 scottq
                                        ; MSDOS 6.0
00005BAD C606[0B06]01               MOV     byte [DISK_FULL],1 ;MS. indicating disk full
00005BB2 F9                             STC
00005BB3 C3                             retn

                                ;--------------------------------------------------------------------------
                                ;
                                ; Procedure Name : RESTFATBYT
                                ;
                                ; SEE ALLOCATE CAVEAT
                                ;       Carry set if error (currently user FAILed to I 24)
                                ;--------------------------------------------------------------------------
                                    ; 27/02/2024 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMDOS.COM)

                                RESTFATBYT:
00005BB4 53                             PUSH    BX
00005BB5 52                             PUSH    DX
00005BB6 57                             PUSH    DI

00005BB7 31DB                           XOR     BX,BX

                                    ; 27/02/2024 - Retro DOS v5.0
                                    ; (PCDOS 7.1 IBMDOS.COM)
                                    ;;;
                                    ;push   word [CLUSTNUM_HW]
                                    ;push   word [CLUSDATA_HW]
                                    ;push   word [CCONTENT_HW]  ; (*) (not necessary ?)
00005BB9 891E[E80A]                 mov     [CLUSTNUM_HW],bx ; 0
00005BBD 8B16[E40A]                 mov     dx,[FATBYT_HW]
00005BC1 8916[EA0A]                 mov     [CLUSDATA_HW],dx
                                    ;;;

00005BC5 8B16[9605]                     MOV     DX,[FATBYT]

00005BC9 E8C407                     call    PACK

                                    ; 27/02/2024 - Retro DOS v5.0
                                    ; (PCDOS 7.1 IBMDOS.COM)
                                    ;;;
                                    ; 28/06/2024 ; (*)
                                    ;pop    word [CCONTENT_HW]
                                    ;pop    word [CLUSDATA_HW]
                                    ;pop    word [CLUSTNUM_HW]
```

```
                                    ;;;
                                        POP     DI
00005BCC 5F                             POP     DX
00005BCD 5A                             POP     BX
00005BCE 5B
                                ; 16/12/2022
                                ; 25/11/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
                                ;RELEASE_flush:
00005BCF C3                         retn

                                ;Break      <RELEASE -- DEASSIGN DISK SPACE>
                                ;-----------------------------------------------------------------------------
                                ;
                                ; Procedure Name : RELEASE
                                ;
                                ; Inputs:
                                ;       BX = Cluster in file
                                ;       ES:BP = Base of drive parameters
                                ; Function:
                                ;       Frees cluster chain starting with [BX]
                                ;       Carry set if error (currently user FAILed to I 24)
                                ; AX,BX,DX,DI all destroyed. Other registers unchanged.
                                ;
                                ;-----------------------------------------------------------------------------

                                    ; 28/02/2024 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMDOS.COM)
                                    ; PCDOS 7.1 IBMDOS.COM - DOSCODE:9D13h

                                    ; (MSDOS 6.22 MSDOS.SYS - DOSCODE:8E86h)
                                    ; (Windows ME IO.SYS - BIOSCODE:0C27Fh)

                                    ; 25/11/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
                                    ; 20/05/2019 - Retro DOS v4.0
                                RELEASE:
00005BD0 31D2                           XOR     DX,DX

                                    ; 28/02/2024 - Retro DOS v5.0
                                    ; (PCDOS 7.1 IBMDOS.COM)
                                    ;;;
00005BD2 8916[EA0A]                 mov     [CLUSDATA_HW],dx ; (dx = 0, release)
                                    ;;;

                                    ; PCDOS 7.1 IBMDOS.COM - DOSCODE:9D19h
                                    ; (MSDOS 6.22 MSDOS.SYS - DOSCODE:8E88h)
                                    ; (Windows ME IO.SYS - BIOSCODE:0C285h)

                                    ;entry  RELBLKS
                                RELBLKS:

                                ;   Enter here with DX=0FFFFH to put an end-of-file mark in the first cluster
                                ;   and free the rest in the chain.

00005BD6 E80807                     call    UNPACK
                                    ;jc     short RELEASE_flush
                                    ;jz     short RELEASE_flush
                                    ; 28/12/2024 (PCDOS 7.1 IBMDOS.COM)
00005BD9 7652                           jbe short RET12 ; jna short RET12
                                                        ; CCONTENT_HW:DI= Content of FAT
                                                        ; for given cluster (next cluster)
00005BDB 89F8                       MOV     AX,DI
00005BDD 52                             PUSH    DX

                                    ;;; 28/12/2024 - Retro DOS v5.0
                                    ;push   ax
                                    ;;;

00005BDE E8AF07                     call    PACK

                                    ;;; 28/12/2024 - Retro DOS v5.0 (PCDOS 7.1 IBMDOS.COM)
                                    ;pop    ax
00005BE1 8B16[EC0A]                 mov     dx,[CCONTENT_HW]
00005BE5 8916[E80A]                 mov     [CLUSTNUM_HW],dx ; next cluster (hw) to be released
00005BE9 89C3                       mov     bx,ax           ; next cluster (lw) to be released
                                    ;;;

00005BEB 5A                         POP     DX
                                    ;jc     short RELEASE_flush
                                    ; 28/12/2024 (PCDOS 7.1 IBMDOS.COM)
00005BEC 723F                       jc      short RET12

00005BEE 09D2                       OR      DX,DX
00005BF0 751F                           JNZ short NO_DEALLOC      ; Was putting EOF mark

                                ; 28/02/2024
                                %if 0
                                    ;;cmp   word [es:bp+1Eh],-1 ; MSDOS 3.3
                                    ;cmp    word [es:bp+1Fh],-1 ; MSDOS 6.0
                                    CMP     word [ES:BP+DPB.FREE_CNT],-1 ; Free count valid?
                                        JZ  short NO_DEALLOC      ; No
                                        INC word [ES:BP+DPB.FREE_CNT] ; Increase free count by 1
                                NO_DEALLOC:
                                        MOV     BX,AX
                                        dec     ax              ; check for "1"
                                    jz      short RELEASE_flush   ; is last cluster of incomplete chain
                                %else
                                    ; 28/02/2024 - Retro DOS v5.0
                                    ; (PCDOS 7.1 IBMDOS.COM)
                                    ;;;
00005BF2 3B16[EA0A]                 cmp     dx,[CLUSDATA_HW]      ; > 0 ? (delete, release)
00005BF6 7519                       jnz     short NO_DEALLOC      ; no, EOF (allocate, -1), (dx = 0)
                                    ;cmp    word [es:bp+1Fh],0FFFFh
00005BF8 26837E1FFF                 cmp     word [es:bp+DPB.FREE_CNT],0FFFFh ; -1
                                                                  ; Free count valid?
00005BFD 7412                       jz      short NO_DEALLOC      ; No (dx = 0)
                                relblks_ifc:
                                    ;inc    word ptr es:[bp+1Fh]
00005BFF 26FF461F                   inc     word [ES:BP+DPB.FREE_CNT] ; Increase free count by 1
00005C03 7519                       jnz     short NO_DEALLOC2

                                    ;cmp    [es:bp+0Fh],dx
00005C05 2639560F                   cmp     [es:bp+DPB.FAT_SIZE],dx
00005C09 7513                       jnz     short NO_DEALLOC2     ; not FAT32

                                    ;inc    word [es:bp+21h]
00005C0B 26FF4621                   inc     word [es:bp+DPB.FREE_CNT+2]
00005C0F EB0D                       jmp     short NO_DEALLOC2
                                NO_DEALLOC:
                                    ;cmp    [es:bp+0Fh],dx
00005C11 2639560F                   cmp     [es:bp+DPB.FAT_SIZE],dx       ; dx is -1 or 0
                                                                  ; (valid 16 bit fat size is another number)
00005C15 7507                       jnz     short NO_DEALLOC2 ; FAT (FAT16 or FAT12)
                                                                  ; dx = 0, FAT32
                                    ;cmp    word [es:bp+21h],0FFFFh
```

```
34582 00005C17 26837E21FF               cmp     word [es:bp+DPB.FREE_CNT+2],0FFFFh
34583 00005C1C 75E1                     jnz     short relblks_ifc
34584                                  NO_DEALLOC2:
34585 00005C1E 833E[E80A]00             cmp     word [CLUSTNUM_HW],0
34586 00005C23 7503                     jnz     short NO_DEALLOC3
34587 00005C25 48                       dec     ax                      ; check for "1"
34588 00005C26 7405                     jz      short RET12             ; is last cluster of incomplete chain
34589                                  NO_DEALLOC3:
34590                                      ;;;
34591                                  %endif
34592
34593 00005C28 E88D06                   call    IsEOF
34594 00005C2B 72A3                         JB  short RELEASE  ; Carry clear if JMP not taken
34595
34596                                      ; 16/12/2022
34597                                  ; 25/11/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
34598                                  ; 28/02/2024 (PCDOS 7.1 IBMDOS.COM)
34599                                  %if 0
34600                                  RELEASE_flush:
34601                                      ; MSDOS 6.0
34602                                      mov     al,[es:bp]
34603                                      ;MOV    AL,[ES:BP+DPB.DRIVE]
34604                                      push    si              ; FLUSHBUF may trash these and we guarantee
34605                                      push    cx              ;  them to be preserved.
34606                                      push    es
34607                                      push    bp
34608                                      call    FLUSHBUF        ; commit buffers for this drive
34609                                      pop     bp
34610                                      pop     es
34611                                      pop     cx
34612                                      pop     si
34613                                  %endif              ; 28/02/2024
34614
34615                                  RET12:
34616 00005C2D C3                       retn
34617
34618                                  ;Break     <GETEOF -- Find the end of a file>
34619                                  ;--------------------------------------------------------------------------
34620                                  ;
34621                                  ; Procedure Name : GETEOF
34622                                  ;
34623                                  ; Inputs:
34624                                  ;       ES:BP Points to DPB
34625                                  ;       BX = Cluster in a file
34626                                  ;       DS = CS
34627                                  ; Outputs:
34628                                  ;       BX = Last cluster in the file
34629                                  ;       Carry set if error (currently user FAILed to I 24)
34630                                  ; DI destroyed. No other registers affected.
34631                                  ;
34632                                  ;--------------------------------------------------------------------------
34633
34634                                      ; 28/02/2024 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMDOS.COM)
34635                                  %if 0
34636                                  GETEOF:
34637                                      call    UNPACK
34638                                          jc  short RET12
34639                                          PUSH    BX
34640                                          MOV     BX,DI
34641                                      call    IsEOF
34642                                          POP     BX
34643                                          JAE     short RET12
34644                                          MOV     BX,DI
34645                                          JMP     short GETEOF
34646                                  %else
34647                                      ; 28/02/2024 - Retro DOS v5.0
34648                                      ;;;
34649                                  GETEOF4:    ; **
34650 00005C2E 8F06[E80A]                   pop     word [CLUSTNUM_HW] ; * ; 28/02/2024
34651                                  GETEOF1:
34652 00005C32 89C7                     mov     di,ax
34653 00005C34 8916[F20A]               mov     [CLUSTERS_HW],dx ; CLUSTERS_HW:DI = Cluster Count
34654 00005C38 58                       pop     ax
34655 00005C39 5A                       pop     dx
34656 00005C3A C3                       retn
34657
34658                                      ; PCDOS 7.1 IBMDOS.COM - DOSCODE:9D7Ch
34659                                  GETEOF:
34660 00005C3B 52                       push    dx
34661 00005C3C 50                       push    ax
34662 00005C3D 31D2                     xor     dx,dx ; 0       ; cluster count = 0
34663 00005C3F 31C0                     xor     ax,ax ; 0
34664                                  GETEOF2:
34665 00005C41 E89D06                   call    UNPACK
34666 00005C44 72EC                     jc      short GETEOF1
34667                                           ; CCONTENT_HW:DI = next cluster (cluster content)
34668 00005C46 40                       inc     ax
34669 00005C47 7501                     jnz     short GETEOF3
34670 00005C49 42                       inc     dx
34671                                  GETEOF3:
34672 00005C4A FF36[E80A]               push    word [CLUSTNUM_HW] ; * ; 28/02/2024
34673 00005C4E 53                       push     bx
34674                                   ;push   word [CLUSTNUM_HW]
34675 00005C4F 8B1E[EC0A]               mov     bx,[CCONTENT_HW]
34676 00005C53 891E[E80A]               mov     [CLUSTNUM_HW],bx
34677 00005C57 89FB                     mov     bx,di
34678 00005C59 E85C06                   call    IsEOF
34679                                   ;pop    word [CLUSTNUM_HW] ; * ; 28/02/2024
34680 00005C5C 5B                       pop     bx
34681                                   ;jae    short GETEOF1   ; EOF
34682 00005C5D 73CF                     jae     short GETEOF4 ; ** ; 28/02/2024
34683                                   ;mov    bx,[CCONTENT_HW] ; not EOF
34684                                   ;mov    [CLUSTNUM_HW],bx
34685 00005C5F 5B                       pop     bx      ; * ; 28/02/2024
34686 00005C60 89FB                     mov     bx,di
34687 00005C62 EBDD                     jmp     short GETEOF2   ; get next cluster
34688                                   ;;;
34689                                  %endif
34690
34691                                  ;============================================================================
34692                                  ; FCB.ASM, MSDOS 6.0, 1991
34693                                  ;============================================================================
34694                                  ; 30/07/2018 - Retro DOS v3.0
34695                                  ; 20/05/2019 - Retro DOS v4.0
34696                                  ; 29/02/2024 - Retro DOS v5.0
34697
34698                                  ;   TITLE   FCB - FCB parse calls for MSDOS
34699                                  ;   NAME    FCB
34700
34701                                  ;** FCB.ASM - Low level routines for parsing names into FCBs and analyzing
34702                                  ;             filename characters
34703                                  ;
34704                                  ;   MakeFcb
34705                                  ;   NameTrans
```

```
                        ;   PATHCHRCMP
                        ;   GetLet
                        ;   UCase
                        ;   GetLet3
                        ;   GetCharType
                        ;   TESTKANJ
                        ;   NORMSCAN
                        ;   DELIM
                        ;
                        ;   Revision history:
                        ;
                        ;          A000  version 4.00  Jan. 1988
                        ;
                        ;   M048 - access FILE_UCASE_TAB using DS rather than SS.

                        TableLook   EQU     -1

                        SCANSEPARATOR       EQU     1
                        DRVBIT              EQU     2
                        NAMBIT              EQU     4
                        EXTBIT              EQU     8

                        ;--------------------------------------------------------------------------
                        ;
                        ; Procedure : MakeFcb
                        ;
                        ;--------------------------------------------------------------------------

                            ; 25/11/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
                            ; DOSCODE:8E77h (MSDOS 5.0, MSDOS.SYS)

                            ; 29/02/2024 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMDOS.COM)
                            ; PCDOS 7.1 IBMDOS.COM - DOSCODE:9DB0h

                            ; (MSDOS 6.22 MSDOS.SYS - DOSCODE:8ED3h)
                            ; (Windows ME IO.SYS - BIOSCODE:8D92h)

                        MAKEFCB:
                        ;hkn; SS override
                            ;MOV    BYTE [SS:SpaceFlag],0
00005C64 30D2               XOR     DL,DL           ; Flag--not ambiguous file name
                            ; 29/02/2024
00005C66 368816[4E03]       mov     [ss:SpaceFlag],dl ; 0
                            ;test   al,2
00005C6B A802               test    AL,DRVBIT       ; Use current drive field if default?
00005C6D 7503               JNZ     short DEFDRV
                            ;MOV    BYTE [ES:DI],0 ; No - use default drive
                            ; 29/02/2024
00005C6F 268815             mov     [es:di],dl ; 0
                        DEFDRV:
00005C72 47                 INC     DI
00005C73 B90800             MOV     CX,8
                            ;test   al,4
00005C76 A804               test    AL,NAMBIT       ; Use current name fields as default?
00005C78 93                 XCHG    AX,BX           ; Save bits in BX
00005C79 B020               MOV     AL," "
00005C7B 7404               JZ      short FILLB     ; If not, go fill with blanks
00005C7D 01CF               ADD     DI,CX
00005C7F 31C9               XOR     CX,CX           ; Don't fill any
                        FILLB:
00005C81 F3AA               REP     STOSB
00005C83 B103               MOV     CL,3
00005C85 F6C308             test    BL,EXTBIT       ; Use current extension as default
00005C88 7404               JZ      short FILLB2
00005C8A 01CF               ADD     DI,CX
00005C8C 31C9               XOR     CX,CX
                        FILLB2:
00005C8E F3AA               REP     STOSB
00005C90 91                 XCHG    AX,CX           ; Put zero in AX
00005C91 AB                 STOSW
00005C92 AB                 STOSW                   ; Initialize two words after to zero
00005C93 83EF10             SUB     DI,16           ; Point back at start
                            ;test   bl,1
00005C96 F6C301             test    BL,SCANSEPARATOR; Scan off separators if not zero
00005C99 7409               JZ      short SKPSPC
00005C9B E88800             CALL    SCANB           ; Peel off blanks and tabs
00005C9E E81E01             CALL    DELIM           ; Is it a one-time-only delimiter?
00005CA1 7504               JNZ     short NOSCAN
00005CA3 46                 INC     SI              ; Skip over the delimiter
                        SKPSPC:
00005CA4 E87F00             CALL    SCANB           ; Always kill preceding blanks and tabs
                        NOSCAN:
00005CA7 E8EC00             CALL    GETLET
00005CAA 761E               JBE     short NODRV     ; Quit if termination character
00005CAC 803C3A             CMP     BYTE [SI],":"  ; Check for potential drive specifier
00005CAF 7519               JNZ     short NODRV
00005CB1 46                 INC     SI              ; Skip over colon
00005CB2 2C40               SUB     AL,"@"          ; Convert drive letter to drive number (A=1)
00005CB4 760F               JBE     short BADDRV    ; Drive letter out of range

00005CB6 50                 PUSH    AX
00005CB7 E8B71E             call    GetVisDrv
00005CBA 58                 POP     AX
00005CBB 730A               JNC     short HAVDRV

                            ; 20/05/2019 - Retro DOS v4.0
                            ; MSDOS 6.0
                        ;hkn; SS override
00005CBD 36803E[1006]1A     CMP     byte [SS:DrvErr],error_not_DOS_disk ; 1Ah
                                                    ; if not FAT drive ;AN000;
00005CC3 7402               JZ      short HAVDRV    ; assume ok       ;AN000;
                        BADDRV:
00005CC5 B2FF               MOV     DL,-1
                        HAVDRV:
00005CC7 AA                 STOSB                   ; Put drive specifier in first byte
00005CC8 46                 INC     SI
00005CC9 4F                 DEC     DI              ; Counteract next two instructions
                        NODRV:
00005CCA 4E                 DEC     SI              ; Back up
00005CCB 47                 INC     DI              ; Skip drive byte

                            ;entry  NORMSCAN
                        NORMSCAN:
00005CCC B90800             MOV     CX,8
00005CCF E82200             CALL    GETWORD         ; Get 8-letter file name
00005CD2 803C2E             CMP     BYTE [SI],"."
00005CD5 7510               JNZ     short NODOT
00005CD7 46                 INC     SI              ; Skip over dot if present

                            ; 24/09/2023
                            ;mov    cx,3
00005CD8 B103               mov     cl,3    ; ch=0

                            ; MSDOS 6.0
```

```
                                ;hkn; SS override
                                    ;TEST  word [SS:DOS34_FLAG],DBCS_VOLID2 ; 100h ;AN000;
                                    ; 10/06/2019
00005CDA 36F606[1206]01             test   byte [SS:DOS34_FLAG+1],(DBCS_VOLID2>>8) ; 1
00005CE0 7402                       JZ     short VOLOK                      ;AN000;
00005CE2 A4                         MOVSB                 ; 2nd byte of DBCS    ;AN000;
                                    ; 24/09/2023
                                    ;MOV   CX,2                             ;AN000;
00005CE3 49                         dec    cx  ; cx=2
                                    ;JMP   SHORT contvol                    ;AN000;
                                VOLOK:
                                    ;MOV   CX,3           ; Get 3-letter extension
                                contvol:
00005CE4 E81300                     CALL   MUSTGETWORD
                                NODOT:
00005CE7 88D0                       MOV    AL,DL

                                    ; MSDOS 6.0
                                    ;and   word [ss:DOS34_FLAG],0FEFFh
                                    ; 18/12/2022
00005CE9 368026[1206]FE             and    byte [ss:DOS34_FLAG+1],0FEh ; (~DBCS_VOLID2)>>8
                                    ;and   word [ss:DOS34_FLAG],~DBCS_VOLID2 ; ### BUG FIX ###

00005CEF C3                         retn

                                NONAM:
00005CF0 01CF                       ADD    DI,CX
00005CF2 4E                         DEC    SI
00005CF3 C3                         retn

                                GETWORD:
00005CF4 E89F00                     CALL   GETLET
00005CF7 76F7                       JBE    short NONAM    ; Exit if invalid character
00005CF9 4E                         DEC    SI

                                ;   UGH!!! Horrible bug here that should be fixed at some point:
                                ;   If the name we are scanning is longer than CX, we keep on reading!

                                MUSTGETWORD:
00005CFA E89900                     CALL   GETLET

                                ;   If spaceFlag is set then we allow spaces in a pathname

                                ;IF NOT TABLELOOK
                                ;   JB     short FILLNAM  ; MSDOS 3.3
                                ;ENDIF
00005CFD 750C                       JNZ    short MustCheckCX

                                ;hkn; SS override
00005CFF 36F606[4E03]FF             test   BYTE [SS:SpaceFlag],0FFh
00005D05 7419                       JZ     short FILLNAM
00005D07 3C20                       CMP    AL," "
00005D09 7515                       JNZ    short FILLNAM

                                MustCheckCX:
00005D0B E3ED                       JCXZ   MUSTGETWORD
00005D0D 49                         DEC    CX
00005D0E 3C2A                       CMP    AL,"*"         ; Check for ambiguous file specifier
00005D10 7504                       JNZ    short NOSTAR
00005D12 B03F                       MOV    AL,"?"
00005D14 F3AA                       REP    STOSB
                                NOSTAR:
00005D16 AA                         STOSB
00005D17 3C3F                       CMP    AL,"?"
00005D19 75DF                       JNZ    short MUSTGETWORD
00005D1B 80CA01                     OR     DL,1           ; Flag ambiguous file name
00005D1E EBDA                       JMP    short MUSTGETWORD
                                FILLNAM:
00005D20 B020                       MOV    AL," "
00005D22 F3AA                       REP    STOSB
00005D24 4E                         DEC    SI
00005D25 C3                         retn

                                SCANB:
00005D26 AC                         LODSB
00005D27 E89D00                     CALL   SPCHK
00005D2A 74FA                       JZ     short SCANB
00005D2C 4E                         DEC    SI
                                scanb_retn:
00005D2D C3                         retn

                                ;--------------------------------------------------------------------------
                                ;
                                ; Procedure Name : NameTrans
                                ;
                                ; NameTrans is used by FindPath to scan off an element of a path. We must
                                ; allow spaces in pathnames
                                ;
                                ;   Inputs: DS:SI points to start of path element
                                ;   Outputs:       Name1 has unpacked name, uppercased
                                ;           ES = DOSGroup
                                ;           DS:SI advanced after name
                                ;   Registers modified: DI,AX,DX,CX
                                ;
                                ;--------------------------------------------------------------------------

                                    ; 25/11/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
                                    ; 20/05/2019 - Retro DOS v4.0

                                    ; 29/02/2024 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMDOS.COM)
                                    ; PCDOS 7.1 IBMDOS.COM - DOSCODE:9E7Ch

                                NameTrans:
                                ;hkn; SS override
00005D2E 36C606[4E03]01             MOV    BYTE [SS:SpaceFlag],1
00005D34 16                         push   ss
00005D35 07                         pop    es

                                ;hkn; NAME1 is in DOSDATA
00005D36 BF[4B05]                   MOV    DI,NAME1
00005D39 57                         PUSH   DI

                                ; 29/02/2024
                                %if 0
                                    MOV    AX,'  ' ; 2020h
                                    MOV    CX,5
                                    STOSB
                                    REP    STOSW          ; Fill "FCB" at NAME1 with spaces
                                    XOR    AL,AL          ; Set stuff for NORMSCAN
                                    MOV    DL,AL
                                %else
                                    ; 29/02/2024
                                    ; (PCDOS 7.1 IBMDOS.COM)
                                    ;;;
```

```
00005D3A B020                       mov     al, 20h ; ' '
00005D3C B90B00                     mov     cx, 11
00005D3F F3AA                       rep stosb               ; Fill "FCB" at NAME1 with spaces
00005D41 91                         xchg    ax, cx
00005D42 99                         cwd
                                    ;;;
                                %endif

00005D43 AA                         STOSB
00005D44 5F                         POP     DI

00005D45 E884FF                     CALL    NORMSCAN

                                ;hkn; SS override for NAME1
00005D48 36803E[4B05]E5             CMP     byte [SS:NAME1],0E5H
00005D4E 75DD                       jnz     short scanb_retn
00005D50 36C606[4B05]05             MOV     byte [SS:NAME1],5 ; Magic name translation
00005D56 C3                         retn

                                ;Break      <GETLET, DELIM -- CHECK CHARACTERS AND CONVERT>
                                ;=============================================================================

                                ; 20/05/2019 - Retro DOS v4.0
                                ; DOSCODE:8FD2h (MSDOS 6.21, MSDOS.SYS)

                                ;If TableLook

                                ;hkn; Table SEGMENT
                                ;   PUBLIC CharType
                                ;-----------------------------------------------------------------------------

                                ; Character type table for file name scanning
                                ; Table provides a mapping of characters to validity bits.
                                ; Four bits are provided for each character. Values 7Dh and above
                                ; have all bits set, so that part of the table is chopped off, and
                                ; the translation routine is responsible for screening these values.
                                ; The bit values are defined in DOSSYM.INC

                                ;       ; ^A and NUL
                                ;CharType:
                                ;    db   LOW ((NOT FFCB+FCHK) SHL 4) OR LOW (NOT FFCB+FCHK) AND 0Fh
                                ;         ; ^C and ^B
                                ;    db   LOW ((NOT FFCB+FCHK) SHL 4) OR LOW (NOT FFCB+FCHK) AND 0Fh
                                ;         ; ^E and ^D
                                ;    db   LOW ((NOT FFCB+FCHK) SHL 4) OR LOW (NOT FFCB+FCHK) AND 0Fh
                                ;         ; ^G and ^F
                                ;    db   LOW ((NOT FFCB+FCHK) SHL 4) OR LOW (NOT FFCB+FCHK) AND 0Fh
                                ;         ; TAB and BS
                                ;    db   LOW ((NOT FFCB+FCHK+FDELIM+FSPCHK) SHL 4) OR LOW (NOT FFCB+FCHK) AND 0Fh
                                ;         ; ^K and ^J
                                ;    db   LOW ((NOT FFCB+FCHK) SHL 4) OR LOW (NOT FFCB+FCHK) AND 0Fh
                                ;         ; ^M and ^L
                                ;    db   LOW ((NOT FFCB+FCHK) SHL 4) OR LOW (NOT FFCB+FCHK) AND 0Fh
                                ;         ; ^O and ^N
                                ;    db   LOW ((NOT FFCB+FCHK) SHL 4) OR LOW (NOT FFCB+FCHK) AND 0Fh
                                ;         ; ^Q and ^P
                                ;    db   LOW ((NOT FFCB+FCHK) SHL 4) OR LOW (NOT FFCB+FCHK) AND 0Fh
                                ;         ; ^S and ^R
                                ;    db   LOW ((NOT FFCB+FCHK) SHL 4) OR LOW (NOT FFCB+FCHK) AND 0Fh
                                ;         ; ^U and ^T
                                ;    db   LOW ((NOT FFCB+FCHK) SHL 4) OR LOW (NOT FFCB+FCHK) AND 0Fh
                                ;         ; ^W and ^V
                                ;    db   LOW ((NOT FFCB+FCHK) SHL 4) OR LOW (NOT FFCB+FCHK) AND 0Fh
                                ;         ; ^Y and ^X
                                ;    db   LOW ((NOT FFCB+FCHK) SHL 4) OR LOW (NOT FFCB+FCHK) AND 0Fh
                                ;         ; ESC and ^Z
                                ;    db   LOW ((NOT FFCB+FCHK) SHL 4) OR LOW (NOT FFCB+FCHK) AND 0Fh
                                ;         ; ^] and ^;       db   LOW ((NOT FFCB+FCHK) SHL 4) OR LOW (NOT FFCB+FCHK) AND 0Fh
                                ;         ; ^_ and ^^
                                ;    db   LOW ((NOT FFCB+FCHK) SHL 4) OR LOW (NOT FFCB+FCHK) AND 0Fh
                                ;         ; ! and SPACE
                                ;    db   LOW (NOT FCHK+FDELIM+FSPCHK)
                                ;         ; # and "
                                ;    db   LOW (NOT FFCB+FCHK)
                                ;         ; $ - )
                                ;    db   3 dup (0FFh)
                                ;         ; + and *
                                ;    db   LOW ((NOT FFCB+FCHK+FDELIM) SHL 4) OR 0Fh
                                ;         ; - and '
                                ;    db   NOT (FFCB+FCHK+FDELIM)
                                ;         ; / and .
                                ;    db   LOW ((NOT FFCB+FCHK) SHL 4) OR LOW (NOT FCHK) AND 0Fh
                                ;         ; 0 - 9
                                ;    db   5 dup (0FFh)
                                ;         ; ; and :
                                ;    db   LOW ((NOT FFCB+FCHK+FDELIM) SHL 4) OR LOW (NOT FFCB+FCHK+FDELIM) AND 0Fh
                                ;         ; = and <
                                ;    db   LOW ((NOT FFCB+FCHK+FDELIM) SHL 4) OR LOW (NOT FFCB+FCHK+FDELIM) AND 0Fh
                                ;         ; ? and >
                                ;    db   NOT FFCB+FCHK+FDELIM
                                ;         ; A - Z
                                ;    db   13 dup (0FFh)
                                ;         ; \ and [
                                ;    db   LOW ((NOT FFCB+FCHK) SHL 4) OR 0Fh
                                ;         ; ^ and ]
                                ;    db   LOW ((NOT FFCB+FCHK) SHL 4) OR LOW (NOT FFCB+FCHK) AND 0Fh
                                ;         ; _ - {
                                ;    db   15 dup (0FFh)
                                ;         ; } and |
                                ;    db   NOT FFCB+FCHK+FDELIM

                                ;CharType_last equ ($ - CharType) * 2    ; This is the value of the last
                                ;                                 ; character in the table

                                ;FCHK       equ 1          ; normal name char, no chks needed
                                ;FDELIM     equ 2          ; is a delimiter
                                ;FSPCHK     equ 4          ; set if character is not a space or equivalent
                                ;FFCB       equ 8          ; is valid in an FCB

                                ; DOSCODE:8FD2h (MSDOS 6.21, MSDOS.SYS)
                                ;-----------------------------------------------------------------------------
                                ; DOSCODE:8F76h (MSDOS 5.0, MSDOS.SYS)

                                CharType: ; 63 bytes
00005D57 6666666606666666               db  66h, 66h, 66h, 66h, 06h, 66h, 66h, 66h ; 0-7
00005D5F 6666666666666666           db  66h, 66h, 66h, 66h, 66h, 66h, 66h, 66h ; 8-15
00005D67 F8F6FFFFFF4FF46E           db 0F8h,0F6h,0FFh,0FFh,0FFh, 4Fh,0F4h, 6Eh ; 16-23
00005D6F FFFFFFFFF4444F4            db 0FFh,0FFh,0FFh,0FFh,0FFh, 44h, 44h,0F4h ; 24-31
00005D77 FFFFFFFFFFFFFFFF           db 0FFh,0FFh,0FFh,0FFh,0FFh,0FFh,0FFh,0FFh ; 32-39
00005D7F FFFFFFFFF6F66FF            db 0FFh,0FFh,0FFh,0FFh,0FFh, 6Fh, 66h,0FFh ; 40-47
00005D87 FFFFFFFFFFFFFFFF           db 0FFh,0FFh,0FFh,0FFh,0FFh,0FFh,0FFh,0FFh ; 48-55
00005D8F FFFFFFFFFFFF4              db 0FFh,0FFh,0FFh,0FFh,0FFh,0FFh,0F4h          ; 56-62

                                CharType_last equ ($ - CharType) * 2
```

```

                               ; Offset 12CAh of IBMDOS.COM (MSDOS 3.3), 1987
                               ;--------------------------------------------------------------------------
                               ;CharType:
                               ;       db 0F6h,0F6h,0F6h,0F6h,0F6h,0F6h,0F6h,0F6h
                               ;   db 0F6h,0F0h,0F6h,0F6h,0F6h,0F6h,0F6h,0F6h
                               ;   db 0F6h,0F6h,0F6h,0F6h,0F6h,0F6h,0F6h,0F6h
                               ;   db 0F6h,0F6h,0F6h,0F6h,0F6h,0F6h,0F6h,0F6h
                               ;   db 0F8h,0FFh,0F6h,0FFh,0FFh,0FFh,0FFh,0FFh
                               ;   db 0FFh,0FFh,0FFh,0F4h,0F4h,0FFh,0FEh,0F6h
                               ;   db 0FFh,0FFh,0FFh,0FFh,0FFh,0FFh,0FFh,0FFh
                               ;   db 0FFh,0FFh,0F4h,0F4h,0F4h,0F4h,0F4h,0FFh
                               ;   db 0FFh,0FFh,0FFh,0FFh,0FFh,0FFh,0FFh,0FFh
                               ;   db 0FFh,0FFh,0FFh,0FFh,0FFh,0FFh,0FFh,0FFh
                               ;   db 0FFh,0FFh,0FFh,0FFh,0FFh,0FFh,0FFh,0FFh
                               ;   db 0FFh,0FFh,0FFh,0F6h,0F6h,0F6h,0FFh,0FFh
                               ;   db 0FFh,0FFh,0FFh,0FFh,0FFh,0FFh,0FFh,0FFh
                               ;   db 0FFh,0FFh,0FFh,0FFh,0FFh,0FFh,0FFh,0FFh
                               ;   db 0FFh,0FFh,0FFh,0FFh,0FFh,0FFh,0FFh,0FFh
                               ;   db 0FFh,0FFh,0FFh,0FFh,0F4h,0FFh,0FFh,0FFh
                               ;   db 0FFh,0FFh,0FFh,0FFh,0FFh,0FFh,0FFh,0FFh
                               ;   db 0FFh,0FFh,0FFh,0FFh,0FFh,0FFh,0FFh,0FFh
                               ;   db 0FFh,0FFh,0FFh,0FFh,0FFh,0FFh,0FFh,0FFh
                               ;   db 0FFh,0FFh,0FFh,0FFh,0FFh,0FFh,0FFh,0FFh
                               ;   db 0FFh,0FFh,0FFh,0FFh,0FFh,0FFh,0FFh,0FFh
                               ;   db 0FFh,0FFh,0FFh,0FFh,0FFh,0FFh,0FFh,0FFh
                               ;   db 0FFh,0FFh,0FFh,0FFh,0FFh,0FFh,0FFh,0FFh
                               ;   db 0FFh,0FFh,0FFh,0FFh,0FFh,0FFh,0FFh,0FFh
                               ;   db 0FFh,0FFh,0FFh,0FFh,0FFh,0FFh,0FFh,0FFh
                               ;   db 0FFh,0FFh,0FFh,0FFh,0FFh,0FFh,0FFh,0FFh
                               ;   db 0FFh,0FFh,0FFh,0FFh,0FFh,0FFh,0FFh,0FFh
                               ;   db 0FFh,0FFh,0FFh,0FFh,0FFh,0FFh,0FFh,0FFh
                               ;   db 0FFh,0FFh,0FFh,0FFh,0FFh,0FFh,0FFh,0FFh
                               ;   db 0FFh,0FFh,0FFh,0FFh,0FFh,0FFh,0FFh,0FFh
                               ;   db 0FFh,0FFh,0FFh,0FFh,0FFh,0FFh,0FFh,0FFh
                               ;   db 0FFh,0FFh,0FFh,0FFh,0FFh,0FFh,0FFh,0FFh

                               ;hkn; Table ENDS

                               ;ENDIF

                               ; 20/05/2019 - Retro DOS v4.0
                               ; DOSCODE:9011h (MSDOS 6.21, MSDOS.SYS)

                               ; 25/11/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
                               ; DOSCODE:8FB5h (MSDOS 5.0, MSDOS.SYS)

                               ;---------------------------------------------------------------------------
                               ;
                               ; Procedure Names : GetLet, UCase, GetLet3
                               ;
                               ; These routines take a character, convert it to upper case, and check
                               ; for delimiters.  Three different entry points:
                               ;   GetLet -  DS:[SI] = character to convert
                               ;   UCase  -  AL = character to convert
                               ;   GetLet3 - AL = character
                               ;             [BX] = translation table to use
                               ;
                               ;   Exit (in all cases) : AL = upper case character
                               ;                         CY set if char is control char other than TAB
                               ;                         ZF set if char is a delimiter
                               ;   Uses : AX, flags
                               ;
                               ; NOTE: This routine exists in a fast table lookup version, and a slow
                               ; inline version.  Return with carry set is only possible in the inline
                               ; version. The table lookup version is the one in use.
                               ;
                               ;---------------------------------------------------------------------------

                               ; This entry point has character at [SI]

                                   ; IBMDOS.COM (MSDOS 3.3, 1987) - Offset 5517h
                               GETLET:
00005D96 AC                        LODSB

                               ; This entry point has character in AL

                                   ;entry  UCase
                               UCase:
                                   ; 09/08/2018
                                   ; IBMDOS.COM (MSDOS 3.3, 1987) - Offset 5518h
                               _UCase:
00005D97 53                        PUSH    BX
00005D98 BB[7E0B]                  MOV     BX,FILE_UCASE_TAB+2

                               ; Convert the character in AL to upper case

                               gl_0:
00005D9B 3C61                      CMP     AL,"a"
00005D9D 7214                      JB      short gl_2      ; Already upper case, go check type
00005D9F 3C7A                      CMP     AL,"z"
00005DA1 7702                      JA      short gl_1
00005DA3 2C20                      SUB     AL,20H          ; Convert to upper case

                               ; Map European character to upper case

                               gl_1:
00005DA5 3C80                      CMP     AL,80H
00005DA7 720A                      JB      short gl_2      ; Not EuroChar, go check type
00005DA9 2C80                      SUB     AL,80H          ; translate to upper case with this index

                                   ; M048 - Start
                                   ; Lantastic call Ucase thru int 2f without setting SS to DOSDATA.
                                   ; So we shall set up DS and to access FILE_UCASE_TAB in BX and also
                                   ; preserve it.

                                   ; 09/08/2018 - Retro DOS v3.0
                                   ; MSDOS 3.3
                                   ;;XLAT  BYTE [CS:BX]   ; ds as file_ucase_tab is in DOSDATA
                                   ;CS     XLAT

                                   ; 20/05/2019 - Retro DOS v4.0

                                   ; MSDOS 6.0
00005DAB 1E                        push    ds
                                   ;getdseg <ds>
00005DAC 2E8E1E[0700]              mov     ds,[cs:DosDSeg]
00005DB1 D7                        XLAT                    ; ds as file_ucase_tab is in DOSDATA
00005DB2 1F                        pop     ds

                                   ; M048 - End

                               ; Now check the type

```

```
                                ;If TableLook
                                gl_2:
                                    ; 20/05/2019 - Retro DOS v4.0
00005DB3 50                         PUSH    AX

                                    ; MSDOS 3.3
                                    ;mov    bx,CharType
                                    ;; 09/08/2018
                                    ;;xlat  byte [cs:bx]
                                    ;cs     xlat

                                    ; MSDOS 6.0
00005DB4 E81800                     CALL    GetCharType    ; returns type flags in AL

                                    ;test   al,1
00005DB7 A801                       TEST    AL,FCHK        ; test for normal character
00005DB9 58                         POP     AX

00005DBA 5B                         POP     BX
00005DBB C3                         RETN

                                ; This entry has character in AL and lookup table in BX

                                    ; MSDOS 6.0
                                ;   ;entry GetLet3
                                GETLET3: ; 10/08/2018
00005DBC 53                         PUSH    BX
00005DBD EBDC                       JMP     short gl_0
                                ;ELSE
                                ;
                                ;gl_2:
                                ;   POP     BX
                                ;   CMP     AL,"."
                                ;   retz
                                ;   CMP     AL,'"'
                                ;   retz
                                ;   CALL    PATHCHRCMP
                                ;   retz
                                ;   CMP     AL,"["
                                ;   retz
                                ;   CMP     AL,"]"
                                ;   retz
                                ;ENDIF

                                ;--------------------------------------------------------------------------
                                ;
                                ; DELIM - check if character is a delimiter
                                ;   Entry : AX = character to check
                                ;   Exit  : ZF set if character is not a delimiter
                                ;   Uses  : Flags
                                ;
                                ;--------------------------------------------------------------------------

                                    ;entry DELIM
                                DELIM:
                                ;IF TableLook
                                    ; 20/05/2019 - Retro DOS v4.0
00005DBF 50                         PUSH    AX

                                    ; MSDOS 3.3
                                    ;push   bx
                                    ;mov    bx,CharType
                                    ;;09/08/2018
                                    ;;xlat  byte [cs:bx]
                                    ;cs     xlat
                                    ;pop    bx

                                    ; MSDOS 6.0
00005DC0 E80C00                     CALL    GetCharType

                                    ;test   al,2
00005DC3 A802                       TEST    AL,FDELIM
00005DC5 58                         POP     AX
00005DC6 C3                         RETN
                                ;ELSE
                                ;   CMP     AL,":"
                                ;   retz
                                ;
                                ;   CMP     AL,"<"
                                ;   retz
                                ;   CMP     AL,"|"
                                ;   retz
                                ;   CMP     AL,">"
                                ;   retz
                                ;
                                ;   CMP     AL,"+"
                                ;   retz
                                ;   CMP     AL,"="
                                ;   retz
                                ;   CMP     AL,";"
                                ;   retz
                                ;   CMP     AL,","
                                ;   retz
                                ;ENDIF

                                ;--------------------------------------------------------------------------
                                ;
                                ;  SPCHK - checks to see if a character is a space or equivalent
                                ;   Entry : AL = character to check
                                ;   Exit  : ZF set if character is a space
                                ;   Uses  : flags
                                ;
                                ;--------------------------------------------------------------------------

                                    ;entry SPCHK
                                SPCHK:
                                ;IF TableLook
                                    ; 20/05/2019 - Retro DOS v4.0
00005DC7 50                         PUSH    AX

                                    ; MSDOS 3.3
                                    ;push   bx
                                    ;mov    bx,CharType
                                    ;; 09/08/2018
                                    ;;xlat  byte [cs:bx]
                                    ;cs     xlat
                                    ;pop    bx

                                    ; MSDOS 6.0
00005DC8 E80400                     CALL    GetCharType

                                    ;test   al,4
00005DCB A804                       TEST    AL,FSPCHK
00005DCD 58                         POP     AX
```

```
                            RETN
                        ;ELSE
                        ;   CMP     AL,9            ; Filter out tabs too
                        ;   retz
                        ;; WARNING! " " MUST be the last compare
                        ;   CMP     AL," "
                        ;   return
                        ;ENDIF

                        ;--------------------------------------------------------------------------
                        ;
                        ;  GetCharType - return flag bits indicating character type
                        ;   Bits are defined in DOSSYM.INC. Uses lookup table
                        ;   defined above at label CharType.
                        ;
                        ;   Entry : AL = character to return type flags for
                        ;   Exit  : AL = type flags
                        ;   Uses  : AL, flags
                        ;
                        ;--------------------------------------------------------------------------

                            ; 29/02/2024 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMDOS.COM)

                            ; 25/11/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)

                            ; 20/05/2019 - Retro DOS v4.0
                            ; MSDOS 6.0

                        GetCharType:
                            ;cmp    al,7Eh
00005DCF 3C7E               cmp     al,CharType_last        ; beyond end of table?
00005DD1 7314               jae     short gct_90            ; return standard value

00005DD3 53                 push    bx
00005DD4 BB[575D]           mov     bx,CharType             ; load lookup table
00005DD7 D0E8               shr     al,1                    ; adjust for half-byte table entry size
                            ;xlat   cs:[bx]                 ; get flags
00005DD9 2ED7               cs      xlat
00005DDB 5B                 pop     bx

                        ; carry clear from previous shift means we want the low nibble.  Otherwise
                        ; we have to shift the flags down to the low nibble

00005DDC 7306               jnc     short gct_80            ; carry clear, no shift needed

                        ; 29/02/2024
                        %if 0
                            shr     al,1                    ; we want high nibble, shift it down
                            shr     al,1
                            shr     al,1
                            shr     al,1
                        %else
                            ; 29/02/2024 (PCDOS 7.1 IBMDOS.COM)
                            ;;;
00005DDE 51                 push    cx
00005DDF B104               mov     cl,4                    ; we want high nibble, shift it down
00005DE1 D2E8               shr     al,cl
00005DE3 59                 pop     cx
                            ;;;
                        %endif

                        gct_80:
00005DE4 240F               and     al,0Fh                  ; clear the unused nibble
00005DE6 C3                 retn
                        gct_90:
00005DE7 B00F               mov     al,0Fh                  ; set all flags
00005DE9 C3                 retn

                        ;--------------------------------------------------------------------------
                        ;
                        ; Procedure : PATHCHRCMP
                        ;
                        ;--------------------------------------------------------------------------

                        PATHCHRCMP:
00005DEA 3C2F               CMP     AL,'/'
00005DEC 7606               JBE     short PathRet
00005DEE 3C5C               CMP     AL,'\'
00005DF0 C3                 retn
                        GotFor:
00005DF1 B05C               MOV     AL,'\'
00005DF3 C3                 retn
                        PathRet:
00005DF4 74FB               JZ      short GotFor
00005DF6 C3                 retn

                        ;==========================================================================
                        ; MSCRTLC.ASM, MSDOS 6.0, 1991
                        ;==========================================================================
                        ; 30/07/2018 - Retro DOS v3.0
                        ; 29/04/2019 - Retro DOS v4.0
                        ; 29/02/2024 - Retro DOS v5.0

                        ; 15/03/2018 - Retro DOS v2.0 (MSDOS 2.11, CTRLC.ASM, 1983)

                        ;** MSCTRLC.ASM - ^C and error handler for MSDOS

                        ;   TITLE   Control C detection, Hard error and EXIT routines
                        ;   NAME    IBMCTRLC

                        ;** Low level routines for detecting special characters on CON input,
                        ;   the ^C exit/int code, the Hard error INT 24 code, the
                        ;   process termination code, and the INT 0 divide overflow handler.
                        ;
                        ;   FATAL
                        ;   FATAL1
                        ;   reset_environment
                        ;   DSKSTATCHK
                        ;   SPOOLINT
                        ;   STATCHK
                        ;   CNTCHAND
                        ;   DIVOV
                        ;   CHARHARD
                        ;   HardErr
                        ;
                        ;   Revision history:
                        ;
                        ;       AN000       version 4.0   Jan 1988
                        ;       A002        PTM     -- dir >lpt3 hangs
                        ;       A003        PTM 3957- fake version for IBMCAHE.COM
                        ;
                        ;   M011: NEC's 8086 clone chip uses Intel's undocumented bit number in
                        ;         flags register. In order to return to user normally DOS used to
                        ;         move F202 into flags, which sets bit number 1 in flags uncondit-
```

```
35451                                   ;         ionally. Now it is modified to maintain the state of bit 1.
35452                                   ;
35453                                   ;   M024: suppressed fail and ignore options if not in the middle of int
35454                                   ;         24 and if Ctrl P or ctrl printscrn is pressed in routine
35455                                   ;         charhard.
35456                                   
35457                                   ; 29/04/2019 - Retro DOS v4.0
35458                                       ; MSDOS 6.0
35459                                   ;           public  LowInt23Addr
35460                                   LowInt23Addr: ;    LABEL   DWORD
35461 00005DF7 [F60F]0000                 DW      LowInt23, 0
35462                                   
35463                                   ;           public  LowInt24Addr
35464                                   LowInt24Addr: ;    LABEL   DWORD
35465 00005DFB [0A10]0000                 DW      LowInt24, 0
35466                                   
35467                                   ;           public  LowInt28Addr
35468                                   LowInt28Addr: ;    LABEL   DWORD
35469 00005DFF [1E10]0000                 DW      LowInt28, 0
35470                                   
35471                                   ;Break     <Checks for ^C in CON I/O>
35472                                   
35473                                   ; 25/11/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
35474                                   ; 05/05/2019 - Retro DOS v4.0
35475                                   
35476                                   ;--------------------------------------------------------------------------
35477                                   ;
35478                                   ; Procedure Name : DSKSTATCHK
35479                                   ;
35480                                   ; Check for ^C if only one level in
35481                                   ;
35482                                   ;--------------------------------------------------------------------------
35483                                   
35484                                           ;procedure DSKSTATCHK,NEAR ; Check for ^C if only one level in
35485                                   
35486                                       ; 29/02/2024 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMDOS.COM)
35487                                       ; PCDOS 7.1 IBMDOS.COM - DOSCODE:9F51h
35488                                   
35489                                   DSKSTATCHK:
35490 00005E03 36803E[2103]01                     cmp     byte [ss:INDOS], 1
35491 00005E09 7401                               jz      short dskstatchk1
35492 00005E0B C3                                 retn
35493                                   dskstatchk1:                            ; ...
35494 00005E0C 51                                 push    cx
35495 00005E0D 06                                 push    es
35496 00005E0E 53                                 push    bx
35497 00005E0F 1E                                 push    ds
35498 00005E10 56                                 push    si
35499 00005E11 8CD3                               mov     bx, ss
35500 00005E13 8EC3                               mov     es, bx
35501 00005E15 8EDB                               mov     ds, bx
35502 00005E17 C606[9403]05                       mov     byte [DSKSTCOM], 5  ; DEVRDND
35503 00005E1C C606[9203]0E                       mov     byte [DSKSTCALL], 14 ; DRDNDHL
35504 00005E21 C606[9503]00                       mov     byte [DSKSTST], 0
35505 00005E26 BB[9203]                           mov     bx, DSKSTCALL
35506 00005E29 C536[3200]                         lds     si, [BCON]
35507 00005E2D E865F4                             call    DEVIOCALL2
35508 00005E30 1E                                 push    ds
35509 00005E31 16                                 push    ss
35510 00005E32 1F                                 pop     ds
35511 00005E33 F606[9603]02                       test    byte [DSKSTST+1], 2
35512 00005E38 7409                               jz      short _GotCh
35513 00005E3A 30C0                               xor     al, al
35514                                   
35515                                   RET36:                                  ; ...
35516 00005E3C 5E                                 pop     si
35517 00005E3D 5E                                 pop     si
35518 00005E3E 1F                                 pop     ds
35519 00005E3F 5B                                 pop     bx
35520 00005E40 07                                 pop     es
35521 00005E41 59                                 pop     cx
35522 00005E42 C3                                 retn
35523                                   _GotCh:
35524 00005E43 A0[9F03]                           mov     al, [DSKCHRET]
35525 00005E46 3C03                               cmp     al, 3           ; "C"-"@"
35526 00005E48 75F2                               jnz     short RET36
35527 00005E4A C606[9403]04                       mov     byte [DSKSTCOM], 4  ; DEVRD
35528 00005E4F C606[9203]16                       mov     byte [DSKSTCALL], 16h ; DRDWRHL
35529 00005E54 880E[9F03]                         mov     [DSKCHRET], cl
35530 00005E58 C706[9503]0000                     mov     word [DSKSTST], 0
35531 00005E5E C706[A403]0100                     mov     word [DSKSTCNT], 1
35532 00005E64 1F                                 pop     ds
35533 00005E65 E82DF4                             call    DEVIOCALL2      ; Eat the ^C
35534 00005E68 5E                                 pop     si
35535 00005E69 1F                                 pop     ds
35536 00005E6A 5B                                 pop     bx
35537 00005E6B 07                                 pop     es
35538 00005E6C 59                                 pop     cx
35539 00005E6D E9CF00                             jmp     CNTCHAND
35540                                   
35541                                       ; 05/05/2019
35542                                   NOSTOP:
35543                                       ; MSDOS 6.0
35544 00005E70 3C10                         CMP     AL,"P"-"@"
35545 00005E72 7509                         JNZ     short check_next
35546                                                                           ; SS override
35547 00005E74 36803E[BC0D]00               CMP     BYTE [SS:SCAN_FLAG],0 ; ALT_Q ?
35548 00005E7A 7405                         JZ      short INCHKJ            ; no
35549                                   check_end:  ; 24/09/2023
35550 00005E7C C3                           retn
35551                                   check_next:
35552                                       ;IF     NOT TOGLPRN
35553                                       ;CMP    AL,"N"-"@"
35554                                       ;JZ     short INCHKJ
35555                                       ;ENDIF
35556                                   
35557 00005E7D 3C03                         CMP     AL,"C"-"@"
35558                                       ; 24/09/2023
35559                                       ;JZ     short INCHKJ
35560                                   ;check_end:
35561                                       ;retn
35562 00005E7F 75FB                         jnz     short check_end
35563                                   
35564                                       ; 24/09/2023
35565                                       ; 08/09/2018
35566                                   INCHKJ:     ; 10/08/2018
35567 00005E81 E9A500                       jmp     INCHK
35568                                   
35569                                       ; MSDOS 3.3
35570                                           ;CMP       AL,"P"-"@"  ; cmp al,16
35571                                           ;JZ short INCHKJ
35572                                   
35573                                       ; 15/04/2018
35574                                           ;;IF       NOT TOGLPRN
```

```
                                        ;CMP       AL,"N"-"@"
                                        ;JZ SHORT INCHKJ
                                        ;;ENDIF

                                    ;CMP    AL,"C"-"@"  ; cmp al,3
                                        ;JZ short INCHKJ
                                    ;RETN

                                ;   ; 08/09/2018
                                ;INCHKJ:; 10/08/2018
                                ;   JMP     INCHK

                                ;--------------------------------------------------------------------------
                                ;
                                ; Procedure Name : SpoolInt
                                ;
                                ; SpoolInt - signal processes that the DOS is truly idle. We are allowed to
                                ; do this ONLY if we are working on a 1-12 system call AND if we are not in
                                ; the middle of an INT 24.
                                ;
                                ;--------------------------------------------------------------------------

                                SPOOLINT:
00005E84 9C                         PUSHF
                                    ; 15/03/2018
00005E85 36803E[5803]00             CMP     BYTE [SS:IDLEINT],0   ; SS override
00005E8B 7423                       JZ      SHORT POPFRET
00005E8D 36803E[2003]00             CMP     BYTE [SS:ERRORMODE],0
00005E93 751B                       JNZ     SHORT POPFRET         ; No spool ints in error mode

                                    ; 30/07/2018

                                    ; Note that we are going to allow an external program to issue system
                                    ; calls at this time. We MUST preserve IdleInt across this.

00005E95 36FF36[5803]               PUSH    WORD [SS:IDLEINT]

                                    ; 05/05/2019 - Retro DOS v4.0

                                    ; MSDOS 6.0
00005E9A 36803E[660D]00             cmp     byte [SS:DosHasHMA],0 ; Q: is dos running in HMA (M021)
00005EA0 7504                       jne     short do_low_int28    ; Y: the int must be done from low mem
00005EA2 CD28                       INT     int_spooler  ; int 28h; N: Execute user int 28 handler
00005EA4 EB05                       jmp     short spool_ret_addr

                                do_low_int28:
                                    ;call   far [ss:LowInt28Addr]
00005EA6 2EFF1E[FF5D]               call    far [cs:LowInt28Addr] ; 05/05/2019

                                spool_ret_addr:
                                    ;INT    int_spooler           ; INT 28h

00005EAB 368F06[5803]               POP     WORD [SS:IDLEINT]
                                POPFRET:
00005EB0 9D                         POPF
                                _RET18:
00005EB1 C3                         RETN

                                ; 05/05/2019 - Retro DOS v4.0
                                ; DOSCODE:9137h (MSDOS 6.21, MSDOS.SYS)
                                ; 25/11/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
                                ; DOSCODE:90DBh (MSDOS 5.0, MSDOS.SYS)

                                ;--------------------------------------------------------------------------
                                ;
                                ; Procedure Name : STATCHK
                                ;
                                ;--------------------------------------------------------------------------

                                STATCHK:
00005EB2 E84EFF                         CALL       DSKSTATCHK              ; Allows ^C to be detected under
                                                                           ; input redirection
00005EB5 53                             PUSH    BX
00005EB6 31DB                           XOR     BX,BX
00005EB8 E80EE0                         CALL       GET_IO_SFT
00005EBB 5B                             POP     BX
00005EBC 72F3                           JC      SHORT _RET18

00005EBE B401                           MOV     AH,1
00005EC0 E805F2                         CALL       IOFUNC
00005EC3 74BF                           JZ      SHORT SPOOLINT
00005EC5 3C13                           CMP     AL,'S'-'@'
00005EC7 75A7                           JNZ     SHORT NOSTOP

                                    ; 05/05/2019
                                    ; MSDOS 6.0                       ; SS override
00005EC9 36803E[BC0D]00             CMP     BYTE [SS:SCAN_FLAG],0 ; AN000; ALT_R ?
00005ECF 75AB                       JNZ     short check_end               ; AN000; yes

00005ED1 30E4                           XOR     AH,AH
00005ED3 E8F2F1                         CALL       IOFUNC                  ; Eat Cntrl-S
00005ED6 EB4A                           JMP     SHORT PAUSOSTRT
                                PRINTOFF:
                                PRINTON:
00005ED8 36F616[FE02]               NOT     BYTE [SS:PFLAG] ; 14/03/2018

                                    ; 30/07/2018 - Retro DOS v3.0
00005EDD 53                         PUSH    BX
00005EDE BB0400                     MOV     BX,4
00005EE1 E8E5DF                     call    GET_IO_SFT
00005EE4 5B                         POP     BX
00005EE5 72CA                       jc      short _RET18
00005EE7 06                         PUSH    ES
00005EE8 57                         PUSH    DI
00005EE9 1E                         PUSH    DS
00005EEA 07                         POP     ES
00005EEB 89F7                       MOV     DI,SI                 ; ES:DI -> SFT
                                    ;test   word [es:di+5],800h
                                    ;TEST   word [ES:DI+SF_ENTRY.sf_flags],sf_net_spool
                                    ; 05/05/2019
00005EED 26F6450608                 test    byte [ES:DI+SF_ENTRY.sf_flags+1],(sf_net_spool>>8)
00005EF2 7418                       JZ      short NORM_PR         ; Not redirected, echo is OK

                                    ;Callinstall NetSpoolEchoCheck,MultNet,38,<AX>,<AX>
                                                                  ; See if allowed
00005EF4 50                         push    ax
00005EF5 B82611                     mov     ax,1126h
00005EF8 CD2F                       int     2Fh     ; Multiplex - NETWORK REDIRECTOR - ???
                                                    ; Return: CF set on error, AX = error code
                                                    ; STACK unchanged
00005EFA 58                         pop     ax

00005EFB 730F                       JNC     short NORM_PR         ; Echo is OK
```

```
                                                        ; SS override
00005EFD 36C606[FE02]00             MOV     BYTE [SS:PFLAG],0     ; If not allowed, disable echo

                                    ;Callinstall NetSpoolClose,MultNet,36,<AX>,<AX> ; and close

00005F03 50                         push    ax
00005F04 B82411                     mov     ax,1124h
00005F07 CD2F                       int     2Fh     ; Multiplex - NETWORK REDIRECTOR - ???
                                                    ; ES:DI -> SFT, SS = DOS CS
00005F09 58                         pop     ax

00005F0A EB10                       JMP     SHORT RETP6
                                NORM_PR:
00005F0C 36803E[FE02]00             CMP     BYTE [SS:PFLAG],0     ; SS override
00005F12 7505                       JNZ     short PRNOPN
00005F14 E805F3                     call    DEV_CLOSE_SFT
00005F17 EB03                       JMP     SHORT RETP6
                                PRNOPN:
00005F19 E8F8F2                     call    DEV_OPEN_SFT
                                RETP6:
00005F1C 5F                         POP     DI
00005F1D 07                         POP     ES
                                STATCHK_RETN:
00005F1E C3                             RETN
                                PAUSOLP:
00005F1F E862FF                         CALL    SPOOLINT
                                PAUSOSTRT:
00005F22 B401                           MOV     AH,1
00005F24 E8A1F1                         CALL       IOFUNC
00005F27 74F6                           JZ      SHORT PAUSOLP
                                INCHK:
00005F29 53                             PUSH    BX
00005F2A 31DB                           XOR     BX,BX
00005F2C E89ADF                         CALL       GET_IO_SFT
00005F2F 5B                             POP     BX
00005F30 72EC                           JC      SHORT STATCHK_RETN ; 30/07/2018
00005F32 30E4                           XOR     AH,AH
00005F34 E891F1                         CALL       IOFUNC
                                        ; 30/07/2018
                                        ; MSDOS 3.3
                                           ;CMP       AL,'P'-'@' ;cmp al,16
                                           ;JNZ       SHORT NOPRINT

                                    ;cmp    byte [SS:SCAN_FLAG],0
                                    ;JZ     SHORT PRINTON
                                    ;mov    byte [ss:SCAN_FLAG],0

                                    ; 05/05/2019
                                    ; MSDOS 6.0
00005F37 3C10                       CMP     AL,"P"-"@"
                                    ;;;;  7/14/86  ALT_Q key fix
00005F39 749D                       JZ      short PRINTON         ; no! must be CTRL_P
                                ;NOPRINT:
                                    ;IF     NOT TOGLPRN
                                    ;CMP    AL,"N"-"@"
                                    ;JZ     short PRINTOFF
                                    ;ENDIF
00005F3B 3C03                       CMP     AL,"C"-"@" ; cmp al,3
                                    ;retnz
00005F3D 75DF                       jnz     short STATCHK_RETN

                                    ; !! NOTE: FALL THROUGH !!

                                ;--------------------------------------------------------------------------
                                ;
                                ; Procedure Name : CNTHAND ( CTRLC_C HANDLER )
                                ;
                                ; "^C" and CR/LF is printed. Then the user registers are restored and the
                                ; user CTRL-C handler is executed. At this point the top of the stack has 1)
                                ; the interrupt return address should the user CTRL-C handler wish to allow
                                ; processing to continue; 2) the original interrupt return address to the code
                                ; that performed the function call in the first place. If the user CTRL-C
                                ; handler wishes to continue, it must leave all registers unchanged and RET
                                ; (not IRET) with carry CLEAR. If carry is SET then an terminate system call
                                ; is simulated.
                                ;
                                ;--------------------------------------------------------------------------

                                    ; 29/02/2024 - Retro DOS v5.0

                                CNTCHAND:
                                    ; MSDOS 6.0                   ; SS override
                                                                  ; AN002; from RAWOUT
                                    ;TEST   word [SS:DOS34_FLAG],CTRL_BREAK_FLAG
                                    ;JNZ    short around_deadlock ; AN002;

                                    ; 05/05/2019 - Retro DOS v4.0
                                    ; (MSDOS 6.21 MSDOS.SYS DOSCODE:91C4h, 29/12/2022)
00005F3F 36F606[1206]02             TEST    byte [SS:DOS34_FLAG+1],(CTRL_BREAK_FLAG>>8)  ; 2
00005F45 7508                       JNZ     short around_deadlock ; AN002;

00005F47 B003                           MOV     AL,3                    ; Display "^C"
00005F49 E879BD                         CALL       BUFOUT
00005F4C E80EBC                         CALL       CRLF
                                around_deadlock:
00005F4F 16                             PUSH    SS
00005F50 1F                             POP     DS
00005F51 803E[5703]00                   CMP     BYTE [CONSWAP],0
00005F56 7403                           JZ      SHORT NOSWAP
00005F58 E802DC                         CALL       SWAPBACK
                                NOSWAP:
00005F5B FA                         CLI                           ; Prepare to play with stack
00005F5C 8E16[8605]                 MOV     SS,[USER_SS]          ; User stack now restored
00005F60 8B26[8405]                 MOV     SP,[USER_SP]
00005F64 E8DAA4                         CALL       restore_world       ; User registers now restored

                                    ; 30/07/2018 - Retro DOS v3.0
                                    ; MSDOS 3.3 (IBMDOS.COM - Offset 56ACh)
                                       ; 14/03/2018 - Retro DOS v2.0
                                    ;MOV    BYTE [CS:INDOS],0
                                       ;MOV       BYTE [CS:ERRORMODE],0
                                       ;MOV       [CS:ConC_Spsave],SP
                                    ;clc    ;30/07/2018
                                       ;INT       int_ctrl_c ; 23h    ; Execute user Ctrl-C handler
                                    ;;int  23h     ; DOS - CONTROL "C" EXIT ADDRESS
                                                   ; Return: return via RETF 2 with CF set
                                                   ; DOS will abort program with errorlevel 0
                                                   ; else
                                                   ; interrupted DOS call continues

                                    ; 05/05/2019 - Retro DOS v4.0
                                    ; MSDOS 6.0 (MSDOS 6.21, MSDOS.SYS,91ECh)

                                    ; CS was used to address these variables. We have to use DOSDATA
```

```
35823
35824 00005F67 07                       pop     es ; *  ; MSDOS 6.21 (MSDOS.SYS, DOSCODE:91ECh)
35825                                                   ; (pop es, after 'call restore_world')
35826 00005F68 1E                       push    ds
35827                                   ;getdseg <ds>                   ; ds -> dosdata
35828 00005F69 2E8E1E[0700]             mov     ds,[cs:DosDSeg]
35829 00005F6E C606[2103]00             mov     byte [INDOS],0          ; Go to known state
35830                                   ; 29/02/2024 (PCDOS 7.1 IBMDOS.COM)
35831                                   ;;;
35832 00005F73 C606[B812]00             mov     byte [INDOS_FLAG],0     ; 2nd 'in dos' flag (what for?)
35833                                   ;;;
35834 00005F78 C606[2003]00             mov     byte [ERRORMODE],0
35835 00005F7D 8926[3203]               mov     [ConC_Spsave],SP        ; save his SP
35836                                   ; User SP has changed because of push. Adjust for it
35837 00005F81 8306[3203]02             add     word [ConC_Spsave],2
35838
35839 00005F86 803E[660D]00             cmp     byte [DosHasHMA],0      ; Q: is dos running in HMA (M021)
35840 00005F8B 1F                       pop     ds      ; restore ds
35841 00005F8C 7505                     jne     short do_low_int23      ; Y: the int must be done from low mem
35842 00005F8E F8                       CLC
35843 00005F8F CD23                     INT     int_ctrl_c  ; int 23h ; N: Execute user Ctrl-C handler
35844 00005F91 EB06                     jmp     short ctrlc_ret_addr
35845
35846                                   ; 05/05/2019
35847                                 do_low_int23:
35848 00005F93 F8                       clc
35849 00005F94 2EFF1E[F75D]             call    far [cs:LowInt23Addr]
35850
35851                                   ; 30/07/2018
35852
35853                                   ; MSDOS 3.3 (IBMDOS.COM - Offset 56C0h)
35854
35855                                 ; The user has returned to us. The circumstances we allow are:
35856                                 ;
35857                                 ;   IRET    We retry the operation by redispatching the system call
35858                                 ;   CLC/RETF        POP the stack and retry
35859                                 ;   ...     Exit the current process with ^C exit
35860                                 ;
35861                                 ; User's may RETURN to us and leave interrupts on.
35862                                 ; Turn 'em off just to be sure
35863
35864                                 ctrlc_ret_addr: ; 05/05/2019
35865
35866 00005F99 FA                       CLI
35867
35868                                   ; MSDOS 3.3
35869                                   ;MOV    [CS:USER_IN_AX],ax      ; save the AX
35870                                   ;PUSHF                          ; and the flags (maybe new call)
35871                                   ;POP    AX
35872
35873                                   ; 05/05/2019
35874                                   ; MSDOS 6.0
35875
35876                                   ; We have to use DOSDATA for these variables. Previously CS was used
35877
35878 00005F9A 50                       push    ax
35879 00005F9B 8CD8                     mov     ax,ds
35880                                   ;getdseg <ds>                   ; ds -> dosdata
35881 00005F9D 2E8E1E[0700]             mov     ds,[cs:DosDSeg]
35882 00005FA2 A3[630D]                 mov     [TEMPSEG],ax
35883 00005FA5 58                       pop     ax
35884 00005FA6 A3[3A03]                 MOV     [USER_IN_AX],ax                 ; save the AX
35885 00005FA9 9C                       pushf                           ; and the flags (maybe new call)
35886 00005FAA 58                       pop     ax
35887
35888                                 ; See if the input stack is identical to the output stack
35889
35890                                   ; MSDOS 3.3
35891                                   ;CMP    SP,[CS:ConC_Spsave]
35892                                   ;JNZ    SHORT ctrlc_try_new     ; current SP not the same as saved SP
35893
35894                                   ; MSDOS 6.0
35895 00005FAB 3B26[3203]               CMP     SP,[ConC_Spsave]
35896 00005FAF 750A                     JNZ     SHORT ctrlc_try_new     ; current SP not the same as saved SP
35897
35898                                 ; Repeat the operation by redispatching the system call.
35899
35900                                 ctrlc_repeat:
35901                                   ; MSDOS 3.3
35902                                   ;MOV    AX,[CS:USER_IN_AX]
35903                                   ; 05/05/2019
35904                                   ; MSDOS 6.0
35905 00005FB1 A1[3A03]                 mov     ax,[USER_IN_AX]
35906 00005FB4 8E1E[630D]               mov     ds,[TEMPSEG]            ; restore ds and original sp
35907                                   ; MSDOS 3.3 & MSDOS 6.0
35908                                   ;transfer COMMAND
35909                                 COMMANDJ:
35910 00005FB8 E936A3                   JMP     COMMAND
35911
35912                                 ; The current SP is NOT the same as the input SP. Presume that he
35913                                 ; RETF'd leaving some flags on the stack and examine the input
35914
35915                                 ctrlc_try_new:
35916                                   ; 29/02/2024
35917                                   ;ADD    SP,2                    ; pop those flags
35918                                   ;
35919                                   ;;test  ax,1
35920                                   ;TEST   AX,f_Carry              ; did he return with carry?
35921 00005FBB A801                     test    al,f_Carry ; test al,1
35922                                   ;
35923                                   ; 29/02/2024
35924 00005FBD 58                       pop     ax  ;  (PCDOS 7.1 IBMDOS.COM)
35925                                   ;
35926 00005FBE 74F1                     JZ      short ctrlc_repeat      ; no carry set, just retry
35927
35928                                   ; MSDOS 6.0
35929 00005FC0 8E1E[630D]               mov     ds,[TEMPSEG]            ; restore ds
35930
35931                                   ; Well...  time to abort the user.
35932                                   ; Signal a ^C exit and use the EXIT system call..
35933
35934                                 ctrlc_abort:
35935                                   ; MSDOS 3.3
35936                                      ;;MOV      AX,(EXIT SHL 8) + 0
35937                                      ;MOV       AX,(EXIT*256) + 0  ; 4C00h
35938                                   ;mov    byte [CS:DidCTRLC],0FFh ; 14/03/2018
35939                                      ;transfer COMMAND       ; give up by faking $EXIT
35940                                   ;;JMP   SHORT COMMANDJ
35941                                   ;JMP    COMMAND
35942
35943                                   ; 05/05/2019 - Retro DOS v4.0
35944                                   ; MSDOS 6.0
35945 00005FC4 B8004C                   MOV     AX,(EXIT<<8)+0 ; 4C00h
35946 00005FC7 1E                       push    ds
```

```
                                    ;getdseg <ds>                ; ds -> dosdata
00005FC8 2E8E1E[0700]               mov     ds,[cs:DosDSeg]
00005FCD C606[4D03]FF               MOV     byte [DidCTRLC],-1 ; 0FFh
00005FD2 1F                         pop     ds
                                    ;transfer COMMAND            ; give up by faking $EXIT
00005FD3 EBE3                       JMP     SHORT COMMANDJ
                                    ;JMP    COMMAND

                                ;Break      <DIVISION OVERFLOW INTERRUPT>
                                ;--------------------------------------------------------------------------
                                ;
                                ; Procedure Name : DIVOV
                                ;
                                ; Default handler for division overflow trap
                                ;
                                ;--------------------------------------------------------------------------

                                    ; 25/11/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
                                DIVOV:
                                    ; 05/05/2019 - Retro DOS v4.0
                                    ; 30/07/2018
                                    ; 07/07/2018 - Retro DOS v3.0
00005FD5 BE[1B0A]                   mov     si,DIVMES
00005FD8 2E8B1E[2E0A]               mov     bx,[cs:DivMesLen]
                                    ;mov    ax,cs
                                    ;mov    ss,ax
                                    ; 05/05/2019
                                    ;getdseg <ss>          ; we are in an ISR, flag is CLI
00005FDD 2E8E16[0700]               mov     ss,[cs:DosDSeg]
00005FE2 BC[A007]                   mov      sp,AUXSTACK
                                    ;call   RealDivOv ; MSDOS 3.3
00005FE5 E80200                     call    _OUTMES ; MSDOS 6.0
00005FE8 EBDA                       jmp     short ctrlc_abort  ; Use Ctrl-C abort on divide overflow

                                ; 30/07/2018

                                ; MSDOS 6.0
                                ;--------------------------------------------------------------------------
                                ;
                                ; Procedure Name : OutMes
                                ;
                                ;
                                ; OutMes: perform message output
                                ; Inputs:   SS:SI points to message
                                ;       BX has message length
                                ; Outputs:  message to BCON
                                ;
                                ;Actually, cs:si points to the message now. The segment address is filled in
                                ;at init. time ([dskchret+2]). This will be temporarily changed to DOSCODE.
                                ;NB. This procedure is called only from DIVOV. -SR
                                ;
                                ;--------------------------------------------------------------------------

                                ;MSDOS 3.3
                                ;--------------------------------------------------------------------------
                                ; RealDivOv: perform actual divide overflow stuff.
                                ; Inputs:   none
                                ; Outputs:  message to BCON
                                ;--------------------------------------------------------------------------

                                    ; 05/05/2019 - Retro DOS v4.0
                                    ; DOSCODE:926Ch (MSDOS 6.21, MSDOS.SYS)

                                    ; 25/11/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
                                    ; DOSCODE:9210h (MSDOS 5.0, MSDOS.SYS)

                                ;--------------------------------------------------------------------------
                                ;
                                ; Procedure Name : OutMes
                                ;
                                ; OutMes: perform message output
                                ; Inputs:   SS:SI points to message
                                ;       BX has message length
                                ; Outputs:  message to BCON
                                ;
                                ;Actually, cs:si points to the message now. The segment address is filled in
                                ;at init. time ([dskchret+2]). This will be temporarily changed to DOSCODE.
                                ;NB. This procedure is called only from DIVOV. -SR
                                ;
                                ;--------------------------------------------------------------------------

                                    ; 30/07/2018
                                    ; MSDOS 6.0
                                _OUTMES:
                                    ; MSDOS 3.3
                                ;RealDivOv:
                                    ; 07/07/2018 - Retro DOS v3.0
                                    ;Context ES
00005FEA 16                     push    ss ; 05/05/2019
                                ;PUSH   CS ; 30/07/2018              ; get ES addressability
00005FEB 07                     POP     ES
                                    ;Context DS
00005FEC 16                     push    ss ; 05/05/2019
                                ;PUSH   CS ; 30/07/2018              ; get DS addressability
00005FED 1F                     POP     DS
00005FEE C606[9403]08               MOV     BYTE [DSKSTCOM],DEVWRT
00005FF3 C606[9203]16               MOV     BYTE [DSKSTCALL],DRDWRHL
00005FF8 C706[9503]0000             MOV     WORD [DSKSTST],0
                                ; BX = [DivMesLen] = 19
00005FFE 891E[A403]                 MOV     [DSKSTCNT],BX
00006002 BB[9203]                   MOV     BX,DSKSTCALL
00006005 8936[A003]                 MOV     [DSKCHRET+1],SI          ; transfer address (need an EQU)
                                ; 08/09/2018
                                ;mov    [DEVIOBUF_PTR],si
                                ; MSDOS 6.0
                                                                     ; CS is used for string, fill in
                                                                     ; segment address
                                ;mov    [DOSSEG_INIT],cs ; 29/02/2024
00006009 8C0E[A203]             MOV     [DSKCHRET+3],CS

                                    LDS     SI,[BCON]
00006011 E881F2                     CALL       DEVIOCALL2

                                ;; 14/03/2018
                                    ;;MOV      WORD [CS:DSKCHRET+1],DEVIOBUF
                                ;; 08/09/2018
                                ;mov    word [CS:DEVIOBUF_PTR],DEVIOBUF
                                    ;MOV       WORD [CS:DSKSTCNT],1

                                ; 05/05/2019 - Retro DOS v4.0 (MSDOS 6.0, MSDOS 6.21)

                                ; ES still points to DOSDATA. ES is
                                                                     ; not destroyed by deviocall2. So use
                                                                     ; ES override.
```

```
36071
36072 00006014 26C706[A003][BC03]          MOV     WORD [ES:DSKCHRET+1],DEVIOBUF
36073 0000601B 26C706[A403]0100           MOV     WORD [ES:DSKSTCNT],1
36074
36075 00006022 C3                         RETN
36076
36077                                   ;Break      <CHARHRD,HARDERR,ERROR -- HANDLE DISK ERRORS AND RETURN TO USER>
36078                                   ;--------------------------------------------------------------------------
36079                                   ;
36080                                   ; Procedure Name : CHARHARD
36081                                   ;
36082                                   ;
36083                                   ; Character device error handler
36084                                   ; Same function as HARDERR
36085                                   ;
36086                                   ;--------------------------------------------------------------------------
36087
36088                                       ; 25/11/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
36089                                   CHARHARD:
36090                                       ; 05/05/2019 - Retro DOS v4.0
36091                                       ; 30/07/2018
36092                                       ; 08/07/2018 - Retro DOS v3.0
36093
36094                                       ; MSDOS 6.0
36095                                                                   ; M024 - start
36096 00006023 36803E[2003]00             cmp     byte [SS:ERRORMODE], 0; Q: are we in the middle of int 24
36097                                      ;jne    short @f              ; Y: allow fail
36098 00006029 750B                        jne     short chard1
36099
36100 0000602B 80CC10                      OR      AH,Allowed_RETRY ; 10h; assume ctrl p
36101
36102 0000602E 36F606[FE02]FF              test    byte [ss:PFLAG],-1    ; Q: has ctrl p been pressed
36103 00006034 7503                        jnz     short ctrlp           ; Y:
36104                                   ;@@:
36105                                   chard1:                                 ; M024 - end
36106                                       ; MSDOS 6.0 & MSDOS 3.3
36107
36108                                   ; Character device error handler
36109                                   ; Same function as HARDERR
36110
36111                                       ;or     ah,38h
36112 00006036 80CC38                      or      ah,Allowed_IGNORE+Allowed_RETRY+Allowed_FAIL
36113                                   ctrlp:                  ; SS override for Allowed and EXITHOLD
36114 00006039 368826[4B03]                mov     [SS:ALLOWED],ah
36115
36116                                       ; 15/03/2018
36117 0000603E 368C06[8205]                   MOV     [SS:EXITHOLD+2],ES
36118 00006043 36892E[8005]                   MOV     [SS:EXITHOLD],BP
36119 00006048 56                             PUSH    SI
36120                                      ;and    di,0FFh
36121 00006049 81E7FF00                       AND     DI,STECODE
36122 0000604D 8CDD                           MOV     BP,DS                   ;Device pointer is BP:SI
36123 0000604F E89D00                         CALL    FATALC
36124 00006052 5E                             POP     SI
36125                                      ;return
36126 00006053 C3                             RETN
36127
36128                                   ;--------------------------------------------------------------------------
36129                                   ;
36130                                   ; Procedure Name : HardErr
36131                                   ;
36132                                   ; Hard disk error handler. Entry conditions:
36133                                   ;   DS:BX = Original disk transfer address
36134                                   ;   DX = Original logical sector number
36135                                   ;   CX = Number of sectors to go (first one gave the error)
36136                                   ;   AX = Hardware error code
36137                                   ;   DI = Original sector transfer count
36138                                   ;   ES:BP = Base of drive parameters
36139                                   ;   [READOP] = 0 for read, 1 for write
36140                                   ;   Allowed Set with allowed responses to this error (other bits MUST BE 0)
36141                                   ; Output:
36142                                   ;   [FAILERR] will be set if user responded FAIL
36143                                   ;
36144                                   ;--------------------------------------------------------------------------
36145
36146                                       ; 25/11/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
36147                                       ; 29/02/2024 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMDOS.COM)
36148                                   HARDERR:
36149                                       ; 05/05/2019 - Retro DOS v4.0
36150                                       ; 30/07/2018
36151                                       ; 08/07/2018 - Retro DOS v3.0
36152 00006054 97                          XCHG    AX,DI                   ; Error code in DI, count in AX
36153                                      ;and    di,0FFh
36154 00006055 81E7FF00                    AND     DI,STECODE              ; And off status bits
36155                                      ;CMP    DI,WRECODE              ; Write Protect Error?
36156                                      ;cmp    di,0
36157 00006059 83FF00                      cmp     DI,error_I24_write_protect ; Write Protect Error?
36158 0000605C 750A                        JNZ     short NOSETWRPERR
36159 0000605E 50                          PUSH    AX
36160                                      ; 25/11/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
36161                                      ;MOV    AL,[ES:BP+DPB.DRIVE]
36162                                      ;;MOV   AL,[ES:BP+0]
36163                                      ; 15/12/2022
36164 0000605F 268A4600                    mov     al,[ES:BP]
36165                                      ; 15/03/2018
36166 00006063 36A2[2203]                  MOV     [SS:WPERR],AL           ; Flag drive with WP error
36167 00006067 58                          POP     AX
36168                                   NOSETWRPERR:
36169 00006068 29C8                        SUB     AX,CX                   ; Number of sectors successfully transferred
36170 0000606A 01C2                        ADD     DX,AX                   ; First sector number to retry
36171                                      ; 29/02/2024 (PCDOS 7.1 IBMDOS.COM)
36172                                      ;;;
36173 0000606C 368316[0706]00              adc     word [ss:HIGH_SECTOR],0
36174                                      ;;;
36175 00006072 52                          PUSH    DX
36176                                      ; 08/07/2018
36177                                      ;MUL    word [ES:BP+2]                  ; Number of bytes transferred
36178 00006073 26F76602                    MUL     word [ES:BP+DPB.SECTOR_SIZE]
36179 00006077 5A                          POP     DX
36180 00006078 01C3                        ADD     BX,AX                   ; First address for retry
36181                                      ;XOR    AH,AH ; *               ; Flag disk section in error
36182                                      ; 29/02/2024 (PCDOS 7.1 IBMDOS.COM)
36183                                      ;;;
36184 0000607A 31C0                        xor     ax,ax ; * ; 29/02/2024 - Retro DOS v5.0
36185                                      ;cmp    word [ss:HIGH_SECTOR],0
36186 0000607C 363906[0706]                cmp     [ss:HIGH_SECTOR],ax ; 0 ; *
36187 00006081 7506                        jnz     short TESTDIR
36188                                      ;;;
36189                                      ;CMP    DX,[ES:BP+6]            ; In reserved area?
36190 00006083 263B5606                    CMP     DX,[ES:BP+DPB.FIRST_FAT]
36191 00006087 7246                        JB      SHORT ERRINT
36192
36193                                   TESTDIR:    ; 29/02/2024
36194 00006089 FEC4                        INC     AH                      ; Flag for FAT
```

```
36195
36196                                   ; 29/02/2024 - Retro DOS v5.0 (PCDOS 7.1 IBMDOS.COM)
36197                                   ;;;
36198                                   ;cmp    word [es:bp+0Fh],0
36199 0000608B 26837E0F00               cmp     word [es:bp+DPB.FAT_SIZE],0
36200 00006090 7525                     jnz     short TESTDIR3 ; not FAT32
36201 00006092 52                       push    dx
36202 00006093 368B16[0706]             mov     dx,[ss:HIGH_SECTOR]
36203                                   ;cmp    dx,[es:bp+2Bh]
36204 00006098 263B562B                 cmp     dx,[es:bp+DPB.FCLUS_FSECTOR+2]
36205 0000609C 5A                       pop     dx
36206 0000609D 7504                     jnz     short TESTDIR1 ; Err in FAT must force recomp of freespace
36207                                   ;cmp    dx,[es:bp+29h]
36208 0000609F 263B5629                 cmp     dx,[es:bp+DPB.FCLUS_FSECTOR]
36209                                 TESTDIR1:
36210 000060A3 730E                     jnb     short TESTDIR2
36211                                   ;mov    word [es:bp+1Fh],0FFFFh ; -1
36212 000060A5 26C7461FFFFF             mov     word [es:bp+DPB.FREE_CNT],0FFFFh
36213                                   ;mov    word [es:bp+21h],0FFFFh ; -1
36214 000060AB 26C74621FFFF             mov     word [es:bp+DPB.FREE_CNT+2],0FFFFh
36215 000060B1 EB1C                     jmp     short ERRINT
36216                                 TESTDIR2:
36217 000060B3 FEC4                     inc     ah
36218                                   ;inc    ah
36219                                   ;jmp    short ERRINT
36220                                   ; 29/02/2024 - Retro DOS v5.0
36221 000060B5 EB16                     jmp     short ERRINT2
36222                                 TESTDIR3:
36223                                   ;;;
36224
36225                                   ;CMP    DX,[ES:BP+10H] ; MSDOS 3.3
36226                                   ;cmp    dx,[ES:BP+11h] ; MSDOS 6.0 - 05/05/2019
36227 000060B7 263B5611                 CMP     DX,[ES:BP+DPB.DIR_SECTOR] ; In FAT?
36228                                   ;JAE    short TESTDIR          ; No
36229 000060BB 7308                     jae     short TESTDIR4 ; 29/02/2024
36230
36231                                           ; Err in FAT must force recomp of freespace
36232                                   ;mov    word [ES:BP+1Eh],-1 ; MSDOS 3.3
36233                                   ;mov    word [ES:BP+1Fh],-1 ; MSDOS 6.0 - 05/05/2019
36234 000060BD 26C7461FFFFF             MOV     word [ES:BP+DPB.FREE_CNT],-1
36235
36236 000060C3 EB0A                     JMP     SHORT ERRINT
36237                                 ;TESTDIR:
36238                                 TESTDIR4:   ; 29/02/2024
36239 000060C5 FEC4                     INC     AH
36240                                   ;CMP    DX,[ES:BP+0BH]         ; In directory?
36241 000060C7 263B560B                 CMP     DX,[ES:BP+DPB.FIRST_SECTOR]
36242 000060CB 7202                     JB      SHORT ERRINT
36243                                 ERRINT2:    ; 29/02/2024
36244 000060CD FEC4                     INC     AH                      ; Must be in data area
36245                                 ERRINT:
36246 000060CF D0E4                     SHL     AH,1                    ; Make room for read/write bit
36247 000060D1 360A26[7505]             OR      AH,[SS:READOP] ; 15/03/2018
36248
36249                                   ; 15/08/2018
36250                                                                   ; SS override for allowed and EXITHOLD
36251 000060D6 360A26[4B03]             OR      AH,[SS:ALLOWED]                 ; Set the allowed_ bits
36252
36253                                   ;entry   FATAL
36254                                 FATAL:
36255                                   ; 25/11/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
36256                                   ;MOV    AL,[ES:BP+DPB.DRIVE]
36257                                   ;;MOV   AL,[ES:BP+0]           ; Get drive number
36258                                   ; 15/12/2022
36259 000060DB 268A4600                 MOV     AL,[ES:BP]
36260
36261                                   ;entry   FATAL1
36262                                 FATAL1:
36263                                   ; 15/03/2018
36264 000060DF 368C06[8205]             MOV     [SS:EXITHOLD+2],ES
36265 000060E4 36892E[8005]             MOV     [SS:EXITHOLD],BP        ; The only things we preserve
36266                                   ;LES    SI,[ES:BP+12H] ; MSDOS 3.3
36267                                   ;LES    SI,[ES:BP+13H] ; MSDOS 6.0 - 05/05/2019
36268 000060E9 26C47613                 LES     SI,[ES:BP+DPB.DRIVER_ADDR]
36269 000060ED 8CC5                     MOV     BP,ES                   ; BP:SI points to the device involved
36270
36271                                   ; DI has the INT-24-style extended error. We now map the error code
36272                                   ; for this into the normalized get extended error set by using the
36273                                   ; ErrMap24 table as a translate table. Note that we translate ONLY
36274                                   ; the device returned codes and leave all others beyond the look up
36275                                   ; table alone.
36276
36277                                   ; 08/07/2018 - Retro DOS v3.0
36278                                 FATALC:
36279 000060EF E89F01                   call    SET_I24_EXTENDED_ERROR
36280                                   ;cmp    di,0Ch
36281 000060F2 83FF0C                   CMP     DI,error_I24_gen_failure
36282 000060F5 7603                     JBE     short GOT_RIGHT_CODE  ; Error codes above gen_failure get
36283 000060F7 BF0C00                   MOV     DI,error_I24_gen_failure; mapped to gen_failure. Real codes
36284                                                                   ;  Only come via GetExtendedError
36285
36286                                 ;** ----------------------------------------------------------------
36287                                 ;
36288                                 ; Entry point used by REDIRector on Network I 24 errors.
36289                                 ;
36290                                 ;   ASSUME  DS:NOTHING,ES:NOTHING,SS:DOSDATA
36291                                 ;
36292                                 ; ALL I 24 regs set up. ALL Extended error info SET. ALLOWED Set.
36293                                 ;   EXITHOLD set for restore of ES:BP.
36294                                 ; ----------------------------------------------------------------
36295                                   ;entry  NET_I24_ENTRY
36296                                 NET_I24_ENTRY:
36297                                 GOT_RIGHT_CODE:
36298 000060FA 36803E[2003]00           CMP     BYTE [SS:ERRORMODE],0 ; No INT 24s if already INT 24
36299 00006100 7404                     JZ      SHORT NoSetFail
36300 00006102 B003                     MOV     AL,3
36301 00006104 EB74                     JMP     short FailRet
36302                                 NoSetFail:
36303 00006106 368926[8805]             MOV     [SS:CONTSTK],SP                 ; SS override
36304 0000610B 16                       PUSH    SS
36305 0000610C 07                       POP     ES
36306
36307                                   ; Wango!!! We may need to free some user state info... In
36308                                   ; particular, we may have locked down a JFN for a user and he may
36309                                   ; NEVER return to us. Thus,we need to free it here and then
36310                                   ; reallocate it when we come back.
36311
36312 0000610D 36833E[AA05]FF           CMP     word [SS:SFN],-1 ; 0FFFFh
36313 00006113 740C                     JZ      short _NoFree
36314 00006115 1E                       push    ds
36315 00006116 56                       push    si
36316 00006117 36C536[AE05]             LDS     SI,[SS:PJFN]
36317 0000611C C604FF                   MOV     BYTE [SI],0FFH
36318 0000611F 5E                       pop     si
```

```
36319 00006120 1F                      pop     ds
36320
36321                                  _NoFree:
36322 00006121 FA                      CLI
36323                                                  ; Prepare to play with stack
36324 00006122 36FE06[2003]            INC     BYTE [SS:ERRORMODE]    ; Flag INT 24 in progress
36325 00006127 36FE0E[2103]            DEC     BYTE [SS:INDOS]                ; INT 24 handler might not return
36326
36327                                  ; 29/02/2024 - Retro DOS v5.0 (PCDOS 7.1 IBMDOS.COM)
36328                                  ;;;
36329 0000612C 36FE0E[B812]            dec     byte [ss:INDOS_FLAG]  ; 2nd 'in dos' flag (what for?)
36330                                  ;;;
36331
36332                                  ; 05/05/2019 - Retro DOS v4.0 (MSDOS 6.0, MSDOS 6.21)
36333
36334                                  ;; Extended Open hooks
36335                                                  ; AN000;IFS.I24 error disabled
36336                                  ;test  byte [ss:EXTOPEN_ON],2
36337 00006131 36F606[F605]02          TEST    byte [ss:EXTOPEN_ON],EXT_OPEN_I24_OFF
36338 00006137 7404                    JZ      short i24yes           ; AN000;IFS.no
36339                                  faili24:                       ; AN000;
36340 00006139 B003                    MOV     AL,3                   ; AN000;IFS.fake fail
36341 0000613B EB27                    JMP     short passi24          ; AN000;IFS.exit
36342                                  i24yes:                        ; AN000;
36343                                  ;; Extended Open hooks
36344
36345 0000613D 368E16[8605]            MOV     SS,[SS:USER_SS]
36346 00006142 268B26[8405]            MOV     SP,[ES:USER_SP]        ; User stack pointer restored
36347
36348                                  ;;int  24h
36349                                  ;IN     int_fatal_abort                ; Fatal error interrupt vector,
36350                                                  ; must preserve ES
36351                                  ; 05/05/2019
36352 00006147 26803E[660D]00          cmp     byte [es:DosHasHMA],0 ; Q: is dos running in HMA (M021)
36353 0000614D 7504                    jne     short do_low_int24    ; Y: the int must be done from low mem
36354 0000614F CD24                    INT     int_fatal_abort        ; Fatal error interrupt vector,
36355                                                  ; must preserve ES
36356 00006151 EB05                    jmp     short criterr_ret_addr
36357
36358                                  do_low_int24:
36359                                      ; 05/05/2019
36360                                      ; MSDOS 6.0
36361 00006153 2EFF1E[FB5D]              call    far [cs:LowInt24Addr]
36362                                  criterr_ret_addr:
36363 00006158 268926[8405]              MOV     [ES:USER_SP],SP       ; restore our stack
36364 0000615D 268C16[8605]              MOV     [ES:USER_SS],SS
36365                                    ;MOV    BP,ES
36366                                    ;MOV    SS,BP
36367                                    ; 30/06/2024
36368 00006162 06                        push    es
36369 00006163 17                        pop     ss
36370                                  passi24:
36371 00006164 368B26[8805]              MOV     SP,[SS:CONTSTK]
36372 00006169 36FE06[2103]              INC     BYTE [SS:INDOS]               ; Back in the DOS
36373
36374                                    ; 29/02/2024 - Retro DOS v5.0 (PCDOS 7.1 IBMDOS.COM)
36375                                    ;;;
36376 0000616E 36FE06[B812]              inc     byte [ss:INDOS_FLAG]
36377                                    ;;;
36378
36379 00006173 36C606[2003]00            MOV     BYTE [SS:ERRORMODE],0 ; Back from INT 24
36380 00006179 FB                        STI
36381                                  FailRet:
36382 0000617A 36C42E[8005]              LES     BP,[SS:EXITHOLD]
36383
36384                                    ; 08/07/2018
36385
36386                                    ; Triage the user's reply.
36387
36388 0000617F 3C01                      CMP     AL,1
36389 00006181 723D                      JB      short CheckIgnore     ; 0 => ignore
36390 00006183 7445                      JZ      short CheckRetry      ; 1 => retry
36391 00006185 3C03                      CMP     AL,3                  ; 3 => fail
36392 00006187 7549                      JNZ     short DoAbort         ; 2, invalid => abort
36393
36394                                    ; The reply was fail. See if we are allowed to fail.
36395
36396                                                  ; SS override for ALLOWED, EXTOPEN_ON,
36397                                                  ; ALLOWED, FAILERR, WPERR, SFN, pJFN
36398                                    ;test  byte [ss:ALLOWED],8
36399 00006189 36F606[4B03]08            test    byte [ss:ALLOWED],Allowed_FAIL ; Can we?
36400 0000618F 7441                      jz      short DoAbort         ; No, do abort
36401                                  DoFail:
36402 00006191 B003                      MOV     AL,3                  ; just in case...
36403                                                  ; AN000;EO. I24 error disabled
36404                                    ; 05/05/2019
36405                                    ;(MSDOS 6.0, MSCTRLC.ASM, 1991)
36406 00006193 36F606[F605]02            test    byte [ss:EXTOPEN_ON],EXT_OPEN_I24_OFF ; 2
36407 00006199 7505                      jnz     short CleanUp         ; AN000;EO. no
36408
36409 0000619B 36FE06[4A03]              inc     byte [SS:FAILERR]     ; Tell everybody
36410                                  CleanUp:
36411 000061A0 36C606[2203]FF            MOV     byte [SS:WPERR],-1
36412 000061A6 36833E[AA05]FF            CMP     word [SS:SFN],-1
36413                                    ; 25/11/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
36414                                    ;jnz    short CleanUp2
36415                                    ;retn
36416                                    ; 17/12/2022
36417 000061AC 7411                      jz      short Cleanup_retn ; 08/07/2018 - Retro DOS v3.0
36418                                  CleanUp2:
36419 000061AE 1E                        push    ds
36420 000061AF 56                        push    si
36421 000061B0 50                        push    ax
36422 000061B1 36A1[AA05]                MOV     AX,[ss:SFN]
36423 000061B5 36C536[AE05]              LDS     SI,[ss:PJFN]
36424 000061BA 8804                      MOV     [SI],AL
36425 000061BC 58                        pop     ax
36426 000061BD 5E                        pop     si
36427 000061BE 1F                        pop     ds
36428                                  Cleanup_retn:
36429 000061BF C3                        retn
36430
36431                                    ; The reply was IGNORE. See if we are allowed to ignore.
36432
36433                                  CheckIgnore:
36434                                    ;test  byte [ss:ALLOWED],20h
36435 000061C0 36F606[4B03]20            test    byte [ss:ALLOWED],Allowed_IGNORE ; Can we?
36436                                  CheckRI:    ; 29/02/2024
36437 000061C6 74C9                      jz      short DoFail                  ; No, do fail
36438 000061C8 EBD6                      jmp     short CleanUp
36439
36440                                    ; The reply was RETRY. See if we are allowed to retry.
36441
36442                                  CheckRetry:
```

```
                                    ;test  byte [ss:ALLOWED],10h
000061CA 36F606[4B03]10             test   byte [ss:ALLOWED],Allowed_RETRY      ; Can we?
                                    ;jz    short DoFail                  ; No, do fail
                                    ;JMP   short CleanUp
                                    ; 29/02/2024 (PCDOS 7.1 IBMDOS.COM)
000061D0 EBF4                       jmp    short CheckRI

                                    ; The reply was ABORT.
                                DoAbort:
000061D2 16                         push   ss
000061D3 1F                         pop    ds

000061D4 803E[5703]00               CMP    byte [CONSWAP],0
000061D9 7403                       JZ     short NOSWAP2
000061DB E87FD9                     call   SWAPBACK
                                NOSWAP2:
                                    ; See if we are to truly abort. If we are in the process of aborting,
                                    ; turn this abort into a fail.

                                    ;test  [fAborting],0FFh
                                    ;jnz   short DoFail

000061DE 803E[5903]00               cmp    byte [fAborting],0
000061E3 75AC                       JNZ    short DoFail

                                    ; Set return code

000061E5 C606[7C05]02               MOV    BYTE [EXIT_TYPE],EXIT_HARD_ERROR ; 2
000061EA 30C0                       XOR    AL,AL

                                    ; we are truly aborting the process. Go restore information from
                                    ; the PDB as necessary.

000061EC E97510                     jmp    exit_inner

                                ;** -----------------------------------------------------------------------
                                ;
                                ; reset_environment checks the DS value against the CurrentPDB. If they are
                                ; different, then an old-style return is performed. If they are the same,
                                ; then we release jfns and restore to parent. We still use the PDB at DS:0 as
                                ; the source of the terminate addresses.
                                ;
                                ; Some subtlety: We are about to issue a bunch of calls that *may* generate
                                ; INT 24s. We *cannot* allow the user to restart the abort process; we may
                                ; end up aborting the wrong process or turn a terminate/stay/resident into a
                                ; normal abort and leave interrupt handlers around. What we do is to set a
                                ; flag that will indicate that if any abort code is seen, we just continue the
                                ; operation. In essence, we dis-allow the abort response.
                                ;
                                ; output:   none.
                                ; -----------------------------------------------------------------------

                                    ;entry  reset_environment

                                reset_environment:
                                    ; 30/07/2018 - Retro DOS v3.0
                                    ; IBMDOS.COM (MSDOS 3.3) - Offset 588Ah

                                ;***invoke  Reset_Version         ; AN007 ;MS. reset version number

000061EF 1E                         PUSH   DS                     ; save PDB of process

                                    ; There are no critical sections in force. Although we may enter
                                    ; here with critical sections locked down, they are no longer
                                    ; relevant. We may safely free all allocated resources.

000061F0 B482                       MOV    AH,82h
                                           ; Microsoft Networks - END DOS CRITICAL SECTIONS 0 THROUGH 7
                                    ;int   2Ah
000061F2 CD2A                       INT    int_IBM

                                                                  ; SS override
000061F4 36C606[5903]FF             MOV    byte [SS:fAborting],-1; signal abort in progress

                                                                  ; DOS 4.00 doesn't need it
                                    ;CallInstall NetResetEnvironment, MultNET, 34
                                                                  ; Allow REDIR to clear some stuff
                                                                  ; On process exit.
000061FA B82211                     mov    ax, 1122h
000061FD CD2F                       int    2Fh    ; Multiplex - NETWORK REDIRECTOR - PROCESS TERMINATION HOOK
                                           ; SS = DOS CS
                                    ;mov   al,22h
000061FF B022                       MOV    AL,int_terminate
00006201 E865AD                     call   _$GET_INTERRUPT_VECTOR; and who to go to

00006204 59                         POP    CX                     ; get ThisPDB
00006205 06                         push   es
00006206 53                         push   bx                     ; save return address

00006207 368B1E[3003]               MOV    BX,[SS:CurrentPDB]     ; get currentPDB
0000620C 8EDB                       MOV    DS,BX
0000620E A11600                     MOV    AX,[PDB.PARENT_PID]    ; get parentPDB

                                    ; AX = parentPDB, BX = CurrentPDB, CX = ThisPDB
                                    ; Only free handles if AX <> BX and BX = CX and [exit_code].upper
                                    ; is not Exit_keep_process

00006211 39D8                       CMP    AX,BX
00006213 7418                       JZ     short reset_return     ; parentPDB = CurrentPDB
00006215 39CB                       CMP    BX,CX
00006217 7514                       JNZ    short reset_return     ; CurrentPDB <> ThisPDB
00006219 50                         PUSH   AX                     ; save parent

                                                                  ; SS override
                                    ;cmp   byte [SS:EXIT_TYPE],3
0000621A 3680[7C05]03               CMP    BYTE [SS:EXIT_TYPE],EXIT_KEEP_PROCESS ; 15/08/2018
00006220 7406                       JZ     short reset_to_parent ; keeping this process

                                    ; We are truly removing a process. Free all allocation blocks
                                    ; belonging to this PDB

                                    ;invoke arena_free_process
00006222 E87E10                     call   arena_free_process

                                    ; Kill off remainder of this process. Close file handles and signal
                                    ; to relevant network folks that this process is dead. Remember that
                                    ; CurrentPDB is STILL the current process!

                                    ;invoke DOS_ABORT
00006225 E888D4                     call   DOS_ABORT

                                reset_to_parent:
                                                                  ; SS override
00006228 368F06[3003]               POP    word [SS:CurrentPDB]  ; set up process as parent
```

```
                                reset_return:                           ; come here for normal return
                                    ;Context DS                 ; DS is used to refer to DOSDATA
0000622D 16                         push    ss
0000622E 1F                         pop     ds

0000622F B0FF                       MOV     AL,-1

                                    ; make sure that everything is clean In this case ignore any errors,
                                    ; we cannot "FAIL" the abort, the program being aborted is dead.

                                    ;EnterCrit critDisk
00006231 E8AEB6                     call    ECritDisk
                                    ;invoke FLUSHBUF
00006234 E86108                     call    FLUSHBUF
                                    ;LeaveCrit critDisk
00006237 E8D5B6                     call    LCritDisk

                                ; 29/02/2024 - Retrto DOS v5.0
                                ; PCDOS 7.1 IBMDOS.COM
                                %if 0
                                    ; Decrement open ref. count if we had done a virtual open earlier.

                                    call    CHECK_VIRT_OPEN
                                %endif

0000623A FA                         CLI
0000623B C606[2103]00               MOV     BYTE [INDOS],0          ; Go to known state

                                    ; 29/02/2024 - Retro DOS v5.0 (PCDOS 7.1 IBMDOS.COM)
                                    ;;;
00006240 C606[B812]00               mov     byte [INDOS_FLAG],0
                                    ;;;

00006245 C606[2203]FF               MOV     BYTE [WPERR],-1             ; Forget about WP error
0000624A C606[5903]00               MOV     byte [fAborting],0      ; let aborts occur
0000624F 8F06[8005]                 POP     WORD [EXITHOLD]
00006253 8F06[8205]                 POP     WORD [EXITHOLD+2]

                                    ; Snake into multitasking... Get stack from CurrentPDB person

00006257 8E1E[3003]                 MOV     DS,[CurrentPDB]
0000625B 8E163000                   MOV     SS,[PDB.USER_STACK+2]
0000625F 8B262E00                   MOV     SP,[PDB.USER_STACK]

00006263 E8DBA1                     call    restore_world

                                    ; 05/05/2019
00006266 07                         pop     es ; * ; MSDOS 6.21 (DOSCODE:94A8h, MSDOS.SYS)

                                    ; MSDOS 6.0
00006267 50                         push    ax                      ; set up ds, but save ds in TEMPSEG
00006268 8CD8                       mov     ax,ds                   ; and not on stack.
                                    ;getdseg <ds>                   ; ds -> dosdata
0000626A 2E8E1E[0700]               mov     ds,[cs:DosDSeg]
0000626F A3[630D]                   mov     [TEMPSEG],ax
00006272 58                         pop     ax
                                                                    ; set up ds to DOSDATA
                                    ;MOV    [CS:USER_SP],AX ; MSDOS 3.3
00006273 A3[8405]                   mov     [USER_SP],ax

00006276 58                         POP     AX                      ; suck off CS:IP of interrupt...
00006277 58                         POP     AX
00006278 58                         POP     AX

                                ; M011 : BEGIN

                                    ; MSDOS 3.3
                                ;   MOV     AX,0F202h       ; STI

                                    ; MSDOS 6.0
00006279 9F                         LAHF
0000627A 86E0                       XCHG    AH,AL
0000627C 2402                       AND     AL,2
0000627E B4F2                       MOV     AH,0F2h

                                ; M011 : END

                                    ; MSDOS 3.3 (& MSDOS 6.0)
00006280 50                         PUSH    AX

                                    ;PUSH   word [CS:EXITHOLD+2]
                                    ;PUSH   word [CS:EXITHOLD]

                                    ; MSDOS 6.0
00006281 FF36[8205]                 PUSH    word [EXITHOLD+2]
00006285 FF36[8005]                 PUSH    word [EXITHOLD]

                                    ;MOV    AX,[CS:USER_SP]

                                    ; MSDOS 6.0
00006289 A1[8405]                   MOV     AX,[USER_SP]
0000628C 8E1E[630D]                 mov     ds,[TEMPSEG]    ; restore ds

00006290 CF                         IRET                    ; Long return back to user terminate address

                                ;--------------------------------------------------------------------------
                                ;
                                ; Procedure Name : SET_I24_EXTENDED_ERROR
                                ;
                                ; This routine handles extended error codes.
                                ; Input : DI = error code from device
                                ; Output: All EXTERR fields are set
                                ;
                                ;--------------------------------------------------------------------------

                                SET_I24_EXTENDED_ERROR:
00006291 50                         PUSH    AX
                                                                    ; ErrMap24End is in DOSDATA
00006292 B8[BB0E]                   MOV     AX,ErrMap24End
00006295 2D[AB0E]                   SUB     AX,ErrMap24
                                                                    ; Change to dosdata to access
                                                                    ; ErrMap24 and EXTERR -SR
                                    ; 05/05/2019 - Retro DOS v4.0

                                    ; MSDOS 6.0
00006298 1E                         push    ds
                                    ;getdseg <ds>                   ; ds ->dosdata
00006299 2E8E1E[0700]               mov     ds,[cs:DosDSeg]

                                    ; AX is the index of the first unavailable error. Do not translate
                                    ; if greater or equal to AX.

0000629E 39C7                       CMP     DI,AX
```

```
36691 000062A0 89F8                       MOV     AX,DI
36692 000062A2 7306                       JAE     short NoTrans
36693
36694                                     ;MOV    AL,[CS:DI+ErrMap24]  ; MSDOS 3.3
36695 000062A4 8A85[AB0E]                 mov     al,[ErrMap24+di] ; MSDOS 6.0
36696 000062A8 30E4                       XOR     AH,AH
36697                                   NoTrans:
36698                                     ;MOV    [CS:EXTERR],AX
36699 000062AA A3[2403]                   mov     [EXTERR],AX
36700 000062AD 1F                         pop     ds
36701                                     ;assume ds:nothing
36702 000062AE 58                         POP     AX
36703
36704                                     ; Now Extended error is set correctly. Translate it to get correct
36705                                     ; error locus class and recommended action.
36706
36707 000062AF 56                         PUSH    SI
36708                                                             ; ERR_TABLE_24 is in DOSCODE
36709 000062B0 BE[5B0E]                   MOV     SI,ERR_TABLE_24
36710 000062B3 E8FBA3                     call    CAL_LK                ; Set other extended error fields
36711 000062B6 5E                         POP     SI
36712 000062B7 C3                         retn
36713
36714                                   ;=============================================================================
36715                                   ; FAT.ASM, MSDOS 6.0, 1991
36716                                   ;=============================================================================
36717                                   ; 30/07/2018 - Retro DOS v3.0
36718                                   ; 20/05/2019 - Retro DOS v4.0
36719                                   ; 02/03/2024 - Retro DOS v5.0
36720
36721                                   ;   TITLE   FAT - FAT maintenance routines
36722                                   ;   NAME    FAT
36723
36724                                   ;** FAT.ASM
36725                                   ;-----------------------------------------------------------------------------
36726                                   ;   Low level local device routines for performing disk change sequence,
36727                                   ;   setting cluster validity, and manipulating the FAT
36728                                   ;
36729                                   ;   IsEof
36730                                   ;   UNPACK
36731                                   ;   PACK
36732                                   ;   MAPCLUSTER
36733                                   ;   FATREAD_SFT
36734                                   ;   FATREAD_CDS
36735                                   ;   FAT_operation
36736                                   ;
36737                                   ;   Revision history:
36738                                   ;
36739                                   ;     AN000  version Jan. 1988
36740                                   ;      A001  PTM         -- disk changed for look ahead buffers
36741                                   ;
36742                                   ;   M014 - if a request for pack\unpack cluster 0 is made we write\read
36743                                   ;          from CL0FATENTRY rather than disk.
36744
36745                                   ; DOSCODE:94FAh (MSDOS 6.21, MSDOS.SYS)
36746
36747                                   ; 02/03/2024 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMDOS.COM)
36748                                   ; PCDOS 7.1 IBMDOS.COM - DOSCODE:A40Ch
36749
36750                                   ;Break <IsEOF - check the quantity in BX for EOF>
36751                                   ;-----------------------------------------------------------------------------
36752                                   ;
36753                                   ; Procedure Name : IsEOF
36754                                   ;
36755                                   ; IsEOF - check the fat value in BX for eof.
36756                                   ;
36757                                   ;   Inputs: ES:BP point to DPB
36758                                   ;           BX has fat value
36759                                   ;   Outputs:       JAE eof
36760                                   ;   Registers modified: none
36761                                   ;
36762                                   ;-----------------------------------------------------------------------------
36763
36764                                   IsEOF:
36765                                       ; 02/03/2024 - Retro DOS v5.0
36766                                       ; PCDOS 7.1 IBMDOS.COM
36767                                       ;;;
36768 000062B8 E89801                     call    IsFAT32               ; is it FAT32 drive ?
36769 000062BB 730D                       jnc     short IsEOF_FAT               ; no, it has 12 or 16 bit FAT
36770
36771                                       ; FAT32 fs
36772 000062BD 36813E[E80A]FF0F           cmp     word [ss:CLUSTNUM_HW],0FFFh
36773 000062C4 7503                       jnz      short IsEOF_other1    ; not EOF
36774 000062C6 83FBF8                     cmp      bx,0FFF8h             ; 32 bit compare
36775                                   IsEOF_other1:
36776 000062C9 C3                         retn                           ; cf=0 -> EOF, cf=1 -> not EOF
36777
36778                                   IsEOF_FAT:
36779                                       ;;;
36780
36781                                       ;cmp    word [es:bp+0Dh],0FF6h
36782 000062CA 26817E0DF60F               CMP     word [ES:BP+DPB.MAX_CLUSTER],4096-10 ; is this 16 bit fat?
36783 000062D0 730B                       JAE     short EOF16                   ; yes, check for eof there
36784
36785                                   ;J.K. 8/27/86
36786                                   ;Modified to accept 0FF0h as an eof. This is to handle the diskfull case
36787                                   ;of any media that has "F0"(Other) as a MediaByte.
36788                                   ;Hopely, this does not create any side effect for those who may use any value
36789                                   ;other than "FF8-FFF" as an EOF for their own file.
36790
36791 000062D2 81FBF00F                     cmp     bx,0FF0h
36792 000062D6 7404                       je      short IsEOF_other
36793
36794 000062D8 81FBF80F                   CMP     BX,0FF8h              ; do the 12 bit compare
36795                                   IsEOF_other:
36796 000062DC C3                         retn
36797                                   EOF16:
36798 000062DD 83FBF8                     CMP     BX,0FFF8h             ; 16 bit compare
36799 000062E0 C3                         retn                          ; cf=0 -> EOF, cf=1 -> not EOF
36800
36801                                   ; DOSCODE:9511h (MSDOS 6.21, MSDOS.SYS)
36802
36803                                   ;Break      <UNPACK -- UNPACK FAT ENTRIES>
36804                                   ;-----------------------------------------------------------------------------
36805                                   ;
36806                                   ; Procedure Name : UNPACK
36807                                   ;
36808                                   ; Inputs:
36809                                   ;   BX = Cluster number (may be full 16-bit quantity)
36810                                   ;   ES:BP = Base of drive parameters
36811                                   ; Outputs:
36812                                   ;   DI = Contents of FAT for given cluster (may be full 16-bit quantity)
36813                                   ;   Zero set means DI=0 (free cluster)
36814                                   ;   Carry set means error (currently user FAILed to I 24)
```

```
                                ; SI Destroyed, No other registers affected. Fatal error if cluster too big.
                                ;
                                ; NOTE: if BX = 0 then DI = contents of CL0FATENTRY
                                ;
                                ;--------------------------------------------------------------------------

                                    ; 02/03/2024 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMDOS.COM)
                                    ; PCDOS 7.1 IBMDOS.COM - DOSCODE:0A44Eh
                                    ;
                                    ; (MSDOS 6.22 MSDOS .SYS - DOSCODE:9511h)
                                    ; (Windows ME IO.SYS - BIOSCODE:0C368h)

                                    ; 25/11/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
                                    ; DOSCODE:94B5h (MSDOS 5.0, MSDOS.SYS)

                                    ; 20/05/2019 - Retro DOS v4.0
                                UNPACK:
                                    ; MSDOS 6.0                   ; M014 - Start
000062E1 09DB                       or      bx,bx                 ; Q: are we unpacking cluster 0
000062E3 7519                       jnz     short up_cont         ; N: proceed with normal unpack

                                ; 02/03/2024
                                %if 0
                                    mov     di,[CL0FATENTRY]      ; Y: return value in CL0FATENTRY
                                    or      di,di                 ; return z if di=0
                                    retn                          ; done
                                %else
                                    ; PCDOS 7.1 IBMDOS.COM - DOSCODE:A435h
                                    ;;;
                                up_1:
000062E5 391E[E80A]                 cmp     [CLUSTNUM_HW],bx ; 0
000062E9 7513                       jne     short up_cont
000062EB 8B3E[3B0F]                 mov     di,[CL0FATENTRY_HW]
000062EF 09FF                       or      di,di
000062F1 893E[EC0A]                 mov     [CCONTENT_HW],di
000062F5 8B3E[8100]                 mov     di,[CL0FATENTRY]
000062F9 7502                       jnz     short unpack_retn
000062FB 09FF                       or      di,di
                                unpack_retn:
000062FD C3                         retn    ; [CCONTENT_HW]:DI = contents of CL0FATENTRY
                                    ;;;

                                %endif

                                up_cont:                          ; M014 - End
                                    ; 02/03/2024 (PCDOS 7.1 IBMDOS.COM)
                                    ;;;
                                    ;cmp    word [es:bp+0Fh],0
000062FE 26837E0F00                 cmp     word [es:bp+DPB.FAT_SIZE],0
00006303 7510                       jnz     short up_fat          ; not FAT32
                                    ; FAT32
                                    ;mov    si,[es:bp+2Fh]
00006305 268B762F                   mov     si,[es:bp+DPB.LAST_CLUSTER+2]
00006309 3936[E80A]                 cmp     [CLUSTNUM_HW],si
0000630D 750A                       jne     short up_2
                                    ;cmp    bx,[es:bp+2Dh]
0000630F 263B5E2D                   cmp     bx,[es:bp+DPB.LAST_CLUSTER]
00006313 EB04                       jmp     short up_2
                                up_fat:
                                    ;;;

                                    ; MSDOS 3.3 & MSDOS 6.0
                                    ;cmp    bx,[es:bp+0Dh]
00006315 263B5E0D                   CMP     BX,[es:bp+DPB.MAX_CLUSTER]
                                up_2:                 ; 02/03/2024
00006319 7748                       JA      short HURTFAT
0000631B E84A01                     CALL    MAPCLUSTER
0000631E 7240                       jc      short _DoContext

                                    ; 02/03/2024 (PCDOS 7.1 IBMDOS.COM)
                                    ;;;
00006320 FF7502                     push    word [di+2]           ; offset CLUSSAVE+2
00006323 368F06[EC0A]               pop     word [ss:CCONTENT_HW] ; high word of cluster number
                                    ;;;

00006328 8B3D                       MOV     DI,[DI]
0000632A 7521                       JNZ     short High12          ; MZ if high 12 bits, go get 'em

                                    ; 02/03/2024 (PCDOS 7.1 IBMDOS.COM)
                                    ;;;
0000632C E82401                     call    IsFAT32
0000632F 7211                       jc      short up_fat32        ; FAT32 volume (drive)
00006331 36C706[EC0A]0000           mov     word [ss:CCONTENT_HW],0
                                    ;;;

00006338 268B760D                   MOV     SI,[ES:BP+DPB.MAX_CLUSTER] ; MZ is this 16-bit fat?
0000633C 81FEF60F                   CMP     SI,4096-10 ; 0FF6h
00006340 721A                       JB      short Unpack12        ; MZ No, go 'AND' off bits

                                ; 02/03/2024
                                %if 0
                                    OR      DI,DI                 ; MZ set zero condition code, clears carry
                                    JMP     SHORT _DoContext      ; MZ go do context
                                High12:
                                    SHR     DI,1
                                    SHR     DI,1
                                    SHR     DI,1
                                    SHR     DI,1
                                %else
                                    ; 02/03/2024 (PCDOS 7.1 IBMDOS.COM)
                                    ;;;
                                up_fat32:
00006342 09FF                       or      di,di                 ; set zero condition code, clears carry
                                    ;mov    si,ss
                                    ;mov    ds,si
00006344 16                         push    ss
00006345 1F                         pop     ds
00006346 7504                       jnz     short up_retn
00006348 093E[EC0A]                 or      [CCONTENT_HW],di      ; [CCONTENT_HW]:DI = 0 -> zf = 1
                                up_retn:
0000634C C3                         retn

                                High12:
0000634D 36C706[EC0A]0000           mov     word [ss:CCONTENT_HW],0
00006354 D1EF                       shr     di,1
00006356 D1EF                       shr     di,1
00006358 D1EF                       shr     di,1
0000635A D1EF                       shr     di,1
                                    ;;;
                                %endif

                                Unpack12:
0000635C 81E7FF0F                   AND     DI,0FFFh              ; Clears carry
                                _DoContext:
```

```
36939 00006360 16                          PUSH    SS
36940 00006361 1F                          POP     DS
36941 00006362 C3                          retn
36942
36943                                  HURTFAT:
36944
36945                                  ; 02/03/2024
36946                                  %if 0
36947                                      ;;mov   word [es:bp+1Eh],0FFFFh
36948                                      ;mov    word [es:bp+1Fh],0FFFFh  ; MSDOS 6.0
36949                                      MOV     word [ES:BP+DPB.FREE_CNT],-1 ; Err in FAT must force recomp of freespace
36950                                  %else
36951                                      ; 02/03/2024 (PCDOS 7.1 IBMDOS.COM)
36952                                      ;;;
36953 00006363 E84C02                    call    chk_set_first_access
36954                                      ;;;
36955                                  %endif
36956
36957 00006366 50                          PUSH    AX
36958 00006367 B488                        MOV     AH,Allowed_FAIL+80h ; 88h
36959
36960                                  ; 02/03/2024
36961                                  %if 0
36962
36963                                  ;hkn; SS override
36964                                      MOV     byte [SS:ALLOWED],Allowed_FAIL ; 8
36965                                  ;
36966                                  ; Signal Bad FAT to INT int_fatal_abort handler. We have an invalid cluster.
36967                                  ;
36968                                      MOV     DI,0FFFh                ; In case INT int_fatal_abort returns (it shouldn't)
36969                                      call    FATAL
36970                                  %else
36971                                      ; 02/03/2024 (PCDOS 7.1 IBMDOS.COM)
36972                                      ;;;
36973                                      ;mov    byte [ss:ALLOWED],8
36974 00006369 36C606[4B03]08            mov     byte [ss:ALLOWED],Allowed_FAIL
36975                                      ;
36976                                      ; 02/03/2024 - Retro DOS v5.0
36977 0000636F 31FF                      xor     di,di
36978 00006371 4F                        dec     di      ; 0FFFFh
36979 00006372 57                        push    di
36980                                              ; NOTE: DI would be 0FFFFh here (or 0FFFh as in MSDOS 6.?)
36981                                              ; because, in SET_I24_EXTENDED_ERROR in FATAL,
36982                                              ; DI is compared with error code
36983                                              ; ;
36984                                              ; xor di,di
36985                                              ; dec di ; 0FFFFh
36986                                              ; push di
36987                                              ; call FATAL
36988                                              ; pop di
36989                                              ; mov word [ss:CCONTENT_HW], di
36990                                              ; ;
36991                                              ; Erdogan Tan - 02/03/2024 - 01/07/2024
36992                                      ;
36993 00006373 36FF36[E80A]              push    word [ss:CLUSTNUM_HW] ; (necessary!?)
36994 00006378 53                        push    bx      ; (necessary!?)
36995 00006379 E85FFD                    call    FATAL
36996 0000637C 5B                        pop     bx
36997 0000637D 368F06[E80A]              pop     word [ss:CLUSTNUM_HW]
36998                                    ;xor    di,di
36999                                    ;dec    di      ; 0FFFFh
37000                                    ;mov    word [ss:CCONTENT_HW],di
37001                                    ; 02/03/2024 - Retro DOS v5.0
37002                                    ;pop    word [ss:CCONTENT_HW]
37003                                    ; 01/07/2024
37004 00006382 5F                        pop     di
37005 00006383 36893E[EC0A]              mov     [ss:CCONTENT_HW], di
37006                                    ;;;
37007                                  %endif
37008 00006388 3C03                      CMP     AL,3
37009 0000638A F8                        CLC
37010 0000638B 7501                      JNZ     short OKU_RET           ; Try to ignore bad FAT
37011 0000638D F9                        STC                             ; User said FAIL
37012                                  OKU_RET:
37013 0000638E 58                        POP     AX
37014                                  hurtfat_retn:
37015 0000638F C3                        retn
37016
37017                                  ; DOSCODE:9565h (MSDOS 6.21, MSDOS.SYS)
37018
37019                                  ;Break     <PACK -- PACK FAT ENTRIES>
37020                                  ;--------------------------------------------------------------------------
37021                                  ;
37022                                  ; Procedure Name : PACK
37023                                  ;
37024                                  ; Inputs:
37025                                  ;   BX = Cluster number
37026                                  ;   DX = Data
37027                                  ;   ES:BP = Pointer to drive DPB
37028                                  ; Outputs:
37029                                  ;   The data is stored in the FAT at the given cluster.
37030                                  ;   SI,DX,DI all destroyed
37031                                  ;   Carry set means error (currently user FAILed to I 24)
37032                                  ;   No other registers affected
37033                                  ;
37034                                  ; NOTE: if BX = 0 then data in DX is stored in CLUSFATENTRY.
37035                                  ;
37036                                  ;--------------------------------------------------------------------------
37037
37038                                      ; 02/03/2024 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMDOS.COM)
37039
37040                                      ; INPUT:
37041                                      ;  [CLUSDATA_HW]:DX = cluster data/content
37042                                      ;  [CLUSTNUM_HW]:BX = cluster number (to be updated)
37043                                      ;  ES:BP = pointer to DPB
37044
37045                                      ; 25/11/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
37046                                      ; 20/05/2019 - Retro DOS v4.0
37047                                  PACK:
37048                                      ; MSDOS 6.0                     ; M014 - start
37049 00006390 09DB                      or      bx,bx                   ; Q: are we packing cluster 0
37050 00006392 7513                      jnz     short p_cont            ; N: proceed with normal pack
37051
37052                                  ; 02/03/2024
37053                                  %if 0
37054                                      mov     [CLUSFATENTRY],dx       ; Y: place value in CLUSFATENTRY
37055                                      retn                            ; done
37056                                  %else
37057                                      ; 02/03/2024 (PCDOS 7.1 IBMDOS.COM)
37058                                      ;;;
37059 00006394 391E[E80A]                cmp     [CLUSTNUM_HW],bx ; 0
37060 00006398 750D                      jnz     short p_cont
37061
37062                                      ;push   ax
```

```
                                    ;mov    ax,[CLUSDATA_HW]      ; Cluster Data/Content (High Word)
                                    ;mov    [CL0FATENTRY_HW],ax
                                    ;pop    ax
0000639A FF36[EA0A]                 push    word [CLUSDATA_HW]
0000639E 8F06[3B0F]                 pop     word [CL0FATENTRY_HW]

000063A2 8916[8100]                 mov     [CL0FATENTRY],dx
000063A6 C3                         retn
                                    ;;;
                                %endif

                                p_cont:                                     ; M014 - end
                                    ; MSDOS 3.3 & MSDOS 6.0
000063A7 E8BE00                     CALL    MAPCLUSTER
000063AA 72B4                       JC      short _DoContext
000063AC 8B35                       MOV     SI,[DI]
000063AE 740B                       JZ      short ALIGNED         ; byte (not nibble) aligned
000063B0 51                         PUSH    CX                    ; move data to upper 12 bits
000063B1 B104                       MOV     CL,4
000063B3 D3E2                       SHL     DX,CL
000063B5 59                         POP     CX
000063B6 83E60F                     AND     SI,0FH                ; leave in original low 4 bits
000063B9 EB2B                       JMP     SHORT PACKIN

                                ALIGNED:

                                    ; 02/03/2024 (PCDOS 7.1 IBMDOS.COM)
                                    ;;;
000063BB E89500                     call    IsFAT32
000063BE 7313                       jnb     short p_fat

000063C0 368B36[EA0A]               mov     si,[ss:CLUSDATA_HW]
000063C5 337502                     xor     si,[di+2]             ; hw of cluster number
                                                                  ; offset CLUSSAVE+2
000063C8 81E6FF0F                   and     si,0FFFh
000063CC 317502                     xor     [di+2],si
000063CF 8915                       mov     [di],dx
000063D1 EB17                       jmp     short PACKIN2
                                p_fat:
                                    ;;;

                                    ;cmp    word [es:bp+0Dh],0FF6h
000063D3 26817E0DF60F               CMP     word [ES:BP+DPB.MAX_CLUSTER],4096-10 ; MZ 16 bit fats?
000063D9 7309                       JAE     short Pack16          ; MZ yes, go clobber original data
000063DB 81E600F0                   AND     SI,0F000h             ; MZ leave in upper 4 bits of original
                                    ;AND    DX,0FFFh              ; MZ store only 12 bits
                                    ; 01/07/2024
000063DF 80E60F                     and     dh,0Fh
000063E2 EB02                       JMP     SHORT PACKIN          ; MZ go store
                                Pack16:
000063E4 31F6                       XOR     SI,SI                 ; MZ no original data
                                PACKIN:
000063E6 09D6                       OR      SI,DX
000063E8 8935                       MOV     [DI],SI

                                PACKIN2:    ; 02/03/2024

                                ;hkn; SS override
000063EA 36C536[E205]               LDS     SI,[SS:CURBUF]
                                    ; MSDOS 6.0
000063EF F6440540                   TEST    byte [SI+BUFFINFO.buf_flags],buf_dirty
                                                                  ;LB. if already dirty          ;AN000;
000063F3 7507                       JNZ     short yesdirty11      ;LB.  don't increment dirty count ;AN000;
                                    ; 10/06/2019
000063F5 E8D207                     call    INC_DIRTY_COUNT             ;LB.                          ;AN000;

                                    ;or     byte [si+5],40h
000063F8 804C0540                   OR      byte [SI+BUFFINFO.buf_flags],buf_dirty
                                yesdirty11:                       ;LB.                          ;AN000;
                                ;hkn; SS override
000063FC 36803E[7805]00             CMP     BYTE [SS:CLUSSPLIT],0 ; 15/08/2018
                                ;hkn; SS is DOSDATA
00006402 16                         push    ss
00006403 1F                         pop     ds
00006404 7489                       jz      short hurtfat_retn    ; Carry clear
00006406 50                         PUSH    AX
                                    ; 02/03/2024 (PCDOS 7.1 IBMDOS.COM)
                                    ;;;
00006407 FF36[E80A]                 push    word [CLUSTNUM_HW]
                                    ;;;
0000640B 53                         PUSH    BX
0000640C 51                         PUSH    CX
0000640D A1[8E05]                   MOV     AX,[CLUSSAVE]
00006410 8E1E[E405]                 MOV     DS,[CURBUF+2]
                                    ;;;add  si,16 ; MSDOS 3.3
                                    ;;add   si,20 ; MSDOS 6.0
                                    ; 02/01/2024
                                    ;add    si,24 ; PCDOS 7.1
00006414 83C618                     ADD     SI,BUFINSIZ
00006417 8824                       MOV     [SI],AH
                                ;hkn; SS is DOSDATA
                                    ;Context DS
00006419 16                         push    ss
0000641A 1F                         pop     ds

0000641B 50                         PUSH    AX

                                    ; MSDOS 6.0
0000641C 8B16[9205]                 MOV     DX,[CLUSSEC+2]        ;F.C. >32mb                   ;AN000;
00006420 8916[0706]                 MOV     [HIGH_SECTOR],DX      ;F.C. >32mb                   ;AN000;

                                    ; MSDOS 3.3 & MSDOS 6.0
00006424 8B16[9005]                 MOV     DX,[CLUSSEC]

                                    ;MOV    SI,1      ; *
                                    ;XOR    AL,AL     ; *
                                    ;call   GETBUFFRB ; *
                                    ; 22/09/2023
00006428 E8F704                     call    GETBUFFRA ; *

0000642B 58                         POP     AX
0000642C 721B                       JC      short POPP_RET
0000642E C53E[E205]                 LDS     DI,[CURBUF]

                                    ; MSDOS 6.0
00006432 F6450540                   TEST    byte [DI+BUFFINFO.buf_flags],buf_dirty
                                                                  ;LB. if already dirty          ;AN000;
00006436 7507                       JNZ     short yesdirty12      ;LB.  don't increment dirty count ;AN000;
00006438 E88F07                     call    INC_DIRTY_COUNT       ;LB.                          ;AN000;

                                    ;or     byte [di+5],40h
0000643B 804D0540                   OR      byte [DI+BUFFINFO.buf_flags],buf_dirty
                                yesdirty12:
                                    ;;;add  di,16
```

```
                                    ;;add  di,20 ; MSDOS 6.0
                                    ;ADD   DI,BUFINSIZ
                                    ;DEC   DI
                                    ; 02/01/2024 - Retro DOS v5.0
                                    ;add   di,23 ; PCDOS 7.1
0000643F 83C717                     add    di,BUFINSIZ-1 ; 23 ; PCDOS 7.1 IBMDOS.COM

                                    ;add   di,[es:bp+2]
00006442 26037E02                   ADD    DI,[ES:BP+DPB.SECTOR_SIZE]
00006446 8805                       MOV    [DI],AL
00006448 F8                         CLC
                                POPP_RET:
                                    ; 02/03/2024
00006449 16                         PUSH   SS
0000644A 1F                         POP    DS
0000644B 59                         POP    CX
0000644C 5B                         POP    BX
                                    ; 02/03/2024 (PCDOS 7.1 IBMDOS.COM)
                                    ;;;
                                    ;pop   word [ss:CLUSTNUM_HW]
                                    ; 01/07/2024
                                    ; ds = ss
0000644D 8F06[E80A]                 pop    word [CLUSTNUM_HW]
                                    ;;;
00006451 58                         POP    AX
00006452 C3                         retn

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

                                    ; 02/03/2024 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMDOS.COM)
                                    ; PCDOS 7.1 IBMDOS.COM - DOSCODE:0A5B9h
                                IsFAT32:
                                    ;cmp   word [es:bp+0Fh],1
00006453 26837E0F01                 cmp    word [es:bp+DPB.FAT_SIZE],1
00006458 730D                       jnb    short isfat32eof_2    ; not FAT32

                                    ;cmp   word [es:bp+2Fh],0
0000645A 26837E2F00                 cmp    word [es:bp+DPB.LAST_CLUSTER+2],0
0000645F 7505                       jnz    short isfat32eof_1    ; FAT32

                                    ;cmp   word [es:bp+2Dh],0FFF6h
00006461 26837E2DF6                 cmp    word [es:bp+DPB.LAST_CLUSTER],0FFF6h
                                isfat32eof_1:
00006466 F5                         cmc    ; cf=1 -> FAT32
                                isfat32eof_2:
00006467 C3                         retn

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

                                ; 31/07/2018 - Retro DOS v3.0

                                ;Break     <MAPCLUSTER - BUFFER A FAT SECTOR>
                                ;--------------------------------------------------------------------------
                                ;
                                ; Procedure Name : MAPCLUSTER
                                ;
                                ; Inputs:
                                ;   ES:BP Points to DPB
                                ;   BX Is cluster number
                                ; Function:
                                ;   Get a pointer to the cluster
                                ; Outputs:
                                ;   DS:DI Points to contents of FAT for given cluster
                                ;   DS:SI Points to start of buffer
                                ;   Zero Not set if cluster data is in high 12 bits of word
                                ;   Zero set if cluster data is in low 12 or 16 bits
                                ;   Carry set if failed.
                                ; SI is destroyed.
                                ;
                                ;--------------------------------------------------------------------------

                                    ; 20/05/2019 - Retro DOS v4.0
                                    ; DOSCODE:9601h (MSDOS 6.21, MSDOS.SYS)
                                    ; 27/11/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
                                    ; DOSCODE:95A5h (MSDOS 5.0, MSDOS.SYS)

                                    ; 02/03/2024 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMDOS.COM)
                                    ; PCDOS 7.1 IBMDOS.COM - DOSCODE:0A5D7h

                                MAPCLUSTER:
                                    ; IBMDOS.COM (MSDOS 3.3, 1987) - Offset 5A15h
00006468 C606[7805]00               MOV    BYTE [CLUSSPLIT],0
                                    ;SAVE  <AX,BX,CX,DX>
0000646D 50                         push   ax
0000646E 53                         push   bx
0000646F 51                         push   cx
00006470 52                         push   dx
                                    ; 02/03/2024
                                    ;MOV   AX,BX                 ; AX = BX

                                    ; 02/03/2024 (PCDOS 7.1 IBMDOS.COM)
                                    ;;;
                                    ;push  word [CLUSTNUM_HW]
00006471 89D8                       mov    ax,bx
00006473 8B16[E80A]                 mov    dx,[CLUSTNUM_HW]
                                    ;
00006477 52                         push   dx ; 02/03/2024 - Retro DOS v5.0
                                    ;
00006478 31FF                       xor    di,di
0000647A E8D6FF                     call   IsFAT32
0000647D 730E                       jnc    short mapcl1 ; not FAT32
                                    ; FAT32
0000647F D1E0                       shl    ax,1
00006481 D1D2                       rcl    dx,1
00006483 D1D7                       rcl    di,1
00006485 D1E0                       shl    ax,1
00006487 D1D2                       rcl    dx,1
00006489 D1D7                       rcl    di,1
                                    ; DI:DX:AX = 4*(DX:AX)
0000648B EB14                       jmp    short Map16           ; 32 bit FAT
                                mapcl1:
                                    ;;;
0000648D 26817E0DF60F               CMP    word [ES:BP+DPB.MAX_CLUSTER],4096-10  ; MZ 16 bit fat?
                                    ; 02/03/2024 - Retro DOS v5.0
                                    ;JAE   short Map16           ; MZ yes, do 16 bit algorithm
                                    ;SHR   AX,1                  ; AX = BX/2
                                    ;
                                    ; 02/03/2024 (PCDOS 7.1 IBMDOS.COM)
                                    ;;;
00006493 7304                       jae    short mapcl2          ; 16 bit FAT
                                    ; 12 bit FAT
00006495 D1EA                       shr    dx,1
00006497 D1D8                       rcr     ax,1
                                mapcl2:
```

```
37311 00006499 01D8                     add     ax,bx
37312 0000649B 1316[E80A]               adc     dx,[CLUSTNUM_HW]
37313 0000649F 11FF                     adc     di,di
37314                                   ;;;
37315                                 Map16:
37316
37317                                 ; 02/04/2024 - Retro DOS v5.0
37318                                 ; (PCDOS 7.1 IBMDOS.COM)
37319                                 %if 0
37320                                     ; MSDOS 6.0                 ; MZ skip prev => AX=2*BX
37321                                     XOR     DI,DI ; *           ; >32mb fat ;AN000;
37322
37323                                     ; MSDOS 3.3 (& MSDOS 6.0)
37324                                     ADD     AX,BX               ; AX = 1.5*fat = byte offset in fat
37325                                     ADC     DI,DI ; * MSDOS 6.0 ; >32mb fat ;DI is zero before op;AN000;
37326                                 %endif
37327
37328 000064A1 268B4E02                 MOV     CX,[ES:BP+DPB.SECTOR_SIZE]
37329
37330                                 ;IF FastDiv
37331                                 ;
37332                                 ; Gross hack: 99% of all disks have 512 bytes per sector. We test for this
37333                                 ; case and apply a really fast algorithm to get the desired results
37334                                 ;
37335                                 ; Divide method takes 157+4*4=173 (MOV and DIV)
37336                                 ; Fast method takes 39+20*4=119
37337                                 ;
37338                                 ; This saves a bunch.
37339
37340 000064A5 81F90002                   CMP     CX,512              ; 4  Is this 512 byte sector?
37341 000064A9 751C                     jne     short _DoDiv        ; 4  for no jump
37342
37343                                 ; 02/04/2024
37344                                 %if 0
37345                                     MOV     DX,AX               ; 2  get set for remainder
37346                                     AND     DX,512-1            ; 4  Form remainder
37347                                     MOV     AL,AH               ; 2  Quotient in formation in AL
37348                                     ; MDOS 3.3
37349                                     ;shr    al,1
37350                                     ; MDOS 6.0
37351                                     shr     di,1                ; 2
37352                                     rcr     al,1                ; 2
37353                                     ; MDOS 3.3 (& MSDOS 6.0)
37354                                     xor     ah,ah               ; 3
37355                                 %else
37356                                     ; 02/04/2024 - Retro DOS v5.0
37357                                     ; (PCDOS 7.1 IBMDOS.COM)
37358                                     ;;;
37359 000064AB 50                       push    ax
37360 000064AC D1EF                     shr     di,1
37361 000064AE D1DA                     rcr     dx,1
37362 000064B0 D1D8                     rcr     ax,1
37363 000064B2 88E0                     mov     al,ah               ; 9 bit shift to right
37364 000064B4 88D4                     mov     ah,dl
37365 000064B6 88F2                     mov     dl,dh
37366 000064B8 30F6                     xor     dh,dh
37367 000064BA D1EF                     shr     di,1
37368 000064BC 10F6                     adc     dh,dh
37369 000064BE 87D7                     xchg    dx,di           ; DI:AX = sector contains requested cluster data
37370 000064C0 5A                       pop     dx
37371 000064C1 81E2FF01                 and     dx,1FFh             ; offset in sector
37372                                   ;;;
37373                                 %endif
37374 000064C5 EB07                     jmp     short DivDone       ; 16
37375
37376                                 _DoDiv:
37377                                 ;ENDIF
37378
37379                                 ; 02/04/2024
37380                                 %if 0
37381                                     ; MSDOS 3.3
37382                                     ;xor    dx,dx
37383                                     ; MSDOS 6.0
37384                                     mov     dx,di               ; 2
37385                                     ; MSDOS 3.3 (& MSDOS 6.0)
37386                                     DIV     CX                  ; 155 AX is FAT sector # DX is sector index
37387                                 %else
37388                                     ; 02/04/2024 - Retro DOS v5.0
37389                                     ; (PCDOS 7.1 IBMDOS.COM)
37390                                     ;;;
37391 000064C7 97                       xchg    ax,di
37392 000064C8 92                       xchg    ax,dx
37393 000064C9 F7F1                     div     cx
37394 000064CB 97                       xchg    ax,di
37395 000064CC F7F1                     div     cx
37396                                   ;jmp    short DivDone
37397                                   ;;;
37398                                 %endif
37399
37400                                 ;IF FastDiv
37401                                 DivDone:
37402                                 ;ENDIF
37403                                     ;add    ax,[es:bp+6]
37404 000064CE 26034606                 ADD     AX,[ES:BP+DPB.FIRST_FAT]
37405
37406                                     ; 02/03/2024 (PCDOS 7.1 IBMDOS.COM)
37407                                     ;;;
37408 000064D2 83D700                   adc     di,0
37409                                   ;;;
37410
37411 000064D5 49                       DEC     CX                  ; CX is sector size - 1
37412
37413                                   ;SAVE   <AX,DX,CX>
37414
37415                                   ; 02/03/2024 (PCDOS 7.1 IBMDOS.COM)
37416                                   ;;;
37417 000064D6 57                       push    di ; 4*
37418                                   ;;;
37419 000064D7 50                       push    ax ; 3*
37420 000064D8 52                       push    dx ; 2*
37421 000064D9 51                       push    cx ; 1*
37422
37423 000064DA 89C2                     MOV     DX,AX
37424
37425                                   ; 02/03/2024 - Retro DOS v5.0
37426                                   ; mov [HIGH_SECTOR],di ; *!* ; PCDOS 7.1 IBMDOS.COM
37427 000064DC 89FB                     mov     bx,di
37428                                   ;
37429
37430                                   ; MSDOS 6.0
37431                                   ; 22/09/2023
37432                                   ;MOV    word [HIGH_SECTOR],0 ; *! ;F.C. >32mb  low sector #
37433                                   ;
37434                                   ; MDOS 3.3 (& MSDOS 6.0)
```

```
                                ;XOR    AL,AL      ; *
                                ;MOV    SI,1       ; *
                                ;;invoke GETBUFFRB ; *
                                ;call   GETBUFFRB  ; *
                                ; 22/09/2023
000064DE E83D04                 call    GETBUFFRC  ; *! ; *!* ; 02/03/2024 - Retro DOS v5.0

                                ;RESTORE <CX,AX,DX>          ; CX is sec siz-1, AX is offset in sec
000064E1 59                     pop     cx ; 1*
000064E2 58                     pop     ax ; 2*
000064E3 5A                     pop     dx ; 3*
                                ; 02/03/2024 (PCDOS 7.1 IBMDOS.COM)
                                ;;;
000064E4 5B                     pop     bx ; 4*
                                ;;;

000064E5 725A                   JC      short MAP_POP

000064E7 C536[E205]             LDS     SI,[CURBUF]
                                ;;;lea  di,[si+16]
                                ;;lea   di,[si+20] ; MSDOS 6.0
                                ;lea    di,[si+24] ; PCDOS 7.1; 02/03/2024
000064EB 8D7C18                 LEA     DI,[SI+BUFINSIZ]
000064EE 01C7                   ADD     DI,AX
000064F0 39C8                   CMP     AX,CX
000064F2 7530                   JNZ     short MAPRET   ; ds<>ss
000064F4 8A05                   MOV     AL,[DI]
                                ;Context DS                  ;hkn; SS is DOSDATA
000064F6 16                     push    ss
000064F7 1F                     pop     ds
000064F8 FE06[7805]             INC     BYTE [CLUSSPLIT]
000064FC A2[8E05]               MOV     [CLUSSAVE],AL
000064FF 8916[9005]             MOV     [CLUSSEC],DX

                                ; MSDOS 6.0
                                ;MOV    WORD [CLUSSEC+2],0      ;F.C. >32mb  ;AN000;
                                ;INC    DX
                                ; 02/03/2024 (PCDOS 7.1 IBMDOS.COM)
                                ;;;
00006503 31C0                   xor     ax,ax ; 0
00006505 83C201                 add     dx,1
00006508 891E[9205]             mov     word [CLUSSEC+2],bx
0000650C 11C3                   adc     bx,ax
                                ;mov    [HIGH_SECTOR],bx ; *!*
                                ;;;

                                ; 22/09/2023
                                ;MOV    word [HIGH_SECTOR],0 ; *! ;F.C. >32mb FAT sector <32mb ;AN000;
                                ;
                                ; MDOS 3.3 (& MSDOS 6.0)
                                ;XOR    AL,AL      ; *
                                ;MOV    SI,1       ; *
                                ;;invoke GETBUFFRB ; *
                                ;call   GETBUFFRB  ; *
                                ; 22/09/2023
0000650E E80D04                 call    GETBUFFRC  ; *! ; *!* ; 02/03/2024 - Retro DOS v5.0
00006511 722E                   JC      short MAP_POP

00006513 C536[E205]             LDS     SI,[CURBUF]
00006517 8D7C18                 LEA     DI,[SI+BUFINSIZ]
0000651A 8A05                   MOV     AL,[DI]
                                ;Context DS                  ;hkn; SS is DOSDATA
0000651C 16                     push    ss
0000651D 1F                     pop     ds
0000651E A2[8F05]               MOV     [CLUSSAVE+1],AL

                               ;hkn; CLUSSAVE is in DOSDATA
00006521 BF[8E05]                  MOV     DI,CLUSSAVE
                               MAPRET:
                                   ; 02/03/2024 (PCDOS 7.1 IBMDOS.COM)
                                   ;;;
00006524 368F06[E80A]              pop     word [ss:CLUSTNUM_HW] ; (ds<>ss)
                                   ;;;
                                   ;RESTORE <DX,CX,BX>
00006529 5A                        pop     dx
0000652A 59                        pop     cx
0000652B 5B                        pop     bx

0000652C 31C0                      XOR     AX,AX                ; MZ allow shift to clear carry

                                   ; 02/03/2024 (PCDOS 7.1 IBMDOS.COM)
                                   ;;;
0000652E E822FF                    call    IsFAT32
00006531 720A                      jc      short MapSet   ; FAT32 fs
                                   ;;;

00006533 26817E0DF60F              CMP     word [ES:BP+DPB.MAX_CLUSTER],4096-10 ; MZ is this 16-bit fat?
00006539 7302                      JAE     short MapSet          ; MZ no, set flags
0000653B 89D8                      MOV     AX,BX
                               MapSet:
0000653D A801                      TEST    AL,1                 ; set zero flag if not on boundary
                                   ;RESTORE <AX>
0000653F 58                        pop     ax
00006540 C3                        retn

                               MAP_POP:
                                   ; 02/03/2024 (PCDOS 7.1 IBMDOS.COM)
                                   ;;;
                                   ;pop    word [ss:CLUSTNUM_HW]
                                   ; 02/03/2024 - Retro DOS v5.0
00006541 8F06[E80A]                pop     word [CLUSTNUM_HW] ; (ds=ss)
                                   ;;;
                                   ;RESTORE <DX,CX,BX,AX>
00006545 5A                        pop     dx
00006546 59                        pop     cx
00006547 5B                        pop     bx
00006548 58                        pop     ax
                               fatread_sft_retn: ; 17/12/2022
00006549 C3                        retn

                               ; 20/05/2019 - Retro DOS v4.0
                               ; DOSCODE:96B3h (MSDOS 6.21, MSDOS.SYS)
                               ; 27/11/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
                               ; DOSCODE:9657h (MSDOS 5.0, MSDOS.SYS)

                               ; 03/03/2024 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMDOS.COM)
                               ; PCDOS 7.1 IBMDOS.COM - DOSCODE:0A6C6h

                               ;Break     <FATREAD_SFT/FATREAD_CDS -- CHECK DRIVE GET FAT>
                               ;--------------------------------------------------------------------------
                               ;
                               ; Procedure Name : FATREAD_SFT
                               ;
                               ; Inputs:
```

```
                                ;   ES:DI points to an SFT for the drive of interest (local only,
                                ;           giving a NET SFT will produce system crashing results).
                                ;   DS DOSDATA
                                ; Function:
                                ;   Can be used by an SFT routine (like CLOSE) to invalidate buffers
                                ;   if disk changed.
                                ;   In other respects, same as FATREAD_CDS.
                                ;   (note ES:DI destroyed!)
                                ; Outputs:
                                ;   Carry set if error (currently user FAILed to I 24)
                                ; NOTE: This routine may cause FATREAD_CDS to "miss" a disk change
                                ;   as far as invalidating curdir_ID is concerned.
                                ;   Since getting a true disk changed on this call is a screw up
                                ;   anyway, that's the way it goes.
                                ;
                                ;--------------------------------------------------------------------------

                                FATREAD_SFT:
0000654A 26C46D07                   LES     BP,[ES:DI+SF_ENTRY.sf_devptr]

                                fatread_gotdpb:    ; 03/03/2024 (PCDOS 7.1 IBMDOS.COM)
                                    ; 27/11/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
                                    ;MOV    AL,[ES:BP+DPB.DRIVE] ; [es:bp+0]
                                    ; 15/12/2022
0000654E 268A4600                   mov     AL,[ES:BP]
00006552 A2[7605]                   MOV     [THISDRV],AL
00006555 E80DA1                     call    GOTDPB                  ; Set THISDPB
                                    ;CALL   FAT_GOT_DPB
                                    ; 17/12/2022
00006558 E99C00                     jmp     FAT_GOT_DPB
                                ;fatread_sft_retn:
                                    ;retn

                                ;--------------------------------------------------------------------------
                                ;
                                ; Procedure Name : FATREAD_CDS
                                ;
                                ; Inputs:
                                ;   DS:DOSDATA
                                ;   ES:DI points to an CDS for the drive of interest (local only,
                                ;           giving a NET or NUL CDS will produce system crashing results).
                                ; Function:
                                ;   If disk may have been changed, media is determined and buffers are
                                ;   flagged invalid. If not, no action is taken.
                                ; Outputs:
                                ;   ES:BP = Drive parameter block
                                ;   THISDPB = ES:BP
                                ;   THISDRV set
                                ;   Carry set if error (currently user FAILed to I 24)
                                ; DS preserved , all other registers destroyed
                                ;
                                ;--------------------------------------------------------------------------

                                    ; 20/05/2019 - Retro DOS v4.0
                                    ; DOSCODE:96C5h (MSDOS 6.21, MSDOS.SYS)
                                    ; 27/11/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
                                    ; DOSCODE:9669h (MSDOS 5.0, MSDOS.SYS)

                                    ; 03/03/2024 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMDOS.COM)
                                    ; PCDOS 7.1 IBMDOS.COM - DOSCODE:0A6D8h
                                    ; (Windows ME IO.SYS - BIOSCODE:0C644h)

                                FATREAD_CDS:
0000655B 06                         PUSH    ES
0000655C 57                         PUSH    DI

                                    ;les    bp,[es:di+45h]
0000655D 26C46D45                   LES     BP,[ES:DI+curdir.devptr]

                                ; 03/03/2024
                                %if 0
                                    ; 27/11/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
                                    ;MOV    AL,[ES:BP+DPB.DRIVE] ; [es:bp+0]
                                    ; 15/12/2022
                                    mov     AL,[ES:BP]
                                    MOV     [THISDRV],AL
                                    call    GOTDPB                  ;Set THISDPB
                                    CALL    FAT_GOT_DPB
                                %else
                                    ; 03/03/2024 (PCDOS 7.1 IBMDOS.COM)
                                    ;;;
00006561 E8EAFF                     call    fatread_gotdpb
                                    ;;;
                                %endif
00006564 5F                         POP     DI                      ;Get back CDS pointer
00006565 07                         POP     ES
00006566 72E1                       jc      short fatread_sft_retn
00006568 7542                       JNZ     short NO_CHANGE             ;Media NOT changed

                                ;   Media changed. We now need to find all CDS structures which use this
                                ;   DPB and invalidate their ID pointers.

                                MED_CHANGE:
                                    ;XOR    AX,AX
                                    ;DEC    AX                      ; AX = -1
                                    ; 03/03/2024
0000656A B8FFFF                     mov     ax,0FFFFh ; -1

0000656D 1E                         PUSH    DS
0000656E 8A0E[4700]                 MOV     CL,[CDSCOUNT]
00006572 30ED                       XOR     CH,CH                   ; CX is number of structures
                                    ;lds    si,[es:di+45h]
00006574 26C57545                   LDS     SI,[ES:DI+curdir.devptr] ; Find all CDS with this devptr

                                ;hkn; SS override

                                ;   Find all CDSs with this DevPtr
                                ;
                                ;   (ax) = -1
                                ;   (ds:si) = DevPtr

00006578 36C43E[3C00]               LES     DI,[SS:CDSADDR]             ; (es:di) = CDS pointer
                                frcd20:
                                    ;;test  word [es:di+43h],8000h
                                    ;TEST   word [ES:DI+curdir.flags],curdir_isnet
0000657D 26F6454480                 TEST    byte [ES:DI+curdir.flags+1],(curdir_isnet>>8)
00006582 7522                       JNZ     short frcd25            ; Leave NET guys alone!!

                                    ; MSDOS 3.3
                                    ;push   es
                                    ;push   di
                                    ;les    di,[es:di+45h]
                                    ;;les   di,[ES:DI+curdir.devptr]
                                    ;call   POINTCOMP
```

```
37683                                   ;pop    di
37684                                   ;pop    es
37685                                   ;jnz    short frcd25
37686
37687                                   ; MSDOS 6.0
37688 00006584 263B7545                 cmp     si,[ES:DI+curdir.devptr]
37689 00006588 751C                     jne     short frcd25          ; no match
37690 0000658A 8CDB                     mov     bx,ds
37691 0000658C 263B5D47                 cmp     bx,[ES:DI+curdir.devptr+2]
37692 00006590 7514                     jne     short frcd25          ; CDS not for this drive
37693
37694                                   ; MSDOS 3.3 (& MSDOS 6.0)
37695                                   ;test   [es:di+49h],ax ; PCDOS 7.1 IBMDOS.COM - DOSCODE:0A70Fh
37696 00006592 26854549                 test    [ES:DI+curdir.ID],AX
37697                                   ;JZ     short frcd25 ; = 0    ; If root (0), leave root
37698                                           ; !!! test es:[di+49h],ax
37699                                           ; !!! jz short frcd25
37700                                           ; PCDOS 7.1 IBMDOS.COM bug (!?) ; Erdogan Tan - 03/03/2024
37701                                           ; test dword ptr es:[di+49h],-1 ; Win ME - BIOSCODE:0C694h
37702                                           ; jz short frcd25
37703                                           ;
37704                                   ; 03/03/2024 - Retro DOS v5.0
37705 00006596 7506                     jnz     short frcd24
37706
37707                                   ; 03/03/2024 (PCDOS 7.1 IBMDOS.COM)
37708                                   ;;;
37709                                   ;test   [es:di+4Bh],ax
37710 00006598 2685454B                 test    [es:di+curdir.ID+2],AX
37711 0000659C 7408                     jz      short frcd25 ; = 0
37712                                 frcd24:           ; 03/03/2024
37713                                   ;;;
37714                                   ;mov    [es:di+49h],ax
37715 0000659E 26894549                 MOV     [ES:DI+curdir.ID],AX  ; else invalid
37716                                   ; 03/03/2024 (PCDOS 7.1 IBMDOS.COM)
37717                                   ;;;
37718                                   ;mov    [es:di+4Bh],ax
37719 000065A2 2689454B                 mov     [es:di+curdir.ID+2],ax ; -1
37720                                   ;;;
37721                                 frcd25:
37722                                     ;;add   di,81  ; MSDOS 3.3
37723                                     ;add    di,88  ; MSDOS 6.0
37724 000065A6 83C758                   ADD     DI,curdir.size        ; Point to next CDS
37725 000065A9 E2D2                     LOOP    frcd20
37726 000065AB 1F                       POP     DS
37727                                 NO_CHANGE:
37728 000065AC C42E[8A05]               LES     BP,[THISDPB]
37729 000065B0 F8                       CLC
37730 000065B1 C3                       retn
37731
37732                                 ; =============== S U B R O U T I N E =======================================
37733                                 ; 05/01/2024 - Retro DOS 5.0
37734                                 ; PCDOS 7.1 IBMDOS.COM - DOSCODE:0A7Fh
37735                                 ;;;
37736                                 chk_set_first_access:
37737                                     ;cmp    word [es:bp+0Fh],0
37738 000065B2 26837E0F00               cmp     word [es:bp+DPB.FAT_SIZE],0
37739 000065B7 750D                     jnz     short chk_set_fa_1    ; FAT (FAT12 or FAT16)
37740                                   ; FAT32
37741                                   ;cmp    word [es:bp+21h],0FFFFh
37742 000065B9 26837E21FF               cmp     word [es:bp+DPB.FREE_CNT_HW],0FFFFh ; -1
37743                                   ;mov    word [es:bp+21h],0FFFFh       ; High word of free cluster count
37744 000065BE 26C74621FFFF             mov     word [es:bp+DPB.FREE_CNT_HW],0FFFFh ; -1
37745 000065C4 7505                     jne     short chk_set_fa_2
37746                                 chk_set_fa_1:
37747                                   ;cmp    word [es:bp+1Fh],0FFFFh
37748 000065C6 26837E1FFF               cmp     word [es:bp+DPB.FREE_CNT],0FFFFh ; -1
37749                                 chk_set_fa_2:
37750                                   ;mov    word [es:bp+1Fh],0FFFFh       ; Count of free clusters, -1 if unknown
37751 000065CB 26C7461FFFFF             mov     word [es:bp+DPB.FREE_CNT],0FFFFh ; -1
37752 000065D1 7405                     je      short chk_set_fa_3
37753                                   ;or     byte [es:bp+18h],1
37754 000065D3 26804E1801               or      byte [es:bp+DPB.FIRST_ACCESS],1
37755                                 chk_set_fa_3:
37756 000065D8 C3                       retn
37757                                 ;;;
37758                                 ;--------------------------------------------------------------------------
37759
37760
37761                                 ;Break      <Fat_Operation - miscellaneous fat stuff>
37762                                 ;--------------------------------------------------------------------------
37763                                 ;
37764                                 ; Procedure Name : FAT_operation
37765                                 ;
37766                                 ;--------------------------------------------------------------------------
37767
37768                                     ; 27/11/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
37769
37770                                     ; 04/03/2024
37771                                     ; 03/03/2024 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMDOS.COM)
37772
37773                                 FAT_operation:
37774                                     ; 31/07/2018 - Retro DOS v3.0
37775                                 FATERR:
37776
37777                                 ; 03/03/2024
37778                                 %if 0
37779                                     ;mov    word [es:bp+1Eh],-1
37780                                     ;mov    word [es:bp+1Fh],-1 ; MSDOS 6.0
37781                                     MOV     word [ES:BP+DPB.FREE_CNT],-1
37782                                                                   ; Err in FAT must force recomp of freespace
37783                                 %else
37784                                     ; 03/03/2024 (PCDOS 7.1 IBMDOS.COM)
37785                                     ;;;
37786 000065D9 E8D6FF                   call    chk_set_first_access
37787                                     ;;;
37788                                 %endif
37789                                     ;and    di,0FFh
37790 000065DC 81E7FF00                 AND     DI,STECODE            ; Put error code in DI
37791                                     ;mov    byte [ALLOWED],18h
37792 000065E0 C606[4B03]18             MOV     byte [ALLOWED],Allowed_FAIL+Allowed_RETRY
37793                                     ;mov    ah,1Ah
37794 000065E5 B41A                     MOV     AH,2+Allowed_FAIL+Allowed_RETRY ; While trying to read FAT
37795 000065E7 A0[7605]                 MOV     AL,[THISDRV]          ; Tell which drive
37796 000065EA E8F2FA                   call    FATAL1
37797 000065ED C42E[8A05]               LES     BP,[THISDPB]
37798 000065F1 3C03                     CMP     AL,3
37799 000065F3 7502                     JNZ     short FAT_GOT_DPB     ; User said retry
37800
37801                                 FATERR_fail:       ; 03/03/2024
37802 000065F5 F9                       STC                           ; User said FAIL
37803 000065F6 C3                       retn
37804
37805                                 FAT_GOT_DPB:
37806                                     ;Context DS                   ;hkn; SS is DOSDATA
```

```
37807 000065F7 16                       push    ss
37808 000065F8 1F                       pop     ds
37809
37810                                   ; 03/03/2024 (PCDOS 7.1 IBMDOS.COM)
37811                                   ;;;
37812 000065F9 8CC0                     mov     ax,es
37813 000065FB 09E8                     or      ax,bp
37814 000065FD 74F6                     jz      short FATERR_fail
37815                                   ;;;
37816
37817                                   ;;mov   al,0Fh
37818                                   ;MOV    AL,DMEDHL
37819                                   ; 03/03/2024 (PCDOS 7.1 IBMDOS.COM)
37820                                   ;;;
37821 000065FF B013                     mov     al,19
37822                                   ;;;
37823
37824                                   ;mov    ah,[es:bp+1]
37825 00006601 268A6601                 MOV     AH,[ES:BP+DPB.UNIT]
37826 00006605 A3[5A03]                 MOV     [DEVCALL_REQLEN],AX ; 09/09/2018
37827 00006608 C606[5C03]01             MOV     BYTE [DEVCALL_REQFUNC],DEVMDCH
37828 0000660D C706[5D03]0000           MOV     word [DEVCALL_REQSTAT],0
37829                                   ;;mov   al,[es:bp+16h]
37830                                   ;mov    al,[es:bp+17h] ; MSDOS 6.0
37831 00006613 268A4617                 MOV     AL,[ES:BP+DPB.MEDIA]
37832 00006617 A2[6703]                 MOV     [CALLMED],AL
37833 0000661A 06                       PUSH    ES
37834 0000661B 1E                       PUSH    DS
37835
37836                                 ;hkn; DEVCALL is in DOSDATA
37837 0000661C BB[5A03]                 MOV     BX,DEVCALL
37838                                   ;;lds   si,[es:bp+12h]
37839                                   ;lds    si,[es:bp+13h] ; MSDOS 6.0
37840 0000661F 26C57613                 LDS     SI,[ES:BP+DPB.DRIVER_ADDR] ; DS:SI Points to device header
37841 00006623 07                       POP     ES                      ; ES:BX Points to call header
37842 00006624 E86EEC                   call    DEVIOCALL2
37843                                   ;Context DS                     ;hkn; SS is DOSDATA
37844 00006627 16                       push    ss
37845 00006628 1F                       pop     ds
37846 00006629 07                       POP     ES                      ; Restore ES:BP
37847 0000662A 8B3E[5D03]               MOV     DI,[DEVCALL_REQSTAT]
37848                                   ;test   di,8000h
37849                                   ;jnz    short FATERR
37850 0000662E 09FF                     or      di,di
37851 00006630 78A7                     js      short FATERR            ; have error
37852
37853                                 ; 03/03/2024
37854                                 %if 0
37855                                     XOR     AH,AH
37856                                     ;xchg   ah,[es:bp+17h] ; MSDOS 3.3
37857                                     ;xchg   ah,[es:bp+18h] ; MSDOS 6.0
37858                                     XCHG    AH,[ES:BP+DPB.FIRST_ACCESS] ; Reset dpb_first_access
37859                                 %else
37860                                     ; 03/03/2024 (PCDOS 7.1 IBMDOS.COM)
37861                                     ;;;
37862                                     ;mov    ah,[es:bp+18h]
37863 00006632 268A6618                 mov     ah,[es:bp+DPB.FIRST_ACCESS]
37864 00006636 80E480                   and     ah,80h          ; izolate (FAT) first access bit
37865                                     ;and    byte [es:bp+18h],7Fh
37866 00006639 268066187F               and     byte [es:bp+DPB.FIRST_ACCESS],7Fh
37867                                                     ; clear first access (FAT) bit 7
37868                                     ;;;
37869                                 %endif
37870
37871 0000663E A0[7605]                 MOV     AL,[THISDRV]            ; Use physical unit number
37872                                 ; See if we had changed volume id by creating one on the diskette
37873 00006641 3806[A10A]               cmp     [VOLCHNG_FLAG],AL
37874 00006645 7508                     jnz     short CHECK_BYT
37875 00006647 C606[A10A]FF             mov     byte [VOLCHNG_FLAG],-1 ; 0FFh
37876 0000664C E9B000                   jmp     GOGETBPB                ; Need to get device driver to read in
37877                                                                   ; new volume label.
37878                                 CHECK_BYT:
37879 0000664F 0A26[6803]               OR      AH,[CALLRBYT]
37880                                   ;JNS    short CHECK_ZR          ; ns = 0 or 1
37881                                   ;JMP    short NEWDSK
37882                                   ; 17/12/2022
37883 00006653 785E                     js      short NEWDSK
37884                                   ; 27/11/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
37885                                   ;JNS    short CHECK_ZR          ; ns = 0 or 1
37886                                   ;JMP    short NEWDSK
37887
37888                                 CHECK_ZR:
37889 00006655 743B                     JZ      short CHKBUFFDIRT       ; jump if I don't know
37890                                   ; 24/09/2023
37891                                   ; cf=0 (after 'or' instruction)
37892                                   ;CLC
37893 00006657 C3                       retn                            ; If Media not changed (NZ)
37894
37895                                 DISK_CHNG_ERR:
37896 00006658 06                       PUSH    ES
37897 00006659 55                       PUSH    BP
37898                                   ;;les   bp,[es:bp+12h]
37899                                   ;les    bp,[es:bp+13h] ; MSDOS 6.0
37900 0000665A 26C46E13                 LES     BP,[ES:BP+DPB.DRIVER_ADDR] ; Get device pointer
37901                                   ;;test  word [es:bp+4],800h
37902                                   ;TEST   word [ES:BP+SYSDEV.ATT],DEVOPCL ; Did it set vol id?
37903 0000665E 26F6460508               test    byte [es:bp+SYSDEV.ATT+1],(DEVOPCL>>8)
37904 00006663 5D                       POP     BP
37905 00006664 07                       POP     ES
37906                                   ;JZ     short FAIL_OPJ2                 ; Nope, FAIL
37907                                   ; 03/03/2024
37908 00006665 7477                     jz      short FAIL_OP
37909 00006667 1E                       PUSH    DS                      ; Save buffer pointer for ignore
37910 00006668 57                       PUSH    DI
37911 00006669 16                       push    ss                      ;hkn; SS is DOSDATA
37912 0000666A 1F                       pop     ds
37913                                   ;mov    byte [ALLOWED],18h
37914 0000666B C606[4B03]18             MOV     byte [ALLOWED],Allowed_FAIL+Allowed_RETRY
37915 00006670 06                       PUSH    ES
37916 00006671 C43E[6903]               LES     DI,[CALLVIDM]           ; Get volume ID pointer
37917 00006675 8C06[2A03]               MOV     [EXTERRPT+2],ES
37918 00006679 07                       POP     ES
37919 0000667A 893E[2803]               MOV     [EXTERRPT],DI
37920                                   ;mov    ax,0Fh
37921 0000667E B80F00                   MOV     AX,error_I24_wrong_disk
37922 00006681 C606[7505]01             MOV     byte [READOP],1                 ; Write
37923                                   ;invoke HARDERR
37924 00006686 E8CBF9                   call    HARDERR
37925 00006689 5F                       POP     DI                      ; Get back buffer for ignore
37926 0000668A 1F                       POP     DS
37927 0000668B 3C03                     CMP     AL,3
37928                                 FAIL_OPJ2:
37929 0000668D 744F                     JZ      short FAIL_OP
37930 0000668F E965FF                   JMP     FAT_GOT_DPB             ; Retry
```

```
37931
37932                                   CHKBUFFDIRT:
37933                                       ; 20/05/2019 - Retro DOS v4.0
37934
37935                                       ; MSDOS 3.3
37936                                       ;lds    di,[BUFFHEAD]
37937
37938                                       ; MSDOS 6.0
37939                                       ;cmp    word [ss:DirtyBufferCount],0  ; any dirty buffers ? ;hkn;
37940                                       ; 03/03/2024 - Retro DOS v5.0
37941                                       ; ds=ss
37942                                       ;;;
37943 00006692 833E[7100]00                  cmp     word [DirtyBufferCount],0 ; (Win ME IO.SYS - BIOSCODE:0C7A7h)
37944                                       ;;;
37945 00006697 741A                         je      short NEWDSK                    ; no, skip the check
37946 00006699 E81D02                       call    GETCURHEAD                      ; get pointer to first buffer
37947                                   nbuffer:
37948                                       ;cmp    al,[di+4]
37949 0000669C 384504                       cmp     [di+BUFFINFO.buf_ID],al         ; Unit OK ?
37950 0000669F 7509                         jne     short lfnxt                     ; no, go for next buffer
37951                                       ;test   byte [di+5],40h
37952 000066A1 F6450540                     TEST    byte [di+BUFFINFO.buf_flags],buf_dirty      ; is the buffer dirty ?
37953 000066A5 7403                         jz      short lfnxt                     ; no, go for next buffer
37954
37955                                   FAIL_OP2:   ; 03/03/2024
37956                                       ;Context DS
37957 000066A7 16                           push    ss
37958 000066A8 1F                           pop     ds
37959                                       ; 24/09/2023
37960                                       ; cf=0 (after 'test' instruction)
37961                                       ;clc
37962 000066A9 C3                           retn
37963
37964                                   lfnxt:
37965                                       ; 15/08/2018 - Retro DOS v3.0
37966                                       ; MSDOS 3.3
37967                                       ;lds    di,[di]
37968
37969                                       ; 20/05/2019 - Retro DOS v4.0
37970 000066AA 8B3D                         mov     di,[di]
37971                                       ;;mov   di,[di+BUFFINFO.buf_next]     ; get next buffer
37972
37973                                       ; MSDOS 3.3
37974                                       ;cmp    di,-1
37975                                       ;jne    short nbuffer
37976
37977                                       ; MSDOS 6.0
37978 000066AC 36393E[7E11]                 cmp     [ss:FIRST_BUFF_ADDR],di             ; is this where we started ?;hkn;
37979 000066B1 75E9                         jne     short nbuffer                   ; no, check this guy also
37980
37981                                   ; If no dirty buffers, assume Media changed
37982                                   NEWDSK:
37983                                       ;;mov   word [es:bp+1Eh],0FFFFh  ; MSDOS 3.3
37984                                       ;mov    word [es:bp+1Fh],0FFFFh  ; MSDOS 6.0
37985 000066B3 26C7461FFFFF                 mov     word [ES:BP+DPB.FREE_CNT],-1  ; Media changed, must
37986                                                                       ;  recompute
37987                                       ; 03/03/2024 (PCDOS 7.1 IBMDOS.COM)
37988                                       ;;;
37989                                       ;cmp    word [es:bp+0Fh],0
37990 000066B9 26837E0F00                   cmp     word [es:bp+DPB.FAT_SIZE],0 ; = 0 for FAT32 fs
37991 000066BE 7506                         jnz     short newdsk2   ; not FAT32
37992                                       ;mov    word [es:bp+21h],0FFFFh
37993 000066C0 26C74621FFFF                 mov     word [ES:BP+DPB.FREE_CNT_HW],0FFFFh ; -1
37994                                   newdsk2:
37995                                       ;;;
37996
37997                                       ; MSDOS 3.3
37998                                       ;call   SETVISIT
37999                                       ; MSDOS 6.0
38000 000066C6 E8F001                       call    GETCURHEAD
38001                                   nxbuffer:
38002                                       ; MSDOS 3.3
38003                                       ;or     byte [di+5],20h
38004                                       ; MSDOS 3.3 & MSDOS 6.0
38005                                       ;cmp    [di+4],al
38006 000066C9 384504                       cmp     [DI+BUFFINFO.buf_ID],al             ; This drive ?
38007 000066CC 7513                         jne     short lfnxt2
38008                                       ;test   byte [di+5],40h
38009 000066CE F6450540                     TEST    byte [DI+BUFFINFO.buf_flags],buf_dirty
38010 000066D2 7584                         jnz     short DISK_CHNG_ERR
38011                                       ;mov    word [di+4],20FFh
38012 000066D4 C74504FF20                   mov     word [DI+BUFFINFO.buf_ID],(buf_visit*256)+0FFh ; free up
38013 000066D9 E8EF01                       call    SCANPLACE
38014                                       ; MSDOS 6.0
38015 000066DC EB05                         jmp     short skpbuff
38016
38017                                       ; 03/03/2024 - Retro DOS v5.0
38018                                   FAIL_OP:                                    ; This label & code is here
38019                                       ;Context DS                             ;  for reachability
38020                                       ;push   ss
38021                                       ;pop    ds
38022 000066DE F9                           STC
38023                                       ; 03/03/2024
38024                                       ;retn
38025                                   FAIL_OP2J:  ; 03/03/2024
38026 000066DF EBC6                         jmp     short FAIL_OP2 ; cf=1
38027
38028                                   lfnxt2:
38029 000066E1 8B3D                         mov     di,[di]
38030                                       ;mov    di,[di+BUFFINFO.buf_next]
38031                                   skpbuff:
38032                                       ; MSDOS 6.0
38033 000066E3 363B3E[7E11]                 cmp     di,[ss:FIRST_BUFF_ADDR]                                 ;hkn;
38034 000066E8 75DF                         jne     short nxbuffer
38035
38036 000066EA 36833E[7700]00               CMP     word [ss:SC_CACHE_COUNT],0 ;LB.  look ahead buffers ?       ;AN001;
38037 000066F0 740D                         JZ      short GOGETBPB          ;LB.  no                            ;AN001;
38038 000066F2 363A06[7511]                 CMP     AL,[ss:CurSC_DRIVE]     ;LB.  same as changed drive         ;AN001;
38039 000066F7 7506                         JNZ     short GOGETBPB          ;LB.  no                            ;AN001;
38040 000066F9 36C606[7511]FF               MOV     byte [ss:CurSC_DRIVE],-1 ;LB.  invalidate look ahead buffers ;AN000;
38041                                   ;lfnxt2:
38042                                       ; MSDOS 3.3
38043                                       ;call   SKIPVISIT
38044                                       ;jnz    short nxbuffer
38045                                   GOGETBPB:
38046                                       ; MSDOS 3.3 & MSDOS 6.0
38047                                       ;;lds   di,[es:bp+12h]
38048                                       ;lds    di,[es:bp+13h] ; MSDOS 6.0
38049 000066FF 26C57E13                     LDS     DI,[ES:BP+DPB.DRIVER_ADDR]
38050                                       ; 20/05/2019
38051                                       ;test   word [di+4],2000h
38052                                       ;TEST   word [DI+SYSDEV.ATT],ISFATBYDEV
38053 00006703 F6450520                     TEST    byte [DI+SYSDEV.ATT+1],(ISFATBYDEV>>8)
38054 00006707 7533                         JNZ     short GETFREEBUF
```

```
                                    ;context DS                 ;hkn; SS is DOSDATA
00006709 16                         push    ss
0000670A 1F                         pop     ds

                                    ; 03/03/2024 (PCDOS 7.1 IBMDOS.COM)
                                    ;;;
                                    ;mov    word [es:bp+2],200h
0000670B 26C746020002               mov     word [es:bp+DPB.SECTOR_SIZE],512 ; bytes per sector
                                    ;mov    word [es:bp+6],1
00006711 26C746060100               mov     word [es:bp+DPB.FIRST_FAT],1  ; starting sector of FATs
                                    ;mov    byte [es:bp+8],1
00006717 26C6460801                 mov     byte [es:bp+DPB.FAT_COUNT],1  ; number of FATs
                                    ;mov    word [es:bp+0Dh],3
0000671C 26C7460D0300               mov     word [es:bp+DPB.MAX_CLUSTER],3       ; cluster count + 1
                                    ;mov    word [es:bp+0Fh],1
00006722 26C7460F0100               mov     word [es:bp+DPB.FAT_SIZE],1   ; FAT sectors (16 bit)
00006728 C706[E80A]0000             mov     word [CLUSTNUM_HW],0  ; high word of cluster number (for UNPACK)
                                    ;;;

0000672E BB0200                     MOV     BX,2
00006731 E8ADFB                     CALL    UNPACK                  ; Read the first FAT sector into CURBUF
                                FAIL_OPJ:
                                    ;JC     short FAIL_OP
                                    ; 03/03/2024
00006734 72A9                       jc      short FAIL_OP2J ; cf=1
00006736 C53E[E205]                 LDS     DI,[CURBUF]
0000673A EB22                       JMP     SHORT GOTGETBUF

                                GETFREEBUF:
                                    ; 03/03/2024 (PCDOS 7.1 IBMDOS.COM)
                                    ;;;
0000673C 31D2                       xor     dx,dx ; 0 ; *
0000673E 36C53E[7A00]               lds     di,[ss:LoMemBuff]
                                    ;sub    di,24
00006743 83EF18                     sub     di,BUFINSIZ
00006746 363816[7900]               cmp     [ss:BuffInHMA],dl ; 0
0000674B 7511                       jnz     short GOTGETBUF             ; buffer is in HMA
                                                                        ; use [LowMemBuff] to transfer at first
                                    ;;;

0000674D 06                         PUSH    ES                      ; Get a free buffer for BIOS to use
0000674E 55                         PUSH    BP
                                    ; MSDOS 3.3
                                    ;LDS    DI,[SS:BUFFHEAD] ; 15/08/2018
                                    ; 03/03/2024  ; *
                                    ; MSDOS 6.0
                                    ;XOR    DX,DX ; *               ;LB.  fake to get 1st   ;AN000;
                                ;hkn; SS override
0000674F 368916[0706]               MOV     [SS:HIGH_SECTOR],DX ; 0       ;LB.  buffer addr       ;AN000;
00006754 E86201                     call    GETCURHEAD              ;LB.                    ;AN000;
                                    ; MSDOS 3.3 & MSDOS 6.0
00006757 E8C203                     call    BUFWRITE
0000675A 5D                         POP     BP
0000675B 07                         POP     ES
                                    ;;JC    short FAIL_OPJ
                                    ;jc     short FAIL_OP
                                    ; 03/03/2024 - Retro DOS v5.0
0000675C 7281                       jc      short FAIL_OP2J ; cf=1

                                GOTGETBUF:
                                    ;;;add  di,16
                                    ;;add   di,20 ; MSDOS 6.0
                                    ;add    di,24 ; PCDOS 7.1 ; 03/03/2024
0000675E 83C718                     ADD     DI,BUFINSIZ

                                ;hkn; SS override
00006761 368C1E[6A03]               MOV     [SS:CALLXAD+2],DS
                                    ;Context DS                 ;hkn; SS is DOSDATA
00006766 16                         push    ss
00006767 1F                         pop     ds
00006768 893E[6803]                 MOV     [CALLXAD],DI
                                    ;mov    al,16h
0000676C B016                       MOV     AL,DBPBHL ; 22
                                    ;mov    ah,[es:bp+1]
0000676E 268A6601                   MOV     AH,[ES:BP+DPB.UNIT]
00006772 A3[5A03]                   MOV     [DEVCALL_REQLEN],AX ; 09/09/2018
00006775 C606[5C03]02               MOV     BYTE [DEVCALL_REQFUNC],DEVBPB ; 2
0000677A C706[5D03]0000             MOV     word [DEVCALL_REQSTAT],0
                                    ;;mov   al,[es:bp+16h]
                                    ;mov    al,[es:bp+17h]
00006780 268A4617                   MOV     AL,[ES:BP+DPB.MEDIA]
00006784 A2[6703]                   MOV     [CALLMED],AL
00006787 06                         PUSH    ES
00006788 1E                         PUSH    DS

                                ; 04/03/2024
                                %if 0
                                    ;;push  word [es:bp+14h]
                                    ;push   word [es:bp+15h] ; MSDOS 6.0
                                    PUSH    WORD [ES:BP+DPB.DRIVER_ADDR+2]
                                    ;;push  word [es:bp+12h]
                                    ;push   word [es:bp+13h] ; MSDOS 6.0
                                    PUSH    WORD [ES:BP+DPB.DRIVER_ADDR]

                                ;hkn; DEVCALL is in DOSDATA
                                    MOV     BX,DEVCALL
                                    POP     SI
                                    POP     DS                      ; DS:SI Points to device header
                                %else
                                    ; 04/03/2024 - Retro DOS v5.0
                                    ; PCDOS 7.1 IBMDOS.COM
00006789 BB[5A03]                   mov     bx,DEVCALL
                                    ;lds    si,[es:bp+13h]
0000678C 26C57613                   lds     si,[es:bp+DPB.DRIVER_ADDR] ; DS:SI Points to device header
                                %endif

00006790 07                         POP     ES                      ; ES:BX Points to call header
                                    ;invoke DEVIOCALL2
00006791 E801EB                     call    DEVIOCALL2
00006794 07                         POP     ES                      ; Restore ES:BP
                                    ;Context DS
00006795 16                         push    ss                      ;hkn; SS is DOSDATA
00006796 1F                         pop     ds
00006797 8B3E[5D03]                 MOV     DI,[DEVCALL_REQSTAT]

                                ; 04/03/2024
                                %if 0
                                    ; MSDOS 3.3
                                    ;test   di,8000h
                                    ;jnz    short FATERRJ
                                    ; MSDOS 6.0
                                    or      di,di
                                    js      short FATERRJ           ; have error

```

```
38179                                   ; 04/03/2024
38180                                   ;;;mov  al,[es:bp+16h]
38181                                   ;;mov   al,[es:bp+17h]  ; MSDOS 6.0
38182                                   ;MOV    AL,[ES:BP+DPB.MEDIA]
38183
38184                                   LDS     SI,[CALLBPB]
38185                                   ;;mov   word [es:bp+1Ch],0
38186                                   ;mov    word [es:bp+1Dh],0 ; MSDOS 6.0
38187                                   MOV     word [ES:BP+DPB.NEXT_FREE],0 ; recycle scanning pointer
38188
38189                                   ;invoke $SETDPB
38190                                   call    _$SETDPB
38191
38192                                 ;hkn; SS override
38193                                   LDS     DI,[SS:CALLXAD]             ; Get back buffer pointer
38194
38195                                   ;mov    al,[es:bp+8]
38196                                   MOV     AL,[ES:BP+DPB.FAT_COUNT]
38197
38198                                   ; MSDOS 3.3
38199                                   ;;mov   ah,[es:bp+0Fh]
38200                                   ;MOV    AH,[ES:BP+DPB.FAT_SIZE]
38201                                   ;;mov   [DI-8],ax
38202                                   ;MOV    [DI+BUFFINFO.buf_wrtcnt-BUFINSIZ],AX
38203
38204                                   ; MSDOS 6.0
38205                                   ;mov    [di-0Ah],al
38206                                   MOV     [DI+BUFFINFO.buf_wrtcnt-BUFINSIZ],AL
38207                                                                   ;>32mb         ;AN000;
38208                                   ;mov    ax,[es:bp+0Fh]
38209                                   MOV     AX,[ES:BP+DPB.FAT_SIZE]             ;>32mb
38210                                   ;mov    [di-09h],ax                         ;AC000;
38211                                   MOV     [DI+BUFFINFO.buf_wrtcntinc-BUFINSIZ],AX
38212                                                                 ;>32mb Correct buffer info ;AC000;
38213                                   ;Context DS                   ;hkn; SS is DOSDATA
38214                                   push    ss
38215                                   pop     ds
38216                                   XOR     AL,AL                 ;Media changed (Z), Carry clear
38217                                   retn
38218
38219                                 FATERRJ:
38220                                   JMP     FATERR
38221
38222                                 %else
38223                                   ; 04/03/2024 - Retro DOS v5.0
38224                                   ; PCDOS 7.1 IBMDOS.COM
38225                                   ;;;
38226 0000679B 09FF                    or      di,di
38227 0000679D 790C                    jns     short gotgetbuf2
38228                                   ; error
38229 0000679F C53E[6803]              lds     di,[CALLXAD] ; Buffer (data) address
38230                                   ;mov    word [di-20],0FFh
38231 000067A3 C745ECFF00              mov     word [di+BUFFINFO.buf_ID-BUFINSIZ],0FFh
38232                                                         ; byte BUFFINFO.buf_ID = 0FFh ; FREE
38233                                                         ; byte BUFFINFO.buf_flags = 0
38234 000067A8 E92EFE                  jmp     FATERR
38235
38236                                 gotgetbuf2:
38237 000067AB C536[6C03]              lds     si,[CALLBPB]   ; Address of the BPB (DEVCALL offset 18)
38238
38239 000067AF 31C9                    xor     cx,cx  ; 0
38240                                   ;mov    [es:bp+1Dh],cx ; [ES:BP+DPB.NEXT_FREE] = 0
38241                                   ; 01/07/2024
38242 000067B1 26894E1D                mov     [es:bp+DPB.NEXT_FREE],cx ; 0
38243                                                         ; recycle scanning pointer
38244
38245 000067B5 BA5241                  mov     dx,4152h       ; 'RA' ; FAT32 extended BPB/DPB signature
38246
38247                                   ;cmp    [si+0Bh],cx    ; BPB.fatsecs ; 16 bit FAT size = 0 for FAT32 fs
38248 000067B8 394C0B                  cmp     [si+A_BPB.SECTORSPERFAT],cx ; 0
38249 000067BB 7514                    jnz     short gotgetbuf3 ; not FAT32
38250
38251                                   ;mov    [es:bp+39h],cx
38252 000067BD 26894E39                mov     [es:bp+DPB.FAT32_NXTFREE],cx ; 0
38253                                   ;mov    [es:bp+3Bh],cx
38254 000067C1 26894E3B                mov     [es:bp+DPB.FAT32_NXTFREE+2],cx ; 0
38255 000067C5 49                      dec     cx      ; -1
38256                                   ;mov    [es:bp+1Fh],cx ; (DPB.FREE_CNT) set free count to -1 (unknown)
38257 000067C6 26894E1F                mov     [es:bp+DPB.FREE_CNT],cx ; 0FFFFh
38258                                   ;mov    [es:bp+21h],cx
38259 000067CA 26894E21                mov     [es:bp+DPB.FREE_CNT_HW],cx ; 0FFFFh
38260                                   ;
38261 000067CE B95845                  mov     cx,4558h       ; 'XE' ; FAT32 extended BPB/DPB signature
38262                                 gotgetbuf3:
38263
38264                                   ;invoke $SETDPB
38265 000067D1 E891AC                  call    _$SETDPB
38266
38267                                 ;hkn; SS override
38268 000067D4 36C53E[6803]              LDS     DI,[SS:CALLXAD]             ; Get back buffer pointer
38269
38270                                   ;mov    dx,[es:bp+25h]
38271 000067D9 268B5625                mov     dx,[es:bp+DPB.FSINFO_SECTOR]
38272 000067DD 31C9                    xor     cx,cx  ; 0
38273                                   ;cmp    [es:bp+0Fh],cx
38274 000067DF 26394E0F                cmp     [es:bp+DPB.FAT_SIZE],cx      ; 16 bit FAT size field
38275 000067E3 7403                    jz      short gotgetbuf4 ; FAT32 fs
38276 000067E5 E99400                  jmp     gotgetbuf12    ; FAT fs
38277
38278                                 gotgetbuf4:
38279 000067E8 83FAFF                  cmp     dx,0FFFFh             ; invalid ?
38280 000067EB 7503                    jnz     short gotgetbuf5      ; no
38281 000067ED E98C00                  jmp     gotgetbuf12           ; skip reading FSINFO sector
38282
38283                                 gotgetbuf5:
38284 000067F0 36890E[0706]            mov     [ss:HIGH_SECTOR],cx ; 0
38285 000067F5 89FB                    mov     bx,di
38286
38287                                   ;cmp    byte [ss:BuffInHMA],0
38288 000067F7 36380E[7900]            cmp     [ss:BuffInHMA],cl ; 0
38289 000067FC 7405                    jz      short gotgetbuf6      ; buffer is in conventional (<=640KB) memory
38290 000067FE 36C51E[7A00]            lds     bx,[ss:LoMemBuff]     ; use a buffer in conventional memory
38291                                   ;mov    di,bx
38292                                 gotgetbuf6:
38293 00006803 36C606[4B03]18          mov     byte [ss:ALLOWED],Allowed_FAIL+Allowed_RETRY ; 18h
38294 00006809 41                      inc     cx
38295                                   ;push   di
38296 0000680A 53                      push    bx
38297 0000680B E8E0D7                  call    DREAD
38298 0000680E 5B                      pop     bx
38299                                   ;pop    di
38300 0000680F 89DF                    mov     di,bx ; Retro DOS v5.0
38301 00006811 725F                    jc      short gotgetbuf11     ; ds:di = (FSINFO sector) buffer
38302
```

```
                                ; FSI_HeadSig = 41615252h
                                
00006813 813D5252               cmp     word [di],5252h             ; 'RR' ; check if it is a valid FSINFO sector
                                ;cmp    word [di+FSINFO.LeadSig],5252
00006817 7559                   jnz     short gotgetbuf11       ; not valid
                                ;cmp    word [di+2],4161h       ; 'aA' ; (NASM syntax)
00006819 817D026141             cmp     word [di+FSINFO.LeadSig+2],4161h
0000681E 7552                   jnz     short gotgetbuf11
                                ;cmp    word [di+484],7272h     ; 'rr' ; FSI_StrucSig = 61417272h
00006820 81BDE4017272           cmp     word [di+FSINFO.StrucSig],7272h
00006826 754A                   jnz     short gotgetbuf11
                                ;cmp    word [di+486],6141h     ; 'Aa'
00006828 81BDE6014161           cmp     word [di+FSINFO.StrucSig+2],6141h
0000682E 7542                   jnz     short gotgetbuf11       ; not valid

                                ; valid
00006830 52                     push    dx
00006831 53                     push    bx
00006832 51                     push    cx

                                ;mov    bx,[es:bp+2Fh]
00006833 268B5E2F               mov     bx,[es:bp+DPB.LAST_CLUSTER+2]
                                ;mov    cx,[es:bp+2Dh]
00006837 268B4E2D               mov     cx,[es:bp+DPB.LAST_CLUSTER]

                                ;mov    ax,[di+488]             ; FSI_FreeCount ; bx:cx = number of clusters + 1
0000683B 8B85E801               mov     ax,[di+FSINFO.Free_Count]
                                ;mov    dx,[di+490]             ; FSI_FreeCount+2
0000683F 8B95EA01               mov     dx,[di+FSINFO.Free_Count+2]

00006843 39DA                   cmp     dx,bx                   ; is Free Count >= (Number of Clusters + 1) ?
00006845 7502                   jne     short gotgetbuf7        ; if yes, it is invalid value (must be 0FFFFFFFFh)
00006847 39C8                   cmp     ax,cx
                              gotgetbuf7:
00006849 7308                   jnb     short gotgetbuf8        ; yes, invalid value (must be 0FFFFFFFFh)

                                ;mov    [es:bp+1Fh],ax          ; no,valid free count
0000684B 2689461F               mov     [es:bp+DPB.FREE_CNT],ax ; save free count into [es:bp+DPB.FREE_CNT]
                                ;mov    [es:bp+21h],dx
0000684F 26895621               mov     [es:bp+DPB.FREE_CNT+2],dx

                              gotgetbuf8:
                                ;mov    ax,[di+492]             ; FSI_Nxt_Free
00006853 8B85EC01               mov     ax,[di+FSINFO.Nxt_Free]
                                ;mov    dx,[di+494]
00006857 8B95EE01               mov     dx,[di+FSINFO.Nxt_Free+2]

0000685B 39DA                   cmp     dx,bx                   ; is the next free clust num >= (num of clusters + 1) ?
0000685D 7502                   jne     short gotgetbuf9        ; invalid (if dx > bx)
0000685F 39C8                   cmp     ax,cx
                              gotgetbuf9:
00006861 730C                   jnb     short gotgetbuf10 ; invalid

                                ;mov    [es:bp+39h],ax          ; save next free (search) cluster number
00006863 26894639               mov     [es:bp+DPB.FAT32_NXTFREE],ax ; into [es:bp+DPB.FAT32_NXTFREE]
                                ;mov    [es:bp+3Bh],dx
00006867 2689563B               mov     [es:bp+DPB.FAT32_NXTFREE+2],dx
                                ;mov    [es:bp+1Dh],ax          ; and into [es:bp+DPB.NEXT_FREE] ; low word
0000686B 2689461D               mov     [es:bp+DPB.NEXT_FREE],ax

                              gotgetbuf10:
0000686F 59                     pop     cx
00006870 5B                     pop     bx
00006871 5A                     pop     dx

                              gotgetbuf11:
00006872 36803E[7900]00         cmp     byte [ss:BuffInHMA],0 ; is buffer in HMA ?
                                ;jnz    short gotgetbuf12       ; yes
00006878 753A                   jnz     short gotgetbuf16 ; 04/03/2024

                                ;mov    word [di-20],0FFh       ; invalidate buffer (set it as free buffer)
                                ; 04/03/2024 - Retro DOS v5.0
0000687A EB0F                   jmp     short gotgetbuf17

                              gotgetbuf12:
0000687C 36803E[7900]00         cmp     byte [ss:BuffInHMA],0
00006882 7530                   jnz     short gotgetbuf16

                                ;cmp    word [es:bp+6],1
00006884 26837E0601             cmp     word [es:bp+DPB.FIRST_FAT],1 ; 1st FAT start sector
00006889 7405                   jz      short gotgetbuf13

                              gotgetbuf17:      ; 04/03/2024
                                ;mov    word [di-20],0FFh       ; invalidate buffer (set it as free buffer)
                                                                ; di = buffer header + 4 (BUFINFO.buf_ID)
0000688B C745ECFF00             mov     word [di+BUFFINFO.buf_ID-BUFINSIZ],0FFh

                              gotgetbuf13:
                                ;mov    al,[es:bp+8]
00006890 268A4608               mov     al,[es:bp+DPB.FAT_COUNT]
                                ;mov    [di-14],al              ; buffer header address + 10
00006894 8845F2                 mov     [di+BUFFINFO.buf_wrtcnt-BUFINSIZ],al

                                ;mov    ax,[es:bp+0Fh]
00006897 268B460F               mov     ax,[es:bp+DPB.FAT_SIZE]       ; 16 bit FAT size field
0000689B 09C0                   or      ax,ax
0000689D 750D                   jnz     short gotgetbuf14 ; FAT (FAT12 or FAT16) fs

                                ; FAT32 fs
                                ;mov    ax,[es:bp+31h]          ; FAT sectors (per one FAT)
0000689F 268B4631               mov     ax,[es:bp+DPB.FAT32_SIZE]
                                ;mov    [di-13],ax      ; BUFFINFO.buf_wrtcntinc ; # sectors between each write
000068A3 8945F3                 mov     [di+BUFFINFO.buf_wrtcntinc-BUFINSIZ],ax
                                ;mov    ax,[es:bp+33h]
000068A6 268B4633               mov     ax,[es:bp+DPB.FAT32_SIZE+2]
000068AA EB05                   jmp     short gotgetbuf15

                              gotgetbuf14:
                                ;mov    [di-13],ax              ; BUFFINFO.buf_wrtcntinc
000068AC 8945F3                 mov     [di+BUFFINFO.buf_wrtcntinc-BUFINSIZ],ax

                                ;xor    ax,ax                   ; 0
000068AF 29C0                   sub     ax,ax ; 0

                              gotgetbuf15:
                                ;;mov   [di-15],ax              ; BUFFINFO.buf_wrtcntinc+2 ; hw of sectors per FAT
                                                                ; PCDOS 7.1 BUG !? This would be 'mov [di-11],ax'
                                                                ; Erdogan Tan - 03/03/2024
                                ;mov    [di-11],ax
000068B1 8945F5                 mov     [di+BUFFINFO.buf_wrtcntinc+2-BUFINSIZ],ax

                              gotgetbuf16:
                                ;mov    ax,ss                   ; SS is DOSDATA
                                ;mov    ds,ax
```

```
000068B4 16                         push    ss
000068B5 1F                         pop     ds
                                    ;xor    al,al                   ; Media changed (Z),Carry clear
000068B6 31C0                       xor     ax,ax
000068B8 C3                         retn

                                    ;;;
                                %endif

                                ;===========================================================================
                                ; STDBUF.ASM
                                ;===========================================================================
                                ; Retro DOS v2.0 - 12/03/2018

                                ;
                                ; Standard buffer management for MSDOS
                                ;

                                ;.xlist
                                ;.xcref
                                ;INCLUDE STDSW.ASM
                                ;.cref
                                ;.list

                                ;TITLE      STDBUF - MSDOS buffer management
                                ;NAME       STDBUF

                                ;INCLUDE BUF.ASM

                                ;===========================================================================
                                ; BUF.ASM
                                ;===========================================================================
                                ; 31/07/2018 - Retro DOS v3.0
                                ; Retro DOS v2.0 - 12/03/2018
                                ;
                                ; buffer management for MSDOS
                                ;
                                ;CODE       SEGMENT BYTE PUBLIC  'CODE'
                                ;       ASSUME  SS:DOSGROUP,CS:DOSGROUP
                                ;
                                ;SUBTTL SETVISIT,SKIPVISIT -- MANAGE BUFFER SCANS
                                ;
                                ;SETVISIT:
                                ;   ; 31/07/2018 - Retro DOS v3.0
                                ;   ; IBMDOS.COM (MSDOS 3.3, 1987) - Offset 5CAFh
                                ;; Inputs:
                                ;;      None
                                ;; Function:
                                ;;      Set up a scan of I/O buffers
                                ;; Outputs:
                                ;;      All visit flags = 0
                                ;;              NOTE: This pre-scan is needed because a hard disk error
                                ;;                    may cause a scan to stop in the middle leaving some
                                ;;                    visit flags set, and some not set.
                                ;;      DS:DI Points to [BUFFHEAD]
                                ;; No other registers altered
                                ;
                                ;       LDS     DI,[SS:BUFFHEAD] ; 15/03/2018
                                ;   PUSH    AX
                                ;       ;;XOR      AX,AX    ;; MSDOS 2.11
                                ;   ;mov    al,0DFh
                                ;   mov     al,~buf_visit
                                ;SETLOOP:
                                ;       ;;MOV      [DI+7],AL ;; MSDOS 2.11
                                ;   ;and    [DI+5],al
                                ;   AND     [DI+BUFFINFO.buf_flags],AL
                                ;       LDS     DI,[DI]
                                ;       CMP     DI,-1
                                ;       JNZ     SHORT SETLOOP
                                ;       POP     AX ; 09/09/2018
                                ;   LDS     DI,[SS:BUFFHEAD] ; 15/03/2018
                                ;SVISIT_RETN:
                                ;       RETN
                                ;
                                ;SKIPVISIT:
                                ;   ; 31/07/2018 - Retro DOS v3.0
                                ;   ; IBMDOS.COM (MSDOS 3.3, 1987) - Offset 5CC8h
                                ;
                                ;; Inputs:
                                ;;      DS:DI Points to a buffer
                                ;; Function:
                                ;;      Skip visited buffers
                                ;; Outputs:
                                ;;      DS:DI Points to next unvisited buffer
                                ;;      Zero is set if skip to LAST buffer
                                ;; No other registers altered
                                ;
                                ;       CMP     DI,-1
                                ;       ;retz
                                ;       JZ  SHORT SVISIT_RETN
                                ;
                                ;   ;;CMP   BYTE [DI+7],1 ;; MSDOS 2.11
                                ;       ;;;retnz
                                ;       ;;JNZ      SHORT SVISIT_RETN
                                ;
                                ;   ;test   byte [di+5],20h
                                ;   TEST    byte [DI+BUFFINFO.buf_flags],buf_visit
                                ;   JNZ     short SKIPLOOP
                                ;
                                ;   push    ax
                                ;   or      al,1
                                ;   pop     ax
                                ;   retn
                                ;
                                ;SKIPLOOP:
                                ;   LDS     DI,[DI]
                                ;       JMP     SHORT SKIPVISIT

                                ;===========================================================================
                                ; BUF.ASM, MSDOS 6.0, 1991
                                ;===========================================================================
                                ; 31/07/2018 - Retro DOS v3.0
                                ; 04/05/2019 - Retro DOS v4.0
                                ; 06/03/2024 - Retro DOS v5.0

                                ;   TITLE   BUF - MSDOS buffer management
                                ;   NAME    BUF

                                ;** BUF.ASM - Low level routines for buffer cache management
                                ;
                                ;   GETCURHEAD
                                ;   ScanPlace
                                ;   PLACEBUF
                                ;   PLACEHEAD
```

```
38551                                 ;   PointComp
38552                                 ;   GETBUFFR
38553                                 ;   GETBUFFRB
38554                                 ;   FlushBuf
38555                                 ;   BufWrite
38556                                 ;   SET_RQ_SC_PARMS
38557                                 ;
38558                                 ;   Revision history:
38559                                 ;
38560                                 ;          AN000  version 4.00  Jan. 1988
38561                                 ;          A004   PTM 3765 -- Disk reset failed
38562                                 ;          M039 DB 10/17/90 - Disk write optimization
38563                                 ;          I001   5.0 PTR 722211 - Preserve CY when in buffer in HMA
38564
38565                                 ;Break     <GETCURHEAD -- Get current buffer header>
38566                                 ;-----------------------------------------------------------------------------
38567                                 ; Procedure Name : GetCurHead
38568                                 ; Inputs:
38569                                 ;    No Inputs
38570                                 ; Function:
38571                                 ;   Returns the pointer to the first buffer in Queue
38572                                 ;   and updates FIRST_BUFF_ADDR
38573                                 ;       and invalidates LASTBUFFER (recency pointer)
38574                                 ; Outputs:
38575                                 ;   DS:DI = pointer to the first buffer in Queue
38576                                 ;   FIRST_BUFF_ADDR = offset ( DI ) of First buffer in Queue
38577                                 ;       LASTBUFFER = -1
38578                                 ; No other registers altered
38579                                 ;-----------------------------------------------------------------------------
38580
38581                                 ; 04/05/2019 - Retro DOS v4.0
38582                                 ; DOSCODE:98D2h (MSDOS 6.21, MSDOS.SYS)
38583                                 ; 27/11/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
38584                                 ; DOSCODE:9876h (MSDOS 5.0, MSDOS.SYS)
38585
38586                                 ; 06/03/2024 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMDOS.COM)
38587                                 ; PCDOS 7.1 IBMDOS.COM - DOSCODE:0AA4Bh
38588
38589                                 GETCURHEAD:
38590 000068B9 36C53E[6D00]             lds     di,[ss:BufferQueue]    ; Pointer to the first buffer
38591 000068BE 36C706[1E00]FFFF         mov     word [ss:LastBuffer],-1       ; invalidate last buffer
38592 000068C5 36893E[7E11]             mov     [ss:FIRST_BUFF_ADDR],di       ; save first buffer address
38593 000068CA C3                       retn
38594
38595                                 ;Break     <SCANPLACE, PLACEBUF -- PUT A BUFFER BACK IN THE POOL>
38596                                 ;-----------------------------------------------------------------------------
38597                                 ; Procedure Name : ScanPlace
38598                                 ; Inputs:
38599                                 ;   Same as PLACEBUF
38600                                 ; Function:
38601                                 ;   Save scan location and call PLACEBUF
38602                                 ; Outputs:
38603                                 ;   DS:DI Points to saved scan location
38604                                 ; All registers, except DS:DI, preserved.
38605                                 ;-----------------------------------------------------------------------------
38606                                 ;M039: Rewritten to preserve registers.
38607
38608                                 ;SCANPLACE:
38609                                 ;   ; 31/07/2018 - Retro DOS v3.0
38610                                 ;   ; IBMDOS.COM (MSDOS 3.3, 1987) - Offset 5DDCh
38611                                 ;   push    es
38612                                 ;   les     si,[di]
38613                                 ;   ;les    si,[DI+BUFFINFO.buf_link]
38614                                 ;   call    PLACEBUF
38615                                 ;   push    es
38616                                 ;   pop     ds
38617                                 ;   mov     di,si
38618                                 ;   pop     es
38619                                 ;scanplace_retn:
38620                                 ;   retn
38621
38622                                     ; MSDOS 6.0
38623                                 SCANPLACE:
38624 000068CB FF35                     push    word [di]
38625                                     ;push   word [di+BUFFINFO.buf_next] ;Save scan location
38626 000068CD E80200                   call    PLACEBUF
38627 000068D0 5F                       pop     di
38628 000068D1 C3                       retn
38629
38630                                 ;-----------------------------------------------------------------------------
38631                                 ; Procedure Name : PlaceBuf
38632                                 ; Input:
38633                                 ;   DS:DI points to buffer (DS->BUFFINFO array, DI=offset in array)
38634                                 ; Function:
38635                                 ;   Remove buffer from queue and re-insert it in proper place.
38636                                 ; NO registers altered
38637                                 ;-----------------------------------------------------------------------------
38638
38639                                 ;procedure PLACEBUF,NEAR
38640
38641                                     ; 27/11/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
38642                                     ; 20/05/2019 - Retro DOS v4.0
38643                                 PLACEBUF:
38644                                     ; 31/07/2018 - Retro DOS v3.0
38645
38646                                     ; MSDOS 6.0
38647 000068D2 50                       push    AX                  ;Save only regs we modify     ;AN000;
38648 000068D3 53                       push    BX                                                ;AN000;
38649                                     ; 23/09/2023
38650                                     ;push   SI                                                ;AN000;
38651
38652 000068D4 8B05                     mov     ax,[di]
38653                                     ;mov    ax,[di+BUFFINFO.buf_next]
38654 000068D6 368B1E[6D00]             mov     bx,[ss:BufferQueue]    ; bx = offset of head of list;smr;SS Override
38655
38656 000068DB 39D8                     cmp     ax,bx                       ;Buf = last?          ;AN000;
38657 000068DD 7422                     je      short nret                  ;Yes, special case    ;AN000;
38658 000068DF 39DF                     cmp     di,bx                       ;Buf = first?         ;AN000;
38659 000068E1 7506                     jne     short not_first             ;Yes, special case    ;AN000;
38660 000068E3 36A3[6D00]               mov     [ss:BufferQueue],ax         ;smr;SS Override
38661 000068E7 EB18                     jmp     short nret                  ;Continue with repositioning;AN000;
38662                                 not_first:
38663                                     ; 23/09/2023
38664 000068E9 56                       push    si
38665                                     ;mov    si,[di+2]
38666 000068EA 8B7502                   mov     SI,[DI+BUFFINFO.buf_prev]     ;No, SI = prior Buf   ;AN000;
38667 000068ED 8904                     mov     [si],ax
38668                                     ;mov    [SI+BUFFINFO.buf_next],AX     ; ax has di->buf_next ;AN000;
38669 000068EF 96                       xchg    si,ax
38670                                     ;mov    [si+2],ax
38671 000068F0 894402                   mov     [SI+BUFFINFO.buf_prev],AX     ;                     ;AN000;
38672
38673 000068F3 8B7702                   mov     SI,[BX+BUFFINFO.buf_prev]     ;SI-> last buffer     ;AN000;
38674 000068F6 893C                     mov     [si],di
```

```
                            ;mov    [SI+BUFFINFO.buf_next],DI     ;Add Buf to end of list ;AN000;
000068F8 897F02             mov     [BX+BUFFINFO.buf_prev],DI                             ;AN000;
000068FB 897502             mov     [DI+BUFFINFO.buf_prev],SI     ;Update link in Buf too       ;AN000;
000068FE 891D               mov     [di],bx
                            ;mov    [DI+BUFFINFO.buf_next],BX                             ;AN000;
                            ; 23/09/2023
00006900 5E                 pop     si
                        nret:
                            ; 23/09/2023                                                  ;AN000;
                            ;pop    SI                                                    ;AN000;
00006901 5B                 pop     BX                                                    ;AN000;
00006902 58                 pop     AX                                                    ;AN000;
                                                                                          ;AN000;
                            ;cmp    byte [di+4],0FFh
00006903 807D04FF           cmp     byte [di+BUFFINFO.buf_ID],-1  ; Buffer FREE?          ;AN000;
00006907 7505                   jne short pbx                     ; M039: -no, jump.
00006909 36893E[6D00]       mov     [ss:BufferQueue],di           ; M039: -yes, make it LRU.
                        pbx:
0000690E C3                 retn                                                          ;AN000;

                            ; 31/07/2018 - Retro DOS v3.0

                            ; MSDOS 3.3
                            ; IBMDOS.COM (MSDOS 3.3, 1987) - Offset 5DDCh

                        ;PLACEBUF:
                        ;   ; 15/03/2018 - Retro DOS v2.0 (MSDOS 2.11)
                        ;
                        ;       CALL       save_world
                        ;       LES     CX,[DI]
                        ;       CMP     CX,-1             ; Buf is LAST?
                        ;       JZ      SHORT NRET        ; Buffer already last
                        ;       MOV     BP,ES             ; Pointsave = Buf.nextbuf
                        ;       PUSH    DS
                        ;       POP     ES                ; Buf is ES:DI
                        ;   ; 15/03/2018
                        ;       LDS     SI,[SS:BUFFHEAD]  ; Curbuf = HEAD
                        ;       CALL    POINTCOMP         ; Buf == HEAD?
                        ;       JNZ     SHORT BUFLOOP
                        ;       MOV     [SS:BUFFHEAD],CX
                        ;       MOV     [SS:BUFFHEAD+2],BP        ; HEAD = Pointsave
                        ;       JMP     SHORT LOOKEND
                        ;BUFLOOP:
                        ;   ; 31/07/2018
                        ;   mov     ax,ds
                        ;   mov     bx,si
                        ;   ;lds    si,[SI+BUFFINFO.buf_link]
                        ;       LDS     SI,[SI]
                        ;       CALL    POINTCOMP
                        ;       jnz short BUFLOOP
                        ;   ;
                        ;   mov     ds,ax
                        ;   mov     si,bx
                        ;   mov     [SI],cx
                        ;   ;mov    [SI+BUFFINFO.buf_link],cx   ; If Curbuf.nextbuf == buf
                        ;   mov     [SI+2],bp
                        ;   ;mov    [BX+BUFFINFO.buf_link+2],bp ; Curbuf.nextbuf = Pointsave
                        ;LOOKEND:
                        ;   mov     ax,ds
                        ;   mov     bx,si
                        ;       LDS     SI,[SI]
                        ;       CMP     SI,-1
                        ;       jnz     short LOOKEND
                        ;GOTHEEND:
                        ;       mov ds,ax
                        ;   mov     [BX],di
                        ;   MOV      [BX+2],ES            ; Curbuf.nextbuf = Buf
                        ;       MOV     WORD [ES:DI],-1
                        ;   ;mov    word [ES:DI+BUFFINFO.buf_link],-1
                        ;       MOV     WORD [ES:DI+2],-1        ; Buf is LAST
                        ;   ;mov    word [ES:DI+BUFFINFO.buf_link+2],-1
                        ;NRET:
                        ;       CALL       restore_world
                        ;
                        ;   ;cmp    byte [di+4],-1
                        ;   cmp     byte [DI+BUFFINFO.buf_ID],-1  ; Free buffer ?
                        ;   jnz      short scanplace_retn
                        ;   call     PLACEHEAD
                        ;   retn

                        ;EndProc PLACEBUF

                        ;M039 - Removed PLACEHEAD.
                        ;--------------------------------------------------------------------------
                        ; places buffer at head
                        ;  NOTE:::::: ASSUMES THAT BUFFER IS CURRENTLY THE LAST
                        ;   ONE IN THE LIST!!!!!!!
                        ; BUGBUG ---- this routine can be removed because it has only
                        ; BUGBUG ---- one instruction. This routine is called from
                        ; BUGBUG ---- 3 places. ( Size = 3*3+6 = 15 bytes )
                        ; BUGBUG ---- if coded in line = 3 * 5 = 15 bytes
                        ; BUGBUG ---- But kept as it is for modularity
                        ;--------------------------------------------------------------------------
                        ;procedure   PLACEHEAD,NEAR
                        ;   mov     word ptr [BufferQueue], di
                        ;   ret
                        ;EndProc PLACEHEAD
                        ;M039

                        ;--------------------------------------------------------------------------
                        ; Procedure Name : PLACEHEAD
                        ;
                        ; SAME AS PLACEBUF except places buffer at head
                        ;--------------------------------------------------------------------------

                            ; MSDOS 3.3 (Retro DOS v3.0)
                            ; 05/09/2018
                            ; MSDOS 2.11 (Retro DOS v2.0)
                        ;PLACEHEAD:
                        ;   ; 31/07/2018 - Retro DOS v3.0
                        ;   ; IBMDOS.COM (MSDOS 3.3, 1987) - Offset 5D4Ah
                        ;
                        ;       CALL       save_world
                        ;       PUSH       DS
                        ;       POP ES
                        ;   ; 15/03/2018 - Retro DOS v2.0 (MSDOS 2.11)
                        ;       LDS     SI,[SS:BUFFHEAD]
                        ;   ; 31/07/2018 - Retro DOS v3.0 (MSDOS 3.3)
                        ;   CALL    POINTCOMP
                        ;       JZ      SHORT GOTHEEND2
                        ;   MOV     [ES:DI],SI
                        ;   ;mov    [ES:DI+BUFFINFO.buf_link],si
                        ;       MOV [ES:DI+2],DS
                        ;   ;mov    [ES:DI+BUFFINFO.buf_link+2],ds
```

```
38799                                   ;       MOV [SS:BUFFHEAD],DI
38800                                   ;       MOV [SS:BUFFHEAD+2],ES
38801                                   ;LOOKEND2:
38802                                   ;       mov ax,ds
38803                                   ;   mov     bx,si
38804                                   ;   ;lds    si,[SI+BUFFINFO.buf_link]
38805                                   ;       LDS     SI,[SI]
38806                                   ;       CALL    POINTCOMP
38807                                   ;       JNZ SHORT LOOKEND2 ; 05/09/2018
38808                                   ;       mov ds,ax
38809                                   ;   mov     word [bx],-1
38810                                   ;   ;mov    word [BX+BUFFINFO.buf_link],-1
38811                                   ;   mov     word [bx+2],-1
38812                                   ;   ;mov    word [BX+BUFFINFO.buf_link+2],-1
38813                                   ;GOTHEEND2:
38814                                   ;           call    restore_world
38815                                   ;placehead_retn:
38816                                   ;   retn
38817
38818                                   ; 20/05/2019 - Retro DOS v4.0
38819                                   ; DOSCODE:9928h (MSDOS 6.21, MSDOS.SYS)
38820
38821                                   ;Break      <POINTCOMP -- 20 BIT POINTER COMPARE>
38822                                   ;--------------------------------------------------------------------------
38823                                   ;
38824                                   ; Procedure Name : PointComp
38825                                   ; Inputs:
38826                                   ;         DS:SI & ES:DI
38827                                   ; Function:
38828                                   ;          Checks for ((SI==DI) && (ES==DS))
38829                                   ;      Assumes that pointers are normalized for the
38830                                   ;      same segment
38831                                   ;
38832                                   ; Compare DS:SI to ES:DI (or DS:DI to ES:SI) for equality
38833                                   ; DO NOT USE FOR < or >
38834                                   ; No Registers altered
38835                                   ;
38836                                   ;--------------------------------------------------------------------------
38837
38838                                   POINTCOMP:
38839                                       ; 31/07/2018 - Retro DOS v3.0
38840                                       ; IBMDOS.COM (MSDOS 3.3, 1987) - Offset 5D84h
38841 0000690F 39FE                       CMP     SI,DI
38842 00006911 750A                       jnz     short _ret_label      ; return if nz
38843                                     ;jnz    short placehead_retn
38844 00006913 51                         PUSH    CX
38845 00006914 52                         PUSH    DX
38846 00006915 8CD9                       MOV     CX,DS
38847 00006917 8CC2                       MOV     DX,ES
38848 00006919 39D1                       CMP     CX,DX
38849 0000691B 5A                         POP     DX
38850 0000691C 59                         POP     CX
38851                                   _ret_label:
38852 0000691D C3                         retn
38853
38854                                   ; 01/08/2018 - Retro DOS v3.0
38855                                   ; IBMDOS.COM (MSDOS 3.3, 1987) - Offset 5D93h
38856
38857                                   ;Break      <GETBUFFR, GETBUFFRB -- GET A SECTOR INTO A BUFFER>
38858
38859                                   ;** GetBuffr - Get a non-FAT Sector into a Buffer
38860                                   ;--------------------------------------------------------------------------
38861                                   ;   GetBuffr does normal ( non-FAT ) sector buffering
38862                                   ;   It gets the specified local sector into one of the I/O buffers
38863                                   ;   and shuffles the queue
38864                                   ;
38865                                   ;   ENTRY   (AL) = 0 means sector must be pre-read
38866                                   ;                  ELSE no pre-read
38867                                   ;           (DX) = Desired physical sector number             (LOW)
38868                                   ;           HIGH_SECTOR = Desired physical sector number (HIGH)
38869                                   ;           (ES:BP) = Pointer to drive parameters
38870                                   ;           ALLOWED set in case of INT 24
38871                                   ;   EXIT    'C' set if error (user FAIL response to INT24)
38872                                   ;           'C' clear if OK
38873                                   ;           CURBUF Points to the Buffer for the sector
38874                                   ;               the buffer type bits OF buf_flags = 0, caller must set it
38875                                   ;   USES    AX, BX, CX, SI, DI, Flags
38876                                   ;--------------------------------------------------------------------------
38877
38878                                   ;** GetBuffrb - Get a FAT Sector into a Buffer
38879                                   ;--------------------------------------------------------------------------
38880                                   ;   GetBuffrb reads a sector from the FAT file system's FAT table.
38881                                   ;   It gets the specified sector into one of the I/O buffers
38882                                   ;   and shuffles the queue. We need a special entry point so that
38883                                   ;   we can read the alternate FAT sector if the first read fails, also
38884                                   ;   so we can mark the buffer as a FAT sector.
38885                                   ;
38886                                   ;   ENTRY   (AL) = 0 means sector must be pre-read
38887                                   ;                  ELSE no pre-read
38888                                   ;           (DX) = Desired physical sector number            (LOW)
38889                                   ;           (SI) != 0
38890                                   ;           HIGH_SECTOR = Desired physical sector number (HIGH)
38891                                   ;           (ES:BP) = Pointer to drive parameters
38892                                   ;           ALLOWED set in case of INT 24
38893                                   ;   EXIT    'C' set if error (user FAIL response to INT24)
38894                                   ;           'C' clear if OK
38895                                   ;           CUR ddBUF Points to the Buffer for the sector
38896                                   ;               the buffer type bits OF buf_flags = 0, caller must set it
38897                                   ;   USES    AX, BX, CX, SI, DI, Flags
38898                                   ;--------------------------------------------------------------------------
38899
38900                                       ; 22/09/2023 - RetroDOS v4.2 MSDOS.SYS (optimization)
38901                                   GETBUFFRC:
38902                                       ;mov    word [HIGH_SECTOR],0
38903                                       ; 02/03/2024 - Retro DOS v5.0
38904 0000691E 891E[0706]                 mov     [HIGH_SECTOR],bx
38905                                   GETBUFFRA:
38906 00006922 30C0                       xor     al,al
38907 00006924 BE0100                     mov     si,1
38908 00006927 EB09                       jmp     short GETBUFFRB
38909
38910                                       ; 22/09/2023
38911                                   GETBUFFER:
38912 00006929 30C0                       xor     al,al
38913                                   GETBUFFRD:
38914                                       ;mov    byte [ALLOWED],18h
38915 0000692B C606[4B03]18               mov     byte [ALLOWED],Allowed_FAIL+Allowed_RETRY
38916
38917                                       ; 20/05/2019 - Retro DOS v4.0
38918                                       ; DOSCODE:9937h (MSDOS 6.21, MSDOS.SYS)
38919                                       ; 27/11/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
38920                                       ; DOSCODE:98DBh (MSDOS 5.0, MSDOS.SYS)
38921
38922                                       ; 07/07/2024
```

```
                                    ; 06/03/2024 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMDOS.COM)
                                    ; PCDOS 7.1 IBMDOS.COM - DOSCODE:0AAB0h

                                    ; (Windows Me IO.SYS - BIOSCODE:0CA3Dh)
                                    ; (MSDOS 6.22 MSDOS.SYS - DOSCODE:9937h)

                                GETBUFFR:
00006930 31F6                       XOR     SI,SI

                                ;   This entry point is called for FAT buffering with SI != 0

                                GETBUFFRB:
00006932 A3[9405]                   MOV     [PREREAD],AX                ; save pre-read flag

                                    ; 06/03/2024 (PCDOS 7.1 IBMDOS.COM)
                                    ;;;
00006935 09F6                       or      si,si
00006937 7429                       jz      short getb1

                                    ;cmp    word [es:bp+0Fh],0
00006939 26837E0F00                 cmp     word [es:bp+DPB.FAT_SIZE],0
0000693E 7522                       jnz     short getb1     ; not FAT32

                                    ;mov    ax,[es:bp+23h]
00006940 268B4623                   mov     ax,[es:bp+DPB.EXT_FLAGS]
00006944 A880                       test    al,80h          ; bit 7 -- 1 means only one FAT is active
00006946 741A                       jz      short getb1
00006948 83E00F                     and     ax,0Fh          ; Active FAT is the one referenced in bits 0-3
0000694B 7415                       jz      short getb1

0000694D 52                         push    dx
0000694E 89C1                       mov     cx,ax                   ; Zero based number of active FAT.
                                                                    ; (Only valid if mirroring is disabled.)
                                    ;mul    word [es:bp+33h]
00006950 26F76633                   mul     word [es:bp+DPB.FAT32_SIZE+2]
00006954 91                         xchg    ax,cx
                                    ;mul    word [es:bp+31h]
00006955 26F76631                   mul     word [es:bp+DPB.FAT32_SIZE]
00006959 01D1                       add     cx,dx
0000695B 5A                         pop     dx
0000695C 01C2                       add     dx,ax
0000695E 110E[0706]                 adc     [HIGH_SECTOR],cx
                                getb1:
                                    ;;;

                                    ; 15/12/2022
00006962 268A4600                   mov     al,[ES:BP]
                                    ; 27/11/2022 MSDOS 5.0 MSDOS.SYS compatibility)
                                    ;MOV    AL,[ES:BP+DPB.DRIVE] ; mov al,[es:bp+0]
00006966 C53E[1E00]                 LDS     DI,[LastBuffer]                     ; Get the recency pointer

                                    ; 06/03/2024 (PCDOS 7.1 IBMDOS.COM)
                                    ;;;
0000696A BBFFFF                     mov     bx,-1 ; 0FFFFh
                                    ;;;

                                    ; MSDOS 6.0
                                ;hkn; SS override
0000696D 368B0E[0706]               MOV     CX,[SS:HIGH_SECTOR]         ; F.C. >32mb   ;AN000;

                                ;   See if this is the buffer that was most recently returned.
                                ;   A big performance win if it is.

                                    ;CMP    DI,-1                       ; Recency pointer valid?
                                    ; 06/03/2024 - Retro DOS v5.0
00006972 39DF                       cmp     di,bx ; -1
00006974 7412                       je      short getb5                 ; No

                                    ;cmp    dx,[di+6]
00006976 3B5506                     CMP     DX,[DI+BUFFINFO.buf_sector]
00006979 750D                       JNZ     short getb5                 ; Wrong sector

                                    ; MSDOS 6.0
                                    ;cmp    cx,[di+8]
0000697B 3B4D08                     CMP     CX,[DI+BUFFINFO.buf_sector+2] ; F.C. >32mb   ;AN000;
0000697E 7508                       JNZ     short getb5                 ; F.C. >32mb   ;AN000;

                                    ;cmp    al,[di+4]
00006980 3A4504                     CMP     AL,[DI+BUFFINFO.buf_ID]
                                    ;JZ     getb35                      ; Just asked for same buffer
00006983 7503                       jnz     short getb5
                                    ;jmp    getb35
                                    ; 17/12/2022
                                    ; 28/07/2019
00006985 E90301                     jmp     getb35x
                                    ; 07/12/2022 MSDOS 5.0 MSDOS.SYS compatibility)
                                    ;jmp    getb35

                                ;   It's not the buffer most recently returned. See if it's in the
                                ;   cache.
                                ;
                                ;   (cx:dx) = sector #
                                ;   (al) = drive #
                                ;   (si) = 0 iff non fat sector, != 0 if FAT sector read
                                ;   ??? list may be incomplete ???

                                getb5:
                                    ; MSDOS 3.3
                                    ;lds    di,[SS:BUFFHEAD]
                                    ; MSDOS 6.0
00006988 E82EFF                     CALL    GETCURHEAD                  ; get Q Head
                                getb10:
                                    ;cmp    dx,[di+6]
0000698B 3B5506                     CMP     DX,[DI+BUFFINFO.buf_sector]
                                    ;jne    short getb12                ; wrong sector lo
                                    ; 06/03/2024 (PCDOS 7.1 IBMDOS.COM)
                                    ;;;
0000698E 750D                       jne     short getb11
                                    ;;;

                                    ; MSDOS 6.0
                                    ;cmp    cx,[di+8]
00006990 3B4D08                     CMP     CX,[DI+BUFFINFO.buf_sector+2]
                                    ;jne    short getb12                ; wrong sector hi
                                    ; 06/03/2024
                                    ;;;
00006993 7508                       jne     short getb11
                                    ;;;

                                    ;cmp    al,[di+4]
00006995 3A4504                     CMP     AL,[DI+BUFFINFO.buf_ID]
                                    ;;je    short getb25 ; 05/09/2018   ; Found the requested sector
                                    ;jne    short getb12
                                    ; 06/03/2024 (PCDOS 7.1 IBMDOS.COM)
```

```
                                    ;;;
000069998 7503                      jne     short getb11
                                    ;;;
0000699A E9A400                     jmp     getb25

                                getb11:
                                    ; 06/03/2024 (PCDOS 7.1 IBMDOS.COM)
                                    ;;;
                                    ;cmp    byte [di+4],0FFh
0000699D 807D04FF                   cmp     byte [di+BUFFINFO.buf_ID],0FFh      ; Free buffer ?
000069A1 7502                       jne     short getb12                ; no
000069A3 89FB                       mov     bx,di                       ; save buffer (offset) addr
                                    ;;;

                                getb12:
                                    ; MSDOS 3.3
                                    ;;mov   di,[DI]
                                    ;;;mov  di,[DI+BUFFINFO.buf_link]
                                    ;
                                    ; 15/08/2018
                                    ;lds    di,[di]

                                    ;cmp    di,-1 ; 0FFFFh
                                    ;jne    short getb10
                                    ;lds    di,[SS:BUFFHEAD]

                                    ; MSDOS 6.0
000069A5 8B3D                       mov     di,[di]
                                    ;mov    DI,[DI+BUFFINFO.BUF_NEXT]
000069A7 363B3E[7E11]               cmp     DI,[SS:FIRST_BUFF_ADDR]             ; back at the front again?
000069AC 75DD                       jne     short getb10                ; no, continue looking

                                    ; 06/03/2024 (PCDOS 7.1 IBMDOS.COM)
                                    ;;;
000069AE 83FBFF                     cmp     bx,0FFFFh       ; -1 ; invalid (not a free buffer address)
000069B1 7404                       je      short getb12x
000069B3 89DF                       mov     di,bx           ; restore free buffer (header offset) address
000069B5 EB16                       jmp     short getb13
                                getb12x:
                                    ;;;

                                ;   The requested sector is not available in the buffers. DS:DI now points
                                ;   to the first buffer in the Queue. Flush the first buffer & read in the
                                ;   new sector into it.
                                ;
                                ;   BUGBUG - what goes on here? Isn't the first guy the most recently
                                ;   used guy? Shuld be for fast lookup. If he is, we shouldn't take
                                ;   him, we should take LRU. And the above lookup shouldn't be
                                ;   down a chain, but should be hashed.
                                ;
                                ;   (DS:DI) = first buffer in the queue
                                ;   (CX:DX) = sector # we want
                                ;   (si) = 0 iff non fat sector, != 0 if FAT sector read

                                    ; MSDOS 3.3 & MSDOS 6.0
                                ;hkn; SS override
000069B7 51                         PUSH    CX  ; MSDOS 6.0
000069B8 56                         push    si
000069B9 52                         push    dx
000069BA 55                         push    bp
000069BB 06                         push    es
000069BC E85D01                     CALL    BUFWRITE                    ; Write out the dirty buffer
000069BF 07                         pop     es
000069C0 5D                         pop     bp
000069C1 5A                         pop     dx
000069C2 5E                         pop     si
000069C3 368F06[0706]               POP     word [SS:HIGH_SECTOR]  ; MSDOS 6.0
                                    ;jc     short getbx                 ; if got hard error
000069C8 7303                       jnc     short getb13
000069CA E9C800                     jmp     getbx

                                getb13:

                                ; 06/03/2024 (PCDOS 7.1 IBMDOS.COM)
                                %if 0
                                    ; MSDOS 6.0
                                    CALL    SET_RQ_SC_PARMS             ; set parms for secondary cache
                                %endif

                                ;   We're ready to read in the buffer, if need be. If the caller
                                ;   wanted to just *write* the buffer then we'll skip reading it in.

000069CD 30E4                       XOR     AH,AH                       ; initial flags
                                ;hkn; SS override
                                    ;test   byte [ss:PREREAD],0FFh
                                    ;jnz    short getb20
000069CF 363826[9405]               CMP     [SS:PREREAD],ah ; 0          ; am to Read in the new sector?
000069D4 7553                       JNZ     short getb20                ; no, we're done
                                    ;;;lea  bx,[di+16] ; MSDOS 3.3
                                    ;;lea   bx,[di+20] ; MSDOS 6.0
                                    ;lea    bx,[di+24] ; PCDOS 7.1; 06/03/2024
000069D6 8D5D18                     LEA     BX,[DI+BUFINSIZ]            ; (ds:bx) = data address
                                    ;MOV    CX,1
                                    ; 22/09/2023
000069D9 29C9                       sub     cx,cx ; 0
000069DB 56                         push    si
000069DC 57                         push    di
000069DD 52                         push    dx
                                    ; MSDOS 6.0
000069DE 06                         push    es ; ***

                                ; Note: As far as I can tell, all disk reads into buffers go through
                                ;   this point. -mrw 10/88

                                    ;cmp    byte [ss:BuffInHMA],0 ; is buffers in HMA?
                                    ; 22/09/2023
000069DF 36380E[7900]               cmp     [ss:BuffInHMA],cl ; 0
000069E4 7407                       jz      short getb14
000069E6 1E                         push    ds ; **
000069E7 53                         push    bx ; *
000069E8 36C51E[7A00]               lds     bx,[ss:LoMemBuff]     ; Then let's read it into scratch buff
                                getb14:
                                ;M039: Eliminated redundant HMA code.

                                    ; 22/09/2023
000069ED 41                         inc     cx ; cx = 1

                                    ; MSDOS 3.3 (& MSDOS 6.0)
000069EE 09F6                       OR      SI,SI                   ; FAT sector ?
000069F0 7407                       JZ      short getb15            ; no

000069F2 E896D5                     call    FATSECRD
                                    ;mov    ah,2
000069F5 B402                       MOV     AH,buf_isFAT            ; Set buf_flags
```

```
39171
39172 000069F7 EB05                     JMP     SHORT getb17            ; Buffer is marked free if read barfs
39173
39174                                   getb15:
39175 000069F9 E8F2D5                   call    DREAD                   ; Buffer is marked free if read barfs
39176 000069FC B400                     MOV     AH,0                    ; Set buf_flags to no type, DO NOT XOR!
39177                                   getb17:
39178
39179                                   ; 06/03/2024 - Retro DOS v5.0
39180                                   %if 0
39181                                       ; 17/12/2022
39182                                   ; 27/11/2022 MSDOS 5.0 MSDOS.SYS compatibility)
39183                                   ;;if 0
39184                                       ; MSDOS 6.0                                                   ;I001
39185                                       pushf                                                         ;I001
39186                                       cmp     byte [SS:BuffInHMA],0 ; did we read into scratch buff ? ;I001
39187                                       jz      short not_in_hma      ; no                            ;I001
39188                                       ;mov    cx,[es:bp+2]
39189                                       mov     cx,[ES:BP+DPB.SECTOR_SIZE]                            ;I001
39190                                       shr     cx,1                                                  ;I001
39191                                       popf                          ; Retrieve possible CY from DREAD ;I001
39192                                       mov     si,bx                                                 ;I001
39193                                       pop     di ; *                                                ;I001
39194                                       pop     es ; **                                               ;I001
39195                                       cld                                                           ;I001
39196                                       pushf                         ; Preserve possible CY from DREAD ;I001
39197                                       rep     movsw                 ; move the contents of scratch buf;I001
39198                                       push    es                                                    ;I001
39199                                       pop     ds                                                    ;I001
39200                                   ;;endif
39201
39202                                   ;; 17/12/2022
39203                                   ;%if 0
39204                                   ;   ; 27/11/2022 MSDOS 5.0 MSDOS.SYS compatibility)
39205                                   ;   ; MSDOS 5.0
39206                                   ;   pushf
39207                                   ;   cmp     byte [SS:BuffInHMA],0 ; did we read into scratch buff ?
39208                                   ;   jz      short not_in_hma      ; no
39209                                   ;   popf
39210                                   ;   mov     cx,[ES:BP+DPB.SECTOR_SIZE]
39211                                   ;   shr     cx,1
39212                                   ;   mov     si,bx
39213                                   ;   pop     di ; *
39214                                   ;   pop     es ; **
39215                                   ;   cld
39216                                   ;   rep     movsw
39217                                   ;   push    es
39218                                   ;   pop     ds
39219                                   ;   jmp     short getb19 ; 27/11/2022
39220                                   ;%endif
39221
39222                                   not_in_hma:                                                       ;I001
39223                                       popf                                                          ;I001
39224
39225                                   %else
39226                                       ; 06/03/2024
39227                                       ; PCDOS 7.1 IBMDOS.COM
39228                                       ;;;
39229 000069FE B500                     mov     ch,0
39230 00006A00 368A0E[7900]             mov     cl,[ss:BuffInHMA] ; did we read into scratch buff ?
39231 00006A05 E31C                     jcxz    getb19            ; no
39232                                   ;mov    cx,[es:bp+2]
39233 00006A07 268B4E02                 mov     cx,[es:bp+DPB.SECTOR_SIZE]
39234 00006A0B 89DE                     mov     si,bx
39235 00006A0D 5F                       pop     di
39236 00006A0E 07                       pop     es
39237 00006A0F 9C                       pushf
39238 00006A10 D1E9                     shr     cx,1
39239 00006A12 FC                       cld
39240 00006A13 36803E[6A00]00           cmp     byte [ss:DDMOVE],0
39241                                   ;jz     short getb18+1 ; rep movsw (skip 32bit prefix)
39242 00006A19 7403                     jz      short getb18
39243 00006A1B D1E9                     shr     cx,1
39244                                   ;getb18:
39245                                       ;rep    movsd           ; move the contents of scratch buffer
39246 00006A1D 66                       db      66h     ; 32bit prefix
39247                                   getb18:
39248 00006A1E F3A5                     rep     movsw
39249
39250                                   ;mov    dx,es
39251                                   ;mov    ds,dx
39252 00006A20 06                       push    es
39253 00006A21 1F                       pop     ds
39254 00006A22 9D                       popf                    ; Retrieve possible CY from DREAD
39255                                   ;;;
39256                                   %endif
39257
39258                                   getb19:
39259 00006A23 07                       pop     es ; ***
39260 00006A24 5A                       pop     dx
39261 00006A25 5F                       pop     di
39262 00006A26 5E                       pop     si
39263 00006A27 726C                     JC      short getbx
39264
39265                                   ;   The buffer has the data setup in it (if we were to read)
39266                                   ;   Setup the various buffer fields
39267                                   ;
39268                                   ;   (ds:di) = buffer address
39269                                   ;   (es:bp) = DPB address
39270                                   ;   (HIGH_SECTOR:DX) = sector #
39271                                   ;   (ah) = BUF_FLAGS value
39272                                   ;   (si) = 0 if non fat sector, != 0 if FAT sector read
39273
39274                                   ;hkn; SS override
39275                                   getb20:     ; MSDOS 6.0
39276 00006A29 368B0E[0706]                 MOV     CX,[SS:HIGH_SECTOR]
39277                                       ;mov    [di+8],cx
39278 00006A2E 894D08                     MOV     [DI+BUFFINFO.buf_sector+2],CX
39279                                       ; MSDOS 3.3 (& MSDOS 6.0)
39280                                       ;mov    [di+6],dx
39281 00006A31 895506                     MOV     [DI+BUFFINFO.buf_sector],DX
39282                                       ;;;mov  [di+0Ah],bp  ; MSDOS 3.3
39283                                       ;;mov   [di+0Dh],bp  ; MSDOS 6.0
39284                                       ;mov    [di+0Fh],bp  ; PCDOS 7.1 ; 06/03/2024
39285 00006A34 896D0F                     MOV     [DI+BUFFINFO.buf_DPB],BP
39286                                       ;;;mov  [di+0Ch],es
39287                                       ;;mov   [di+0Fh],es  ; MSDOS 6.0
39288                                       ;mov    [di+11h],es  ; PCDOS 7.1 ; 06/03/2024
39289 00006A37 8C4511                     MOV     [DI+BUFFINFO.buf_DPB+2],ES
39290                                       ; 15/12/2022
39291 00006A3A 268A4600                   mov     al,[es:bp]
39292                                       ;mov    al,[es:bp+0]
39293                                       ; 27/11/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
39294                                       ;MOV    AL,[ES:BP+DPB.DRIVE]
```

```
39295                                   ;mov    [di+4],ax
39296 00006A3E 894504                   MOV     [DI+BUFFINFO.buf_ID],AX             ; Set ID and Flags
39297                                 getb25:
39298                                   ; MSDOS 3.3
39299                                   ;mov     ax,1
39300
39301                                   ; MSDOS 6.0
39302                                   ;mov    byte [di+0Ah],1
39303 00006A41 C6450A01                 MOV     byte [DI+BUFFINFO.buf_wrtcnt],1     ; Default to not a FAT sector ;AC000;
39304 00006A45 31C0                     XOR     AX,AX
39305
39306                                   ; 07/07/2024 (PCDOS 7.1)
39307                                   ;mov    [di+0Dh],ax ; 0
39308 00006A47 89450D                   mov     [di+BUFFINFO.buf_wrtcntinc+2],ax ; 0
39309
39310                                   ; MSDOS 3.3 (& MSDOS 6.0)
39311 00006A4A 09F6                     OR      SI,SI                       ; FAT sector ?
39312 00006A4C 742C                     JZ      short getb30                ; no
39313
39314                                   ; 06/03/2024 (PCDOS 7.1 IBMDOS.COM)
39315                                   ;;;
39316                                   ;;cmp   word [es:bp+0Fh],0
39317                                   ;cmp    word [es:bp+DPB.FAT_SIZE],0
39318 00006A4E 2639460F                 cmp     [es:bp+DPB.FAT_SIZE],ax ; 0
39319 00006A52 751B                     jnz     short getb27   ; not FAT32
39320
39321                                   ; FAT32
39322                                   ;;test  word [es:bp+23h],80h
39323                                   ;test   byte [es:bp+23h],80h
39324 00006A54 26F6462380               test    byte [es:bp+DPB.EXT_FLAGS],80h
39325 00006A59 7507                     jnz     short getb26          ; bit 7 -- 1 means only one FAT is active
39326                                   ;mov    al,[es:bp+8]
39327 00006A5B 268A4608                 mov     al,[es:bp+DPB.FAT_COUNT]
39328                                   ;mov    [di+0Ah],al
39329 00006A5F 88450A                   mov     [di+BUFFINFO.buf_wrtcnt],al
39330                                 getb26:
39331                                   ;mov    ax,[es:bp+33h]
39332 00006A62 268B4633                 mov     ax,[es:bp+DPB.FAT32_SIZE+2]
39333                                   ;mov    [di+0Dh],ax
39334 00006A66 89450D                   mov     [di+BUFFINFO.buf_wrtcntinc+2],ax
39335                                   ;mov    ax,[es:bp+31h]
39336 00006A69 268B4631                 mov     ax,[es:bp+DPB.FAT32_SIZE]
39337 00006A6D EB0B                     jmp     short getb30
39338                                 getb27:
39339                                   ;;;
39340
39341                                   ;mov    al,[es:bp+8]
39342 00006A6F 268A4608                 MOV     AL,[ES:BP+DPB.FAT_COUNT]    ; update number of copies of
39343
39344                                   ; MSDOS 6.0
39345 00006A73 88450A                   MOV     [DI+BUFFINFO.buf_wrtcnt],AL   ;  this sector present on disk
39346                                   ;mov    ax,[es:bp+0Fh]
39347 00006A76 268B460F                 MOV     AX,[ES:BP+DPB.FAT_SIZE]             ; offset of identical FAT
39348                                                                               ;  sectors
39349                                   ; MSDOS 3.3
39350                                   ;;mov   ah,[es:bp+0Fh]
39351                                   ;MOV    AH,[ES:BP+DPB.FAT_SIZE]
39352
39353                                 ;   BUGBUG - dos 6 can clean this up by not setting wrtcntinc unless wrtcnt
39354                                 ;           is set
39355
39356                                 getb30:
39357                                   ; MSDOS 6.0
39358                                   ;mov    [di+0Bh],ax
39359 00006A7A 89450B                   MOV     [DI+BUFFINFO.buf_wrtcntinc],AX
39360
39361                                   ; MSDOS 3.3
39362                                   ;;mov   [di+8],ax ; 15/08/2018
39363                                   ;MOV    [DI+BUFFINFO.buf_wrtcnt],AX
39364
39365 00006A7D E852FE                   CALL    PLACEBUF
39366
39367                                 ;hkn; SS override for next 4
39368                                 getb35:
39369                                   ; 17/12/2022
39370                                   ; 07/12/2022 MSDOS 5.0 MSDOS.SYS compatibility)
39371                                   ; MSDOS 3.3 & MSDOS 5.0 & MSDOS 6.0
39372                                   ;MOV    [SS:CURBUF+2],DS
39373                                   ;MOV    [SS:LastBuffer+2],DS
39374                                   ;MOV    [SS:CURBUF],DI
39375                                   ;MOV    [SS:LastBuffer],DI
39376                                   ;CLC
39377
39378                                   ; 17/12/2022
39379                                   ; 07/12/2022
39380                                   ; Retro DOS v4.0
39381 00006A80 368C1E[2000]             mov     [ss:LastBuffer+2],ds
39382 00006A85 36893E[1E00]             mov     [ss:LastBuffer],di
39383 00006A8A F8                       clc
39384                                 getb35x: ; 28/07/2019
39385 00006A8B 368C1E[E405]             MOV     [ss:CURBUF+2],ds
39386 00006A90 36893E[E205]             MOV     [ss:CURBUF],di
39387
39388                                 ;   Return with 'C' set appropriately
39389                                 ;   (dx) = caller's original value
39390
39391                                 getbx:
39392 00006A95 16                       push    ss
39393 00006A96 1F                       pop     ds
39394                                   ;retn
39395                                   ; 27/11/2022 MSDOS 5.0 MSDOS.SYS compatibility)
39396                                 getbuffrb_retn:
39397                                 ;flushbuf_retn:    ; 17/12/2022
39398 00006A97 C3                       retn
39399
39400                                 ;Break      <FLUSHBUF -- WRITE OUT DIRTY BUFFERS>
39401                                 ;--------------------------------------------------------------------------
39402                                 ; Input:
39403                                 ;   DS = DOSGROUP
39404                                 ;   AL = Physical unit number local buffers only
39405                                 ;      = -1 for all units and all remote buffers
39406                                 ; Function:
39407                                 ;   Write out all dirty buffers for unit, and flag them as clean
39408                                 ;   Carry set if error (user FAILed to I 24)
39409                                 ;       Flush operation completed.
39410                                 ; DS Preserved, all others destroyed (ES too)
39411                                 ;--------------------------------------------------------------------------
39412
39413                                   ; 20/05/2019 - Retro DOS v4.0
39414                                   ; DOSCODE:9A35h (MSDOS 6.21, MSDOS.SYS)
39415
39416                                   ; 27/11/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
39417                                   ; DOSCODE:99DAh (MSDOS 5.0, MSDOS.SYS)
39418
```

```
                                    ; 06/03/2024 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMDOS.COM)
                                    ; PCDOS 7.1 IBMDOS.COM - DOSCODE:0AC2Bh

                                FLUSHBUF:
                                    ; MSDOS 3.3
                                    ;;mov   ah,-1 ; 01/08/2018 - Retro DOS v3.0
                                    ;lds    di,[BUFFHEAD]

                                    ; 06/03/2024 (PCDOS 7.1 IBMDOS.COM)
                                    ;;;
00006A98 3CFF                       cmp     al,0FFh         ; -1
00006A9A 750C                       jne     short flshbuf_2
00006A9C BB1A00                     mov     bx,26
                                flshbuf_1:
                                    ;and    ss:drv_flags_1[bx],0F7h
00006A9F 3680A7[0713]F7             and     byte [ss:bx+drive_flags-1],0F7h ; clear bit 3
00006AA5 4B                         dec     bx              ; backward
00006AA6 75F7                       jnz     short flshbuf_1
                                flshbuf_2:
                                    ;;;

                                    ; MSDOS 6.0
00006AA8 E80EFE                     call    GETCURHEAD
                                    ;TEST   word [ss:DOS34_FLAG],FROM_DISK_RESET ; from disk reset ? ;hkn;
00006AAB 36F606[1106]04             TEST    byte [ss:DOS34_FLAG],FROM_DISK_RESET ; 4
00006AB1 7508                       jnz     short scan_buf_queue
00006AB3 36833E[7100]00             cmp     word [ss:DirtyBufferCount],0                ;hkn;
00006AB9 7423                       je      short end_scan

                                scan_buf_queue:
00006ABB E82900                     call    CHECKFLUSH
                                    ;push   ax  ; MSDOS 3.3
                                    ; MSDOS 6.0
                                    ;mov    ah,[di+4]
00006ABE 8A6504                     mov     ah,[DI+BUFFINFO.buf_ID]
00006AC1 363826[2203]               cmp     [SS:WPERR],ah                               ;hkn;
00006AC6 7408                       je      short free_the_buf
                                    ;TEST   word [ss:DOS34_FLAG],FROM_DISK_RESET ; from disk reset ? ;hkn;
00006AC8 36F606[1106]04             TEST    byte [ss:DOS34_FLAG],FROM_DISK_RESET ; 4
00006ACE 7405                       jz      short dont_free_the_buf
                                    ; MSDOS 3.3
                                    ;;mov   al,[di+4]
                                    ;mov    al,[DI+BUFFINFO.buf_ID]
                                    ;cmp    [SS:WPERR],al                               ;hkn;
                                    ; 15/08/2018
                                    ;jne    short dont_free_the_buf
                                free_the_buf:
                                    ; MSDOS 6.0 (& MSDOS 3.3)
00006AD0 C74504FF00                 mov     word [DI+BUFFINFO.buf_ID],00FFh
                                dont_free_the_buf:
                                    ;pop    ax  ; MSDOS 3.3

                                    ; MSDOS 3.3
                                    ;mov    di,[DI]
                                    ;;mov   di,[DI+BUFFINFO.buf_link] ; .buf_next
                                    ;
                                    ; 15/08/2018
                                    ;lds    di,[di]
                                    ;
                                    ;cmp    di,-1 ; 0FFFFh
                                    ;jnz    short scan_buf_queue

                                    ; MSDOS 6.0
00006AD5 8B3D                       mov     di,[di]
                                    ;mov    di,[DI+BUFFINFO.buf_next] ; .buf_link
00006AD7 363B3E[7E11]               cmp     di,[SS:FIRST_BUFF_ADDR]                     ;hkn;
00006ADC 75DD                       jne     short scan_buf_queue

                                end_scan:
00006ADE 16                         push    ss
00006ADF 1F                         pop     ds
                                    ; 01/08/2018 - Retro DOS v3.0
                                    ;cmp    byte [FAILERR],0
                                    ;jne    short bad_flush
                                    ;retn
                                ;bad_flush:
                                    ;stc
                                    ;retn

                                    ; 17/12/2022
                                    ; 27/11/2022 MSDOS 5.0 MSDOS.SYS compatibility)
                                    ; 01/08/2018 - Retro DOS v3.0
00006AE0 803E[4A03]01               cmp     byte [FAILERR],1
00006AE5 F5                         cmc
                                flushbuf_retn:
00006AE6 C3                         retn

                                    ; 17/12/2022
                                    ; 27/11/2022 MSDOS 5.0 MSDOS.SYS compatibility)
                                    ;cmp    byte [FAILERR],0
                                    ;jne    short bad_flush
                                    ;retn
                                ;bad_flush:
                                    ;stc
                                    ;retn

                                ;--------------------------------------------------------------------------
                                ;
                                ; Procedure Name : CHECKFLUSH
                                ;
                                ; Inputs : AL - Drive number, -1 means do not check for drive
                                ;      DS:DI - pointer to buffer
                                ;
                                ; Function : Write out a buffer if it is dirty
                                ;
                                ; Carry set if problem (currently user FAILed to I 24)
                                ;
                                ;--------------------------------------------------------------------------

                                    ; 07/03/2024 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMDOS.COM)

                                CHECKFLUSH:
                                    ; MSDOS 6.0
00006AE7 B4FF                       mov     ah,-1   ; 01/08/2018 Retro DOS v3.0
                                    ;cmp    [di+4],ah
00006AE9 386504                     CMP     [DI+BUFFINFO.buf_ID],AH
00006AEC 74F8                       jz      short flushbuf_retn   ; Skip free buffer, carry clear
00006AEE 38C4                       CMP     AH,AL                 ;
00006AF0 7412                       JZ      short DOBUFFER        ; do this buffer
                                    ;cmp    al,[di+4]
00006AF2 3A4504                     CMP     AL,[DI+BUFFINFO.buf_ID]
00006AF5 F8                         CLC
00006AF6 75EE                       jnz     short flushbuf_retn   ; Buffer not for this unit or SFT
```

```
                                    ; 07/03/2024 (PCDOS 7.1 IBMDOS.COM)
                                    ;;;
00006AF8 31DB                       xor     bx,bx
00006AFA 88C3                       mov     bl,al
                                    ;test   ss:drive_flags[bx],8
00006AFC 36F687[0813]08             test    byte [ss:bx+drive_flags],8 ; bit 3
00006B02 75E2                       jnz     short flushbuf_retn
                                    ;;;

                                DOBUFFER:
                                    ;test   byte [di+5],40h
00006B04 F6450540                   TEST    byte [DI+BUFFINFO.buf_flags],buf_dirty
00006B08 74DC                       jz      short flushbuf_retn   ; Buffer not dirty, carry clear by TEST
00006B0A 50                         PUSH    AX
                                    ;push   word [di+4]
00006B0B FF7504                     PUSH    WORD [DI+BUFFINFO.buf_ID]
00006B0E E80B00                     CALL    BUFWRITE
00006B11 58                         POP     AX
00006B12 7206                       JC      short LEAVE_BUF               ; Leave buffer marked free (lost).
                                    ;and    ah,0BFh
00006B14 80E4BF                     AND     AH,~buf_dirty         ; Buffer is clean, clears carry
                                    ;mov    [di+4],ax
00006B17 894504                     MOV     [DI+BUFFINFO.buf_ID],AX
                                LEAVE_BUF:
00006B1A 58                         POP     AX                    ; Search info
                                checkflush_retn:
00006B1B C3                         retn

                                ;Break      <BUFWRITE -- WRITE OUT A BUFFER IF DIRTY>
                                ;-----------------------------------------------------------------------------
                                ;
                                ;   BufWrite writes a buffer to the disk, if it's dirty.
                                ;
                                ;   ENTRY   DS:DI Points to the buffer
                                ;
                                ;   EXIT    Buffer marked free
                                ;           Carry set if error (currently user FAILed to I 24)
                                ;
                                ;   USES    All buf DS:DI
                                ;           HIGH_SECTOR
                                ;-----------------------------------------------------------------------------

                                    ; 20/05/2019 - Retro DOS v4.0
                                    ; DOSCODE:9AA0h (MSDOS 6.21, MSDOS.SYS)

                                    ; 27/11/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
                                    ; DOSCODE:9A45h (MSDOS 5.0, MSDOS.SYS)

                                    ; 07/03/2024 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMDOS.COM)
                                    ; PCDOS 7.1 IBMDOS.COM - DOSCODE:0ACB1h

                                BUFWRITE:
                                    ; 10/09/2018
                                    ; 01/08/2018 - Retro DOS v3.0
                                    ; IBMDOS.COM (MSDOS 3.3, 1987) - Offset 5E94h
00006B1C B8FF00                     MOV     AX,00FFH
                                    ;xchg   ax,[di+4]
00006B1F 874504                     XCHG    AX,[DI+BUFFINFO.buf_ID]       ; Free, in case write barfs
00006B22 3CFF                       CMP     AL,0FFH
00006B24 74F5                       jz       short checkflush_retn ; Buffer is free, carry clear.
                                    ;test   ah,40h
00006B26 F6C440                     test    AH,buf_dirty
00006B29 74F0                       jz       short checkflush_retn ; Buffer is clean, carry clear.
                                    ; MSDOS 6.0
00006B2B E8A200                     call    DEC_DIRTY_COUNT       ; LB. decrement dirty count

                                ;hkn; SS override
00006B2E 363A06[2203]               CMP     AL,[SS:WPERR]
00006B33 74E6                       jz       short checkflush_retn ; If in WP error zap buffer

                                ; 07/03/2024 (PCDOS 7.1 IBMDOS.COM)
                                %if 0
                                ;hkn; SS override
                                    ; MSDOS 6.0
                                    MOV     [SS:SC_DRIVE],AL      ;LB. set it for invalidation ;AN000;
                                %endif
                                    ; 07/03/2024
                                    ;;;;les bp,[di+10] ; MSDOS 3.3
                                    ;;;les  bp,[di+13] ; MSDOS 6.0
                                    ;;les   bp,[di+15] ; PCDOS 7.1 ; 07/03/2024
                                    ;LES    BP,[DI+BUFFINFO.buf_DPB]

                                    ;;;lea  bx,[di+16]
                                    ;;lea   bx,[di+20] ; MSDOS 6.0
                                    ;lea    bx,[di+24] ; PCDOS 7.1 ; 07/03/2024
00006B35 8D5D18                     LEA     BX,[DI+BUFINSIZ]      ; Point at buffer

                                ; 07/03/2024
                                %if 0
                                    ;mov    dx,[di+6]
                                    MOV     DX,[DI+BUFFINFO.buf_sector] ;F.C. >32mb             ;AN000;

                                    ; MSDOS 6.0
                                    ;mov    cx,[di+8]
                                    MOV     CX,[DI+BUFFINFO.buf_sector+2] ;F.C. >32mb   ;AN000;

                                ;hkn; SS override
                                    MOV     [SS:HIGH_SECTOR],CX   ;F.C. >32mb           ;AN000;
                                %else
                                    ; 07/03/2024 (PCDOS 7.1 IBMDOS.COM)
                                    ;;;
                                    ;les    dx,[di+6]
00006B38 C45506                     les     dx,[di+BUFFINFO.buf_sector]
00006B3B 368C06[0706]               mov     [ss:HIGH_SECTOR],es

                                    ;;;les  bp,[di+10] ; MSDOS 3.3
                                    ;;les   bp,[di+13] ; MSDOS 6.0
                                    ;les    bp,[di+15] ; PCDOS 7.1 ; 07/03/2024
00006B40 C46D0F                     les     bp,[di+BUFFINFO.buf_DPB]
                                    ;;;
                                %endif
                                    ;mov    cl,[di+10] ; MSDOS 6.0 & PCDOS 7.1
00006B43 8A4D0A                     MOV     CL,[DI+BUFFINFO.buf_wrtcnt] ;>32mb          ;AC000;
                                    ; MSDOS 3.3
                                    ;; mov  cx,[DI+8]
                                    ;mov    cx,[DI+BUFFINFO.buf_wrtcnt]
                                    ;MOV    AL,CH                 ; [DI+BUFFINFO.buf_wrtcntinc]
00006B46 30ED                       XOR     CH,CH
                                    ;;mov   ah,ch ; MSDOS 3.3

                                ;hkn; SS override for ALLOWED
                                    ;mov    byte [SS:ALLOWED],18h
00006B48 36C606[4B03]18             MOV     byte [SS:ALLOWED],Allowed_RETRY+Allowed_FAIL
                                    ;test   byte [di+5],8
```

```
                                    ; MSDOS 6.0 (& Retro DOS 3.0)
                                    ;test  ah,8
00006B4E F6C408                     test   AH,buf_isDATA
00006B51 7406                       JZ     short NO_IGNORE
                                    ;or    byte [SS:ALLOWED],20h
00006B53 36800E[4B03]20             OR     byte [SS:ALLOWED],Allowed_IGNORE
                                NO_IGNORE:
                                    ;xor   ah,ah ; 10/09/2018 (MSDOS 3.3, Retro DOS v3.0)
                                    ; MSDOS 6.0
                                    ;mov   ax,[di+11]
00006B59 8B450B                     MOV    AX,[DI+BUFFINFO.buf_wrtcntinc]       ;>32mb         ;AC000;

                                    ; 07/03/2024 (PCDOS 7.1 IBMDOS.COM)
                                    ;;;
                                    ;mov   si,[di+13]
00006B5C 8B750D                     mov    si,[di+BUFFINFO.buf_wrtcntinc+2]
                                    ;;;

00006B5F 57                         PUSH   DI             ; Save buffer pointer
00006B60 31FF                       XOR    DI,DI          ; Indicate failure

00006B62 1E                         push   ds ; *
00006B63 53                         push   bx ; **
                                WRTAGAIN:
                                    ; 07/03/2024 (PCDOS 7.1 IBMDOS.COM)
                                    ;;;
00006B64 56                         push   si             ; SI:AX = FAT size (in sectors)
00006B65 36FF36[0706]               push   word [ss:HIGH_SECTOR]
                                    ;;;
00006B6A 57                         push   di ; ***
00006B6B 51                         push   cx ; ****
00006B6C 50                         push   ax ; *****
                                    ;MOV   CX,1
                                    ; 17/12/2022
                                    ; ch = 0
00006B6D B101                       mov    cl,1 ; 24/07/2019
                                    ; 27/11/2022 MSDOS 5.0 MSDOS.SYS compatibility)
                                    ;mov   cx,1
00006B6F 53                         push   bx ; ******
00006B70 52                         push   dx ; *******
00006B71 1E                         push   ds ; ********

                                ; Note: As far as I can tell, all disk reads into buffers go through this point. -mrw 10/88

                                    ; MSDOS 6.0
                                    ;cmp   byte [ss:BuffInHMA],0 ; 10/06/2019
                                    ; 22/09/2023
00006B72 36382E[7900]               cmp    [ss:BuffInHMA],ch ; 0
00006B77 7423                       jz     short NBUFFINHMA
00006B79 51                         push   cx
00006B7A 06                         push   es
00006B7B 89DE                       mov    si,bx
00006B7D 268B4E02                   mov    cx,[es:bp+DPB.SECTOR_SIZE]
00006B81 D1E9                       shr    cx,1
00006B83 36C43E[7A00]               les    di,[ss:LoMemBuff] ; 10/06/2019
00006B88 89FB                       mov    bx,di
00006B8A FC                         cld
                                    ;rep   movsw
                                    ; 07/03/2024 - Retro DOS v5.0
                                    ; (PCDOS 7.1 IBMDOS.COM)
                                    ;;;
00006B8B 36803E[6A00]00             cmp    byte [ss:DDMOVE],0
00006B91 7403                       jz      short bufwr_movsw ; skip 32bit prefix
00006B93 D1E9                       shr     cx,1
00006B95 66                         db     66h    ; 32bit op prefix (rep movsd)
                                bufwr_movsw:
00006B96 F3A5                       rep    movsw
                                    ;;;
00006B98 06                         push   es
00006B99 1F                         pop    ds
00006B9A 07                         pop    es
00006B9B 59                         pop    cx
                                NBUFFINHMA:
00006B9C E8B4D4                     call   DWRITE         ; Write out the dirty buffer
00006B9F 1F                         pop    ds ; ********
00006BA0 5A                         pop    dx ; *******
00006BA1 5B                         pop    bx ; ******
00006BA2 58                         pop    ax ; *****
00006BA3 59                         pop    cx ; ****
00006BA4 5F                         pop    di ; ***
                                    ; 07/03/2024 (PCDOS 7.1 IBMDOS.COM)
                                    ;;;
00006BA5 368F06[0706]               pop    word [ss:HIGH_SECTOR]
00006BAA 5E                         pop    si
                                    ;;;
00006BAB 7201                       JC     short NOSET
                                    ; 05/07/2024 (PCDOS 7.1)
                                    ;mov   di,1
00006BAD 47                         INC    DI             ; If at least ONE write succeedes,
                                NOSET:                         ;    the operation succeedes.
00006BAE 01C2                       ADD    DX,AX
                                    ; 07/03/2024 (PCDOS 7.1 IBMDOS.COM)
                                    ;;;
00006BB0 361136[0706]               adc    [ss:HIGH_SECTOR],si
                                    ;;;
00006BB5 E2AD                       LOOP   WRTAGAIN
00006BB7 5B                         pop    bx ; **
00006BB8 1F                         pop    ds ; *
                                    ;OR    DI,DI          ; Clears carry
                                    ;JNZ   short BWROK    ; At least one write worked
                                    ;STC                  ; DI never got INCed, all writes failed.
                                    ; 05/07/2024 (PCDOS 7.1)
                                    ;sub   di,1
                                    ; 22/09/2023
00006BB9 83FF01                     cmp    di,1
                                BWROK:
00006BBC 5F                         POP    DI
00006BBD C3                         retn

                                ; 07/03/2024 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMDOS.COM)
                                %if 0
                                ; NOTE: Secondary Buffer Cache is not used in PCDOS 7.1 IBMDOS.COM.

                                ;** Set_RQ_SC_Parms - Set Secondary Cache Parameters
                                ;-----------------------------------------------------------------------------
                                ;   Set_RQ_SC_Parms sets the sector size and drive number value
                                ;   for the secondary cache. This updates SC_SECTOR_SIZE &
                                ;   SC_DRIVE even if SC is disabled to save the testing
                                ;   code and time
                                ;
                                ;   ENTRY  ES:BP = drive parameter block
                                ;
                                ;   EXIT   [SC_SECTOR_SIZE]= drive sector size
                                ;          [SC_DRIVE]= drive #
```

```
                                ;
                                ;   USES    Flags
                                ;--------------------------------------------------------------------------------

                                ; 27/11/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
                                ; 04/05/2019 - Retro DOS v4.0

                                SET_RQ_SC_PARMS:
                                ;hkn; SS override for all variables used in this procedure.
                                    push    ax
                                    ;mov    ax,[es:bp+2]
                                    MOV     ax,[ES:BP+DPB.SECTOR_SIZE]    ; save sector size
                                    MOV     [ss:SC_SECTOR_SIZE],ax
                                    ;;mov   al,[es:bp+0]
                                    ; 27/11/2022 MSDOS 5.0 MSDOS.SYS compatibility)
                                    ;MOV    al,[ES:BP+DPB.DRIVE]          ; save drive #
                                    ; 15/12/2022
                                    mov     al,[ES:BP]
                                    MOV     [ss:SC_DRIVE],al
                                    pop     ax
                                srspx:
                                    retn                                  ;LB. return

                                %endif

                                ; 07/03/2024 -Retro DOS v5.0
                                ; (PCDOS 7.1 IBMDOS.COM - DOSCODE:0AD57h)
                                %if 0
                                null_sub:
                                SET_RQ_SC_PARMS:
                                    retn
                                %endif

                                ; 01/02/2024 - Retro DOS v5.0
                                ;--------------------------------------------------------------------------------
                                ; PCDOS 7.1 IBMDOS.COM - DOSCODE:AD58h

                                SET_BUF_DIRTY:                            ; ...
00006BBE 26F6450540                 ;test   byte [es:di+5],40h
                                    test    byte [es:di+BUFFINFO.buf_flags],buf_dirty
00006BC3 750A                       jnz     short yesdirty2
                                    ;or     byte [es:di+5],40h
00006BC5 26804D0540                 or      byte [es:di+BUFFINFO.buf_flags],buf_dirty

                                ;INC_DIRTY_COUNT:
                                ;   inc     word [ss:DirtyBufferCount]
                                ;yesdirty2:
                                ;   retn

                                ;Break      <INC_DIRTY_COUNT-increment dirty count>
                                ;--------------------------------------------------------------------------------
                                ; Input:
                                ;   none
                                ; Function:
                                ;   increment dirty buffers count
                                ; Output:
                                ;   dirty buffers count is incremented
                                ;
                                ; All registers preserved
                                ;--------------------------------------------------------------------------------

                                INC_DIRTY_COUNT:
                                ;; BUGBUG  ---- remove this routine
                                ;; BUGBUG ---- only one instruction is needed (speed win, space loose)
00006BCA 36FF06[7100]               inc     word [ss:DirtyBufferCount]    ;hkn;
                                yesdirty2:  ; 01/02/2024
00006BCF C3                         retn

                                ;Break      <DEC_DIRTY_COUNT-decrement dirty count>
                                ;--------------------------------------------------------------------------------
                                ; Input:
                                ;   none
                                ; Function:
                                ;   decrement dirty buffers count
                                ; Output:
                                ;   dirty buffers count is decremented
                                ;
                                ; All registers preserved
                                ;--------------------------------------------------------------------------------

                                DEC_DIRTY_COUNT:
00006BD0 36833E[7100]00             cmp     word [ss:DirtyBufferCount],0 ;hkn;
00006BD6 7405                       jz      short ddcx            ; BUGBUG - shouldn't it be an
00006BD8 36FF0E[7100]               dec     word [ss:DirtyBufferCount]
                                                                  ; error condition to underflow here? ;hkn;
                                ddcx:
00006BDD C3                         retn

                                ;=============================================================================
                                ; MSPROC.ASM, MSDOS 6.0, 1992
                                ;=============================================================================
                                ; 02/08/2018 - Retro DOS v3.0
                                ; 29/04/2019 - Retro DOS v4.0
                                ; 07/03/2024 - Retro DOS v5.0

                                ; (15/04/2018 - RetrO DOS v2.0, MSDOS 2.11 - PROC.ASM - 1983)

                                ; Pseudo EXEC system call for DOS

                                ;   TITLE   MSPROC - process maintenance
                                ;   NAME    MSPROC

                                ; ============================================================================
                                ;** Process related system calls and low level routines for DOS 2.X.
                                ;   I/O specs are defined in DISPATCH.
                                ;
                                ;   $WAIT
                                ;   $EXEC
                                ;   $Keep_process
                                ;   Stay_resident
                                ;   $EXIT
                                ;   $ABORT
                                ;   abort_inner
                                ;
                                ;   Modification history:
                                ;
                                ;           Created: ARR 30 March 1983
                                ;           AN000   version 4.0 jan. 1988
                                ;           A007    PTM 3957 - fake vesrion for IBMCACHE.COM
                                ;           A008    PTM 4070 - fake version for MS WINDOWS
                                ;
                                ;           M000    added support for loading programs into UMBs 7/9/90
                                ;
                                ;           M004 - MS PASCAL 3.2 support. Please see under tag M003 in
```

```
                                ;               dossym.inc. 7/30/90
                                ;       M005 - Support for EXE programs with out STACK segment and
                                ;              with resident size < 64K - 256 bytes. A 256 byte
                                ;              stack is provided at the end of the program. Note that
                                ;              only SP is changed.
                                ;       M020 - Fix for Rational bug for details see exepatch.asm
                                ;
                                ;       M028 - 4b04 implementation
                                ;
                                ;       M029 - Support for EXEs without stack rewritten. If EXE is
                                ;              in memory block >= 64K, sp = 0. If memory block
                                ;              obtained is <64K, point sp at the end of the memory
                                ;              block. For EXEs smaller than 64K, 256 bytes are still
                                ;              added for a stack segment which may be needed if it
                                ;              is loaded in low memory situations.
                                ;
                                ;       M030 - Fixing bug in EXEPACPATCH & changing 4b04 to 4b05
                                ;
                                ;       M040 - Bug #3052. The environment sizing code would flag a
                                ;              a bad environment if it reached 32767 bytes. Changed
                                ;              to allow 32768 bytes of environment.
                                ;
                                ;       M047 - Release the allocated UMB when we failed to load a
                                ;              COM file high. Also ensure that if the biggest block
                                ;              into which we load the com file is less than 64K then
                                ;              we provide atleast 256 bytes of stack to the user.
                                ;
                                ;       M050 - Made Lie table search CASE insensitive
                                ;
                                ;       M060 - Removed special version table from the kernal and
                                ;                   put it in a device drive which puts the address
                                ;                   in the DOS DATA area location UU_IFS_DOS_CALL
                                ;              as a DWORD.
                                ;
                                ;       M063 - Modified UMB support. If the HIGH_ONLY bit is set on
                                ;              entry do not try to load low if there is no space in
                                ;              UMBs.
                                ;
                                ;       M068 - Support for copy protect apps. Call ChkCopyProt to
                                ;              set a20off_count. Set bit EXECA20BIT in DOS_FLAG. Also
                                ;              change return address to LeaveDos if AL=5.
                                ;
                                ;           20-Jul-1992 bens  Added ifdef RESTRICTED_BUILD code that
                                ;                 controls building a version of MSDOS.SYS that only
                                ;                 runs programs from a fixed list (defined in the
                                ;                 file RESTRICT.INC).  Search for "RESTRICTED_BUILD"
                                ;                 for details.  This feature is used to build a
                                ;                 "special" version of DOS that can be handed out to
                                ;                 OEM/ISV customers as part of a "service" disk.
                                ;
                                ; ============================================================================

                                ;SAVEXIT  EQU  10

                                ;BREAK      <$WAIT - return previous process error code>
                                ; ============================================================================
                                ;   $WAIT - Return previous process error code.
                                ;
                                ;   Assembler usage:
                                ;
                                ;       MOV     AH, WaitProcess
                                ;       INT     int_command
                                ;
                                ;   ENTRY   none
                                ;   EXIT    (ax) = exit code
                                ;   USES    all
                                ; ============================================================================

                                    ; 20/05/2019 - Retro DOS v4.0
                                    ; DOSCODE:9B55h (MSDOS 6.21, MSDOS.SYS)

                                    ; 27/11/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
                                    ; DOSCODE:9A5Ah (MSDOS 5.0, MSDOS.SYS)

                                    ; 07/03/2024 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMDOS.COM)
                                    ; PCDOS 7.1 IBMDOS.COM - DOSCODE:0AD78h

                                    ; (MSDOS 6.22 MSDOS.SYS - DOSCODE:9B55h)
                                    ; (Windows ME IO.SYS - BIOSCODE:944Ch)

                                _$WAIT:
                                    ; 02/08/2018 - Retro DOS v3.0
                                    ; IBMDOS.COM (MSDOS 3.3, 1987) - Offset 5E1h

00006BDE 31C0                       xor     AX,AX
00006BE0 368706[3403]               xchg    AX,[ss:exit_code]
00006BE5 E9869A                     jmp     SYS_RET_OK

                                ; ============================================================================
                                ;BREAK <$exec - load/go a program>
                                ;   EXEC.ASM - EXEC System Call
                                ;
                                ;
                                ; Assembler usage:
                                ;       lds     DX, Name
                                ;       les     BX, Blk
                                ;       mov     AH, Exec
                                ;       mov     AL, FUNC
                                ;       int     INT_COMMAND
                                ;
                                ;   AL  Function
                                ;   --  --------
                                ;    0  Load and execute the program.
                                ;    1  Load, create the program header but do not
                                ;       begin execution.
                                ;    3  Load overlay. No header created.
                                ;
                                ;       AL = 0 -> load/execute program
                                ;
                                ;       +--------------------------+
                                ;       | WORD segment address of  |
                                ;       | environment.             |
                                ;       +--------------------------+
                                ;       | DWORD pointer to ASCIZ   |
                                ;       | command line at 80h      |
                                ;       +--------------------------+
                                ;       | DWORD pointer to default       |
                                ;       | FCB to be passed at 5Ch  |
                                ;       +--------------------------+
                                ;       | DWORD pointer to default       |
                                ;       | FCB to be passed at 6Ch  |
                                ;       +--------------------------+
                                ;
                                ;       AL = 1 -> load program
```

```
40039                                   ;
40040                                   ;       +---------------------------+
40041                                   ;       | WORD segment address of   |
40042                                   ;       | environment.              |
40043                                   ;       +---------------------------+
40044                                   ;       | DWORD pointer to ASCIZ    |
40045                                   ;       | command line at 80h       |
40046                                   ;       +---------------------------+
40047                                   ;       | DWORD pointer to default      |
40048                                   ;       | FCB to be passed at 5Ch   |
40049                                   ;       +---------------------------+
40050                                   ;       | DWORD pointer to default      |
40051                                   ;       | FCB to be passed at 6Ch   |
40052                                   ;       +---------------------------+
40053                                   ;       | DWORD returned value of   |
40054                                   ;       | CS:IP                     |
40055                                   ;       +---------------------------+
40056                                   ;       | DWORD returned value of   |
40057                                   ;       | SS:IP                     |
40058                                   ;       +---------------------------+
40059                                   ;
40060                                   ;       AL = 3 -> load overlay
40061                                   ;
40062                                   ;       +---------------------------+
40063                                   ;       | WORD segment address where|
40064                                   ;       | file will be loaded.      |
40065                                   ;       +---------------------------+
40066                                   ;       | WORD relocation factor to |
40067                                   ;       | be applied to the image.      |
40068                                   ;       +---------------------------+
40069                                   ;
40070                                   ; Returns:
40071                                   ;       AX = error_invalid_function
40072                                   ;          = error_bad_format
40073                                   ;          = error_bad_environment
40074                                   ;          = error_not_enough_memory
40075                                   ;          = error_file_not_found
40076                                   ; ============================================================================
40077                                   ;
40078                                   ;   Revision history:
40079                                   ;
40080                                   ;    A000  version 4.00  Jan. 1988
40081                                   ;
40082                                   ; ============================================================================
40083
40084                                   Exec_Internal_Buffer             EQU    OPENBUF
40085                                   Exec_Internal_Buffer_Size  EQU    (128+128+53+curdirLen)
40086
40087                                   ; ============================================================================
40088
40089                                   ;IF1        ; warning message on buffers
40090                                   ;%out       Please make sure that the following are contiguous and of the
40091                                   ;%out       following sizes:
40092                                   ;%out
40093                                   ;%out       OpenBuf     128
40094                                   ;%out       RenBuf     128
40095                                   ;%out       SearchBuf    53
40096                                   ;%out       DummyCDS    curdirLen
40097                                   ;ENDIF
40098
40099                                   ; ============================================================================
40100
40101                                   ; ============================================================================
40102                                   ;
40103                                   ; ============================================================================
40104
40105                                       ; 20/05/2019 - Retro DOS v4.0
40106                                       ; DOSCODE:9B5Fh (MSDOS 6.21, MSDOS.SYS)
40107
40108                                       ; 30/11/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
40109                                       ; DOSCODE:9B04h (MSDOS 5.0, MSDOS.SYS)
40110
40111                                       ; 08/03/2024
40112                                       ; 07/03/2024 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMDOS.COM)
40113                                       ; PCDOS 7.1 IBMDOS.COM - DOSCODE:0AD82h
40114
40115                                       ; (MSDOS 6.22 MSDOS.SYS - DOSCODE:9B5Fh)
40116                                       ; (Windows ME IO.SYS - BIOSCODE:946Fh)
40117
40118                                   _$EXEC:
40119                                       ; 02/08/2018 - Retro DOS v3.0
40120                                       ; IBMDOS.COM (MSDOS 3.3, 1987) - Offset 5EF1h
40121
40122                                   EXEC001S:
40123                                       ;LocalVar    Exec_Blk         ,DWORD
40124                                       ;LocalVar    Exec_Func        ,BYTE
40125                                       ;LocalVar    Exec_Load_High   ,BYTE
40126                                       ;LocalVar    Exec_FH          ,WORD
40127                                       ;LocalVar    Exec_Rel_Fac     ,WORD
40128                                       ;LocalVar    Exec_Res_Len_Para       ,WORD
40129                                       ;LocalVar    Exec_Environ     ,WORD
40130                                       ;LocalVar    Exec_Size        ,WORD
40131                                       ;LocalVar    Exec_Load_Block  ,WORD
40132                                       ;LocalVar    Exec_DMA         ,WORD
40133                                       ;LocalVar    ExecNameLen      ,WORD
40134                                       ;LocalVar    ExecName         ,DWORD
40135                                       ;
40136                                       ;LocalVar    Exec_DMA_Save    ,WORD
40137                                       ;LocalVar    Exec_NoStack     ,BYTE
40138
40139                                       ; MSDOS 3.3 (& MSDOS 6.0)
40140                                       ;%define       Exec_Blk       dword [bp-4]
40141                                       %define        Exec_Blk       [bp-4] ; 09/08/2018
40142                                       %define        Exec_BlkL      word [bp-4]
40143                                       %define        Exec_BlkH      word [bp-2]
40144                                       %define        Exec_Func      byte [bp-5]
40145                                       %define        Exec_Load_High byte [bp-6]
40146                                       %define        Exec_FH        word [bp-8]
40147                                       %define        Exec_Rel_Fac   word [bp-10]
40148                                       %define        Exec_Res_Len_Para word [bp-12]
40149                                       %define        Exec_Environ   word [bp-14]
40150                                       %define        Exec_Size      word [bp-16]
40151                                       %define        Exec_Load_Block       word [bp-18]
40152                                       %define        Exec_DMA       word [bp-20]
40153                                       %define        ExecNameLen    word [bp-22]
40154                                       ;%define       ExecName       dword [bp-26]
40155                                       %define        ExecName       [bp-26] ; 09/08/2018
40156                                       %define        ExecNameL      word [bp-26]
40157                                       %define        ExecNameH      word [bp-24]
40158                                       ; MSDOS 6.0
40159                                       %define        Exec_DMA_Save  word [bp-28]
40160                                       %define        Exec_NoStack   byte [bp-29]
40161
40162                                       ; ================================================================
```

```
                                    ; validate function
                                    ; ==================================================================

                                    ; M068 - Start
                                    ;
                                    ; Reset the A20OFF_COUNT to 0. This is done as there is a
                                    ; possibility that the count may not be decremented all the way to
                                    ; 0. A typical case is if the program for which we intended to keep
                                    ; the A20 off for a sufficiently long time (A20OFF_COUNT int 21
                                    ; calls), exits pre-maturely due to error conditions.

                                    ; MSDOS 6.0
00006BE8 36C606[8500]00             mov     byte [SS:A20OFF_COUNT], 0

                                    ; If al=5 (ExecReady) we'll change the return address on the stack
                                    ; to be LeaveDos in msdisp.asm. This ensures that the EXECA20OFF
                                    ; bit set in DOS_FLAG by ExecReady is not cleared in msdisp.asm

00006BEE 3C05                       cmp     al,5                    ; Q: is this ExecReady call
                                    ;jne    short @f
00006BF0 7505                       jne     short Exec_@f           ; N: continue
                                                                    ; Y: change ret addr. to LeaveDos.
00006BF2 59                         pop     cx                      ; Note CX is not input to ExecReady
00006BF3 B9[F603]                   mov     cx,LeaveDOS
00006BF6 51                         push    cx
                                ;@@:
                                Exec_@f:
                                    ; M068 - End

                                    ;Enter

00006BF7 55                         push    bp
00006BF8 89E5                       mov     bp,sp
                                    ;;sub   sp,26   ; MSDOS 3.3
                                    ; 30/11/2022 (MSDOS 5.0, MSDOS.SYS compatibility)
                                    ;sub    sp,29   ; MSDOS 6.0 & MSDOS 6.22 & PCDOS 7.1 ; 07/03/2024
                                    ; 17/12/2022
                                    ; 20/05/2019
00006BFA 83EC1E                     sub     sp,30   ; Retro DOS v4.0

                                    ; MSDOS 6.0
00006BFD 3C05                       cmp     AL,5                    ; only 0, 1, 3 or 5 are allowed ;M028
                                                                    ; M030
00006BFF 7611                       jna     short Exec_Check_2

                                    ; MSDOS 3.3
                                    ;cmp    AL,3
                                    ;jna    short Exec_Check_2

                                Exec_Bad_Fun:
                                    ;mov    byte [ss:EXTERR_LOCUS],errLOC_Unk ; 1
                                                                    ; Extended Error Locus;smr;SS Override
                                    ; 01/07/2024
00006C01 E8D9A6                     call    set_exerr_locus_unk

                                    ;mov    al,1
00006C04 B001                       mov     al,error_invalid_function

                                Exec_Ret_Err:
                                    ;Leave
00006C06 89EC                       mov     sp,bp
00006C08 5D                         pop     bp
                                    ;transfer SYS_RET_ERR
00006C09 E96C9A                     jmp     SYS_RET_ERR

                                    ; MSDOS 6.0
                                ExecReadyJ:
00006C0C E89C18                     call    ExecReady               ; M028
00006C0F E91204                     jmp     norm_ovl                ; do a Leave & xfer sysret_OK ; M028

                                Exec_Check_2:
00006C12 3C02                       cmp     AL,2
00006C14 74EB                       je      short Exec_Bad_Fun

                                    ; MSDOS 6.0
00006C16 3C04                       cmp     al,4                    ; 2 & 4 are not allowed
00006C18 74E7                       je      short Exec_Bad_Fun

00006C1A 3C05                       cmp     al,5                    ; M028 ; M030
00006C1C 74EE                       je      short ExecReadyJ        ; M028

                                    ;mov    [bp-4],bx
00006C1E 895EFC                     mov     Exec_BlkL,BX            ; stash args
                                    ;mov    [bp-2],es
00006C21 8C46FE                     mov     Exec_BlkH,ES
                                    ;mov    [bp-5],al
00006C24 8846FB                     mov     Exec_Func,AL

                                    ;mov    byte [bp-6],0
00006C27 C646FA00                   mov     Exec_Load_High,0

                                    ;mov    [bp-26],dx
00006C2B 8956E6                     mov     ExecNameL,DX            ; set up length of exec name
                                    ;mov    [bp-24],ds
00006C2E 8C5EE8                     mov     ExecNameH,DS
00006C31 89D6                       mov     SI,DX                   ; move pointer to convenient place
                                    ;invoke DStrLen
00006C33 E8A1AB                     call    DStrLen
                                    ;mov    [bp-22],cx
00006C36 894EEA                     mov     ExecNameLen,CX          ; save length

                                    ; MSDOS 6.0
00006C39 36A0[0203]                 mov     al,[ss:AllocMethod]     ; M063: save alloc method in
00006C3D 36A2[8400]                 mov     [ss:ALLOCMSAVE],al      ; M063: AllocMsave

                                    ;xor    AL,AL                   ; open for reading
                                    ; 07/03/2024 (PCDOS 7.1 IBMDOS.COM)
                                    ;;;
00006C41 B0A0                       mov     al,0A0h                 ; Access mode bits: (0 to 2)
                                                                    ; 000  read access

                                                                    ; Sharing mode bits: (4 to 6)
                                                                    ; 010  deny others write access

                                                                    ; bit 7 - private
                                    ;;;

00006C43 55                         push    BP

                                    ; MSDOS 6.0
                                    ;or     byte [ss:DOS_FLAG],1
00006C44 36800E[8600]01             or      byte [ss:DOS_FLAG],EXECOPEN ; this flag is set to indicate to
                                                                    ; the redir that this open call is
                                                                    ; due to an exec.

```

```
                                    ; 07/03/2024 (PCDOS 7.1 IBMDOS.COM)
                                    ;;;
00006C4A 1E                         push    ds
00006C4B 52                         push    dx
                                    ;;;

                                    ;invoke $OPEN                ; is the file there?
00006C4C E87A13                     call    _$OPEN

                                    ; 07/03/2024 (PCDOS 7.1 IBMDOS.COM)
                                    ;;;
00006C4F 5A                         pop     dx
00006C50 1F                         pop     ds
00006C51 7305                       jnc     short Exec_Open_Ok

00006C53 30C0                       xor     al,al                ; open for reading
00006C55 E87113                     call    _$OPEN

                                Exec_Open_Ok:
                                    ;;;

                                    ; MSDOS 6.0
00006C58 9C                         pushf
                                    ; 02/06/2019
                                    ;and    byte [ss:DOS_FLAG],0FEh
00006C59 368026[8600]FE             and     byte [ss:DOS_FLAG],~EXECOPEN ; reset flag
00006C5F 9D                         popf

00006C60 5D                         pop     BP

                                    ; MSDOS 3.3 & MSDOS 6.0
00006C61 72A3                       jc      short Exec_Ret_Err

                                    ;mov    [bp-8],ax
00006C63 8946F8                     mov     Exec_FH,AX
00006C66 89C3                       mov     BX,AX
00006C68 30C0                       xor     AL,AL
                                    ;invoke $Ioctl
00006C6A E817BC                     call    _$IOCTL
00006C6D 7207                       jc      short Exec_BombJ

                                    ;test   dl,80h
00006C6F F6C280                     test    DL,devid_ISDEV
00006C72 740A                       jz      short Exec_Check_Environ

                                    ;mov    al,2
00006C74 B002                       mov     AL,error_file_not_found
                                Exec_BombJ:
00006C76 E9C800                     jmp     Exec_Bomb

                                BadEnv:
                                    ;mov    al,0Ah
00006C79 B00A                       mov     AL,error_bad_environment
00006C7B E9C300                     jmp     Exec_Bomb

                                Exec_Check_Environ:
                                    ;mov    word [bp-18],0
00006C7E C746EE0000                 mov     Exec_Load_Block,0
                                    ;mov    word [bp-14],0
00006C83 C746F20000                 mov     Exec_Environ,0
                                                                 ; overlays... no environment
                                    ;test   byte [bp-5],2
00006C88 F646FB02                   test    Exec_Func,exec_func_overlay
00006C8C 7552                       jnz     short Exec_Read_Header

                                    ;lds    si,[bp-4]
00006C8E C576FC                     lds     SI,Exec_Blk          ; get block
00006C91 8B04                       mov     ax,[SI]
                                    ;mov    AX,[SI+EXEC1.ENVIRON] ; address of environ
00006C93 09C0                       or      AX,AX
00006C95 750C                       jnz     short Exec_Scan_Env

00006C97 368E1E[3003]               mov     DS,[SS:CurrentPDB]    ;smr;SS Override
                                    ;mov    ax,[44]
00006C9C A12C00                     mov     AX,[PDB.ENVIRON]

                                ; MSDOS 6.0
                                ;----------------------------------------------BUG 92 4/30/90-----------------
                                ;
                                ; Exec_environ is being correctly initialized after the environment has been
                                ; allocated and copied form the parent's env. It must not be initialized here.
                                ; Because if the call to $alloc below fails Exec_dealloc will deallocate the
                                ; parent's environment.
                                ;   mov     Exec_Environ,AX
                                ;
                                ;-------------------------------------------------------------------------------

                                    ;mov    [bp-14],ax
                                    ;mov    Exec_Environ,ax

00006C9F 09C0                       or      AX,AX
00006CA1 743D                       jz      short Exec_Read_Header

                                Exec_Scan_Env:
00006CA3 8EC0                       mov     ES,AX
00006CA5 31FF                       xor     DI,DI
                                    ;mov    cx,7FFFh ; MSDOS 3.3
00006CA7 B90080                     mov     CX,8000h ; MSDOS 6.0  ; at most 32k of environment ;M040
00006CAA 30C0                       xor     AL,AL

                                Exec_Get_Environ_Len:
00006CAC F2AE                       repnz   scasb                ; find that nul byte
00006CAE 75C9                       jnz     short BadEnv

00006CB0 49                         dec     CX                   ; Dec CX for the next nul byte test
00006CB1 78C6                       js      short BadEnv         ; gone beyond the end of the environment

00006CB3 AE                         scasb                        ; is there another nul byte?
00006CB4 75F6                       jnz     short Exec_Get_Environ_Len ; no, scan some more

00006CB6 57                         push    DI
                                    ;lea    bx,[DI+11h]
00006CB7 8D5D11                     lea     BX,[DI+0Fh+2]
                                    ;add    bx,[bp-22]
00006CBA 035EEA                     add     BX,ExecNameLen       ; BX <- length of environment
                                                                 ; remember argv[0] length
                                                                 ; round up and remember argc
00006CBD B104                       mov     CL,4
00006CBF D3EB                       shr     BX,CL                ; number of paragraphs needed
00006CC1 06                         push    ES
                                    ;invoke $Alloc               ; can we get the space?
00006CC2 E84906                     call    _$ALLOC
00006CC5 1F                         pop     DS
00006CC6 59                         pop     CX
```

```
                                    ;jnc    short Exec_Save_Environ
                                    ;jmp    SHORT Exec_No_Mem     ; nope... cry and sob
                                    ; 17/12/2022
00006CC7 7272                       jc      short Exec_No_Mem ; 02/06/2019
                                    ; 30/11/2022 (MSDOS 5.0, MSDOS.SYS compatibility)
                                    ;jnc    short Exec_Save_Environ
                                    ;jmp    SHORT Exec_No_Mem

                                Exec_Save_Environ:
00006CC9 8EC0                       mov     ES,AX
                                    ;mov    [bp-14],ax
00006CCB 8946F2                     mov     Exec_Environ,AX       ; save him for a rainy day
00006CCE 31F6                       xor     SI,SI
00006CD0 89F7                       mov     DI,SI
00006CD2 F3A4                       rep     movsb                 ; copy the environment
00006CD4 B80100                     mov     AX,1
00006CD7 AB                         stosw
                                    ;lds    si,[bp-26]
00006CD8 C576E6                     lds     SI,ExecName
                                    ;mov    cx,[bp-22]
00006CDB 8B4EEA                     mov     CX,ExecNameLen
00006CDE F3A4                       rep     movsb

                                Exec_Read_Header:
                                    ; We read in the program header into the above data area and
                                    ; determine where in this memory the image will be located.

                                    ;Context DS
00006CE0 16                         push    ss
00006CE1 1F                         pop     ds
                                    ;mov    cx,26
00006CE2 B91A00                     mov     CX,exec_header_len    ; header size
00006CE5 BA[C70E]                   mov     DX,exec_signature
00006CE8 06                         push    ES
00006CE9 1E                         push    DS
00006CEA E88204                     call    ExecRead
00006CED 1F                         pop     DS
00006CEE 07                         pop     ES
00006CEF 724E                       jc      short Exec_Bad_File

00006CF1 09C0                       or      AX,AX
00006CF3 744A                       jz      short Exec_Bad_File
                                    ;cmp    ax,26
00006CF5 83F81A                     cmp     AX,exec_header_len    ; did we read the right number?
00006CF8 7519                       jnz     short Exec_Com_Filej  ; yep... continue

00006CFA F706[D30E]FFFF             test    word [exec_max_BSS],-1        ; indicate load high?
00006D00 7504                       jnz     short Exec_Check_Sig

                                    ;mov    byte [bp-6],0FFh
00006D02 C646FAFF                   mov     Exec_Load_High,-1

                                Exec_Check_Sig:
00006D06 A1[C70E]                   mov     AX,[exec_signature]   ; rms;NSS
                                    ;cmp    ax,5A4Dh ; 'MZ'
00006D09 3D4D5A                     cmp     AX,exe_valid_signature; zibo arises!
00006D0C 7408                       jz      short Exec_Save_Start ; assume com file if no signature

                                    ;cmp    ax,4D5Ah ; 'ZM'
00006D0E 3D5A4D                     cmp     AX,exe_valid_old_signature ; zibo arises!
00006D11 7403                       jz      short Exec_Save_Start ; assume com file if no signature

                                Exec_Com_Filej:
00006D13 E9EC01                     jmp     Exec_Com_File

                                    ; We have the program header... determine memory requirements

                                Exec_Save_Start:
00006D16 A1[CB0E]                   mov     AX,[exec_pages]               ; get 512-byte pages  ;rms;NSS
00006D19 B105                       mov     CL,5                  ; convert to paragraphs
00006D1B D3E0                       shl     AX,CL
00006D1D 2B06[CF0E]                 sub     AX,[exec_par_dir]     ; AX = size in paragraphs ;rms;NSS
                                    ;mov    [bp-12],ax
00006D21 8946F4                     mov     Exec_Res_Len_Para,AX

                                            ; Do we need to allocate memory?
                                            ; Yes if function is not load-overlay

                                    ;test   byte [bp-5],2
00006D24 F646FB02                   test    Exec_Func,exec_func_overlay
00006D28 7443                       jz      short Exec_Allocate   ; allocation of space

                                            ; get load address from block

                                    ;les    di,[bp-4]
00006D2A C47EFC                     les     DI,Exec_Blk

                                ; 07/03/2024
                                %if 0
                                    mov     ax,[es:di]
                                    ;mov    AX,[ES:DI+EXEC3.load_addr]
                                    ;mov    [bp-20],ax
                                    mov     Exec_DMA,AX

                                    ; 17/12/2022
                                    ;;mov   ax,[es:di+2]
                                    ;mov    AX,[ES:DI+EXEC3.reloc_fac]
                                    ;;mov   [bp-10],ax
                                    ;mov    Exec_Rel_Fac,AX

                                    ; 17/12/2022
                                    ; 30/11/2022 (!most proper code!)
                                    ;mov    dx,[es:di+2]
                                    mov     dx,[ES:DI+EXEC3.reloc_fac]
                                    ;mov    [bp-10],dx
                                    mov     Exec_Rel_Fac,dx
                                %else
                                    ; 07/03/2024 (PCDOS 7.1 IBMDOS.COM)
                                    ;;;
00006D2D 06                         push    es
00006D2E 26C405                     les     ax,[es:di]
                                    ;les    ax,[ES:DI+EXEC3.load_addr]
                                    ;mov    [bp-20],ax
00006D31 8946EC                     mov     Exec_DMA,ax
                                    ;mov    [bp-10],es
00006D34 8C46F6                     mov     Exec_Rel_Fac,es
00006D37 07                         pop     es
                                    ;;;
                                %endif
                                    ; ax = Exec_DMA
00006D38 E9DE00                     jmp     Exec_Find_Res

                                ; 17/12/2022
```

```
                                ; 30/11/2022 (MSDOS 5.0, MSDOS.SYS compatibility)
                                ; 27/09/2023
                                %if 0
                                    ; 02/06/2019 - Retro DOS v4.0
                                    ;mov    ax,[bp-20]  ; *+*
                                    mov     AX,Exec_DMA ; *+*
                                    ; 10/08/2018
                                    jmp     Exec_Find_Res         ; M000
                                %endif

                                Exec_No_Mem:
                                    ;mov    al,8
00006D3B B008                       mov     AL,error_not_enough_memory
00006D3D EB02                       jmp     short Exec_Bomb

                                Exec_Bad_File:
                                    ;mov    al,0Bh
00006D3F B00B                       mov     AL,error_bad_format

                                Exec_Bomb:
                                    ;mov    bx,[bp-8]
00006D41 8B5EF8                     mov     BX,Exec_FH
00006D44 E84104                     call    Exec_Dealloc
                                    ;LeaveCrit CritMem
00006D47 E8C5AB                     call    LCritMEM
                                    ;save   <AX,BP>
00006D4A 50                         push    ax
00006D4B 55                         push    bp
                                    ;invoke $CLOSE
00006D4C E81E0A                     call    _$CLOSE
                                    ;restore <BP,AX>
00006D4F 5D                         pop     bp
00006D50 58                         pop     ax
00006D51 E9B2FE                     jmp     Exec_Ret_Err

                                Exec_Chk_Mem:
                                    ; 24/09/2023
                                    ; ds = DOSDATA
                                    ; 17/12/2022
                                    ; 30/11/2022 (MSDOS 5.0, MSDOS.SYS compatibility)
                                    ;%if 0
                                        ; MSDOS 6.0                           ; M063 - Start
                                        ;mov    al,[ss:AllocMethod]   ; save current alloc method in ax
                                        ; 10/06/2019
00006D54 A0[0203]                       mov     al,[AllocMethod]
                                        ;mov    bl,[ss:ALLOCMSAVE]
00006D57 8A1E[8400]                     mov     bl,[ALLOCMSAVE]
                                        ;mov    [ss:AllocMethod],bl   ; restore original allocmethod
00006D5B 881E[0203]                     mov     [AllocMethod],bl

00006D5F F6C340                         test    bl,HIGH_ONLY ; 40h    ; Q: was the HIGH_ONLY bit already set
00006D62 75D7                           jnz     short Exec_No_Mem     ; Y: no space in UMBs. Quit
                                        ;                             ; N: continue
                                        ;
00006D64 A840                           test    al,HIGH_ONLY          ; Q: did we set the HIGH_ONLY bit
00006D66 74D3                           jz      short Exec_No_Mem     ; N: no memory
                                        ; 02/06/2019
                                        ;mov    ax,[ss:SAVE_AX]               ; Y: restore ax and
00006D68 A1[8A00]                       mov     ax,[SAVE_AX]
                                        ;jmp    short Exec_Norm_Alloc ;    Try again
                                                                      ; M063 - End
00006D6B EB2B                           jmp     short Exec_Norm_Alloc1
                                    ;%endif

                                    ; 17/12/2022
                                    %if 0
                                        ; 30/11/2022 (MSDOS 5.0, MSDOS.SYS compatibility)
                                        ; MSDOS 6.0                           ; M063 - Start
                                        mov     al,[ss:AllocMethod]   ; save current alloc method in ax
                                        mov     bl,[ss:ALLOCMSAVE]
                                        mov     [ss:AllocMethod],bl   ; restore original allocmethod

                                        test    bl,HIGH_ONLY ; 40h    ; Q: was the HIGH_ONLY bit already set
                                        jnz     short Exec_No_Mem     ; Y: no space in UMBs. Quit
                                        ;                             ; N: continue
                                        ;
                                        test    al,HIGH_ONLY          ; Q: did we set the HIGH_ONLY bit
                                        jz      short Exec_No_Mem     ; N: no memory

                                        mov     ax,[ss:SAVE_AX]               ; Y: restore ax and
                                        jmp     short Exec_Norm_Alloc ;    Try again
                                                                      ; M063 - End
                                    %endif

                                Exec_Allocate:
                                    ; 09/09/2018
                                    ; M005 - START
                                    ; If there is no STACK segment for this exe file and if this
                                    ; not an overlay and the resident size is less than 64K -
                                    ; 256 bytes we shall add 256 bytes to the programs
                                    ; resident memory requirement and set Exec_SP to this value.

                                    ; 17/12/2022
00006D6D 29DB                       sub     bx,bx ; 0

                                    ; MSDOS 6.0
                                    ;;mov   byte [bp-29],0
                                    ;mov    Exec_NoStack,0
                                    ; 17/12/2022
00006D6F 885EE3                     mov     Exec_NoStack,bl ; 0
00006D72 391E[D50E]                 cmp     [exec_SS],bx ; 0
                                    ;cmp    word [exec_SS],0      ; Q: is there a stack seg
00006D76 7511                       jne     short ea1             ; Y: continue normal processing
00006D78 391E[D70E]                 cmp     [exec_SP],bx ; 0
                                    ;cmp    word [exec_SP],0      ; Q: is there a stack ptr
00006D7C 750B                       jne     short ea1             ; Y: continue normal processing

                                    ;inc    byte [bp-29]
00006D7E FE46E3                     inc     Exec_NoStack
00006D81 3DF00F                     cmp     ax,1000h-10h ; 0FF0h  ; Q: is this >= 64K-256 bytes
00006D84 7303                       jae     short ea1             ; Y: don't set Exec_SP

00006D86 83C010                     add     ax,10h                ; add 10h paras to mem requirement
                                ea1:
                                    ; M005 - END

                                    ; MSDOS 6.0                   ; M000 - start
                                    ; 20/05/2019
                                    ; (ds = ss = DOSDATA)
00006D89 F606[0203]80               test    byte [AllocMethod],HIGH_FIRST ; 80h
                                                                  ; Q: is the alloc strat high_first
00006D8E 7405                       jz      short Exec_Norm_Alloc ; N: normal allocate
                                                                  ; Y: set high_only bit
```

```
40659 00006D90 800E[0203]40               or      byte [AllocMethod],HIGH_ONLY ; 40h
40660                                                                  ; M000 - end
40661                                  Exec_Norm_Alloc:
40662 00006D95 A3[8A00]                    mov     [SAVE_AX],ax        ; M000: save ax for possible 2nd
40663                                  Exec_Norm_Alloc1:  ; 02/06/2019
40664                                                                  ; M000: attempt at allocating memory
40665                                      ; MSDOS 3.3
40666                                      ;push   ax                  ; M000
40667
40668 00006D98 BBFFFF                      mov     BX,0FFFFh           ; see how much room in arena
40669 00006D9B 1E                          push    DS
40670                                      ;invoke $Alloc              ; should have carry set and BX has max
40671 00006D9C E86F05                      call    _$ALLOC
40672 00006D9F 1F                          pop     DS
40673
40674                                      ; MSDOS 6.0
40675 00006DA0 A1[8A00]                    mov     AX,[SAVE_AX]        ; M000
40676                                      ; MSDOS 3.3
40677                                      ;pop    ax                  ; M000
40678
40679 00006DA3 83C010                      add     AX,10h              ; room for header
40680 00006DA6 83FB11                      cmp     BX,11h              ; enough room for a header
40681                                      ; MSDOS 6.0
40682 00006DA9 72A9                        jb      short Exec_Chk_Mem  ; M000
40683                                      ; MSDOS 3.3
40684                                      ;jb     short Exec_No_Mem
40685
40686 00006DAB 39D8                        cmp     AX,BX               ; is there enough for bare image?
40687                                      ; MSDOS 6.0
40688 00006DAD 77A5                        ja      short Exec_Chk_Mem  ; M000
40689                                      ; MSDOS 3.3
40690                                      ;ja     short Exec_No_Mem
40691
40692                                      ;test   byte [bp-6],0FFh
40693 00006DAF F646FAFF                    test    Exec_Load_High,-1   ; if load high, use max
40694 00006DB3 7518                        jnz     short Exec_BX_Max   ; use max
40695
40696                                      ; 09/09/2018
40697
40698 00006DB5 0306[D10E]                  add     AX,[exec_min_BSS]   ; go for min allocation;rms;NSS
40699                                      ; MSDOS 6.0
40700 00006DB9 7299                        jc      short Exec_Chk_Mem  ; M000
40701                                      ; MSDOS 3.3
40702                                      ;jc     short Exec_No_Mem
40703
40704 00006DBB 39D8                        cmp     AX,BX               ; enough space?
40705                                      ; MSDOS 6.0
40706 00006DBD 7795                        ja      short Exec_Chk_Mem  ; M000: nope...
40707                                      ; MSDOS 3.3
40708                                      ;ja     short Exec_No_Mem
40709
40710 00006DBF 2B06[D10E]                  sub     AX,[exec_min_BSS]   ; rms;NSS
40711 00006DC3 0306[D30E]                  add     AX,[exec_max_BSS]   ; go for the MAX
40712 00006DC7 7204                        jc      short Exec_BX_Max
40713
40714 00006DC9 39D8                        cmp     AX,BX
40715 00006DCB 7602                        jbe     short Exec_Got_Block
40716
40717                                  Exec_BX_Max:
40718 00006DCD 89D8                        mov     AX,BX
40719
40720                                  Exec_Got_Block:
40721                                      ; 03/08/2018 - Retro DOS v3.0
40722
40723 00006DCF 1E                          push    DS
40724 00006DD0 89C3                        mov     BX,AX
40725                                      ;mov    [bp-16],bx
40726 00006DD2 895EF0                      mov     Exec_Size,BX
40727                                      ;invoke $Alloc              ; get the space
40728 00006DD5 E83605                      call    _$ALLOC
40729 00006DD8 1F                          pop     DS
40730                                      ; MSDOS 6.0
40731                                      ;jc     short Exec_Chk_Mem  ; M000
40732                                      ; MSDOS 3.3
40733                                      ;;jc    short Exec_No_Mem
40734                                      ; 20/05/2019
40735 00006DD9 7303                        jnc     short ea0
40736 00006DDB E976FF                      jmp     Exec_Chk_Mem
40737                                  ea0:
40738                                      ; MSDOS 6.0
40739 00006DDE 8A0E[8400]                  mov     cl,[ALLOCMSAVE]             ; M063:
40740 00006DE2 880E[0203]                  mov     [AllocMethod],cl    ; M063: restore allocmethod
40741
40742                                  ;M029; Begin changes
40743                                  ; This code does special handling for programs with no stack segment. If so,
40744                                  ;check if the current block is larger than 64K. If so, we do not modify
40745                                  ;Exec_SP. If smaller than 64K, we make Exec_SP = top of block. In either
40746                                  ;case Exec_SS is not changed.
40747
40748                                      ; MSDOS 6.0
40749                                      ;cmp    byte [bp-29],0
40750 00006DE6 807EE300                    cmp     Exec_NoStack,0
40751                                      ;je     @f
40752 00006DEA 7412                        je      short ea2
40753
40754 00006DEC 81FB0010                    cmp     bx,1000h            ; Q: >= 64K memory block
40755                                      ;jae    @f                  ; Y: Exec_SP = 0
40756 00006DF0 730C                        jae     short ea2
40757
40758                                  ;Make Exec_SP point at the top of the memory block
40759
40760 00006DF2 B104                        mov     cl,4
40761 00006DF4 D3E3                        shl     bx,cl               ; get byte offset
40762 00006DF6 81EB0001                    sub     bx,100h             ; take care of PSP
40763 00006DFA 891E[D70E]                  mov     [exec_SP],bx        ; Exec_SP = top of block
40764                                  ea2:
40765                                  ;@@:
40766                                  ;M029; end changes
40767
40768                                      ;mov    [bp-18],ax
40769 00006DFE 8946EE                      mov     Exec_Load_Block,AX
40770 00006E01 83C010                      add     AX,10h
40771                                      ;test   byte [bp-6],0FFh
40772 00006E04 F646FAFF                    test    Exec_Load_High,-1
40773 00006E08 7409                        jz      short Exec_Use_AX   ; use ax for load info
40774
40775                                      ;add    ax,[bp-16]
40776 00006E0A 0346F0                      add     AX,Exec_Size        ; go to end
40777                                      ;sub    ax,[bp-12]
40778 00006E0D 2B46F4                      sub     AX,Exec_Res_Len_Para ; drop off header
40779 00006E10 83E810                      sub     AX,10h              ; drop off pdb
40780
40781                                  Exec_Use_AX:
40782                                      ;mov    [bp-10],ax
```

```
                                    mov     Exec_Rel_Fac,AX       ; new segment
                                    ;mov    [bp-20],ax
00006E16 8946EC                     mov     Exec_DMA,AX ; *+*     ; beginning of dma

                                    ; Determine the location in the file of the beginning of
                                    ; the resident

                                ; 17/12/2022
                                ; 30/11/2022 (MSDOS 5.0, MSDOS.SYS compatibility)
                                ;%if 0

                                Exec_Find_Res:
                                    ; MSDOS 6.0
                                    ;;mov   dx,[bp-20]
                                    ;mov    DX,Exec_DMA ; *+*
                                    ;;mov   [bp-28],dx
                                    ;mov    Exec_DMA_Save,DX

                                    ; 17/12/2022
                                    ; AX = Exec_DMA

                                    ; 02/06/2019 - Retro DOS v4.0
                                    ;mov    [bp-28],ax ; *+*
00006E19 8946E4                     mov     Exec_DMA_Save,AX ; *+*

                                ;%endif

                                ; 17/12/2022
                                %if 0
                                    ; 30/11/2022 (MSDOS 5.0, MSDOS.SYS compatibility)
                                Exec_Find_Res:
                                    ;mov    dx,[bp-20]
                                    mov     DX,Exec_DMA ; *+*
                                    ;mov    [bp-28],dx
                                    mov     Exec_DMA_Save,DX
                                %endif

                                    ; MSDOS 3.3 (& MSDOS 6.0)
00006E1C 8B16[CF0E]                 mov     DX,[exec_par_dir]
00006E20 52                         push    DX
00006E21 B104                       mov     CL,4
00006E23 D3E2                       shl     DX,CL                 ; low word of location
00006E25 58                         pop     AX
00006E26 B10C                       mov     CL,12
00006E28 D3E8                       shr     AX,CL                 ; high word of location
00006E2A 89C1                       mov     CX,AX                 ; CX <- high

                                    ; Read in the resident image (first, seek to it)
                                    ;mov    bx,[bp-8]
00006E2C 8B5EF8                     mov     BX,Exec_FH
00006E2F 1E                         push    DS
00006E30 30C0                       xor     AL,AL
                                    ;invoke $Lseek                ; Seek to resident
00006E32 E8A10A                     call    _$LSEEK
00006E35 1F                         pop     DS
00006E36 7303                       jnc     short Exec_Big_Read

00006E38 E906FF                     jmp     Exec_Bomb

                                Exec_Big_Read:                          ; Read resident into memory
                                    ;mov    bx,[bp-12]
00006E3B 8B5EF4                     mov     BX,Exec_Res_Len_Para
00006E3E 81FB0010                   cmp     BX,1000h              ; Too many bytes to read?
00006E42 7203                       jb      short Exec_Read_OK

00006E44 BBE00F                     mov     BX,0FE0h              ; Max in one chunk FE00 bytes

                                Exec_Read_OK:
                                    ;sub    [bp-12],bx
00006E47 295EF4                     sub     Exec_Res_Len_Para,BX  ; We read (soon) this many
00006E4A 53                         push    BX
00006E4B B104                       mov     CL,4
00006E4D D3E3                       shl     BX,CL                 ; Get count in bytes from paras
00006E4F 89D9                       mov     CX,BX                 ; Count in correct register
00006E51 1E                         push    DS
                                    ;mov    ds,[bp-20]
00006E52 8E5EEC                     mov     DS,Exec_DMA           ; Set up read buffer

00006E55 31D2                       xor     DX,DX
00006E57 51                         push    CX                    ; Save our count
00006E58 E81403                     call    ExecRead
00006E5B 59                         pop     CX                    ; Get old count to verify
00006E5C 1F                         pop     DS
00006E5D 7248                       jc      short Exec_Bad_FileJ

00006E5F 39C1                       cmp     CX,AX                 ; Did we read enough?
00006E61 5B                         pop     BX                    ; Get paragraph count back
00006E62 7408                       jz      short ExecCheckEnd    ; and do reloc if no more to read

                                    ; The read did not match the request. If we are off by 512
                                    ; bytes or more then the header lied and we have an error.

00006E64 29C1                       sub     CX,AX
00006E66 81F90002                   cmp     CX,512
00006E6A 733B                       jae     short Exec_Bad_FileJ

                                    ; We've read in CX bytes... bump DTA location

                                ExecCheckEnd:
                                    ;add    [bp-20],bx
00006E6C 015EEC                     add     Exec_DMA,BX           ; Bump dma address
                                    ;test   word [bp-12],0FFFFh
00006E6F F746F4FFFF                 test    Exec_Res_Len_Para,-1
00006E74 75C5                       jnz     short Exec_Big_Read

                                    ; The image has now been read in. We must perform relocation
                                    ; to the current location.

                                exec_do_reloc:
                                    ;mov    cx,[bp-10]
00006E76 8B4EF6                     mov     CX,Exec_Rel_Fac
00006E79 A1[D50E]                   mov     AX,[exec_SS]          ; get initial SS ;rms;NSS
00006E7C 01C8                       add     AX,CX                 ; and relocate him
00006E7E A3[C10E]                   mov     [exec_init_SS],AX     ; rms;NSS

00006E81 A1[D70E]                   mov     AX,[exec_SP]          ; initial SP ;rms;NSS
00006E84 A3[BF0E]                   mov     [exec_init_SP],AX     ; rms;NSS

00006E87 C406[DB0E]                 les     AX,[exec_IP]          ; rms;NSS
00006E8B A3[C30E]                   mov     [exec_init_IP],AX     ; rms;NSS
00006E8E 8CC0                       mov     AX,ES                 ; rms;NSS
00006E90 01C8                       add     AX,CX                 ; relocated...
00006E92 A3[C50E]                   mov     [exec_init_CS],AX     ; rms;NSS

```

```
40907 00006E95 31C9                     xor     CX,CX
40908 00006E97 8B16[DF0E]               mov     DX,[exec_rle_table]   ; rms;NSS
40909                                   ;mov    bx,[bp-8]
40910 00006E9B 8B5EF8                   mov     BX,Exec_FH
40911 00006E9E 1E                       push    DS
40912 00006E9F 31C0                     xor     AX,AX
40913                                   ;invoke $Lseek
40914 00006EA1 E8320A                   call    _$LSEEK
40915 00006EA4 1F                       pop     DS
40916 00006EA5 7303                     jnc     short exec_get_entries
40917
40918                                 Exec_Bad_FileJ:
40919 00006EA7 E995FE                   jmp     Exec_Bad_File
40920
40921                                 exec_get_entries:
40922 00006EAA 8B16[CD0E]               mov     DX,[exec_rle_count]   ; Number of entries left ;rms;NSS
40923
40924                                 exec_read_reloc:
40925 00006EAE 52                       push    DX
40926                                   ;mov    dx,OPENBUF
40927 00006EAF BA[BE03]                 mov     DX,Exec_Internal_Buffer
40928                                   ;;mov   cx,388 ; MSDOS 3.3 ; (390>>2)<<2
40929                                   ;mov    cx,396 ; MSDOS 6.0    ; PCDOS 7.1 (08/03/2024)
40930 00006EB2 B98C01                   mov     CX,((Exec_Internal_Buffer_Size)/4)*4 ; (397>>2)<<2
40931 00006EB5 1E                       push    DS
40932 00006EB6 E8B602                   call    ExecRead
40933 00006EB9 07                       pop     ES
40934 00006EBA 5A                       pop     DX
40935 00006EBB 72EA                     jc      short Exec_Bad_FileJ
40936
40937                                   ;;mov   cx,97 ;  MSDOS 3.3 ; (390>>2)
40938                                   ;mov    cx,99 ; MSDOS 6.0     ; PCDOS 7.1 (08/03/2024)
40939 00006EBD B96300                   mov     CX,(Exec_Internal_Buffer_Size)/4 ; (397>>2)
40940                                                                   ; Pointer to byte location in header
40941                                   ;mov    di,OPENBUF
40942 00006EC0 BF[BE03]                 mov     DI,Exec_Internal_Buffer
40943                                   ;mov    si,[bp-10]
40944 00006EC3 8B76F6                   mov     SI,Exec_Rel_Fac       ; Relocate a single address
40945
40946                                 exec_reloc_one:
40947 00006EC6 09D2                     or      DX,DX                 ; Any more entries?
40948 00006EC8 7416                     jz      short Exec_Set_PDBJ
40949
40950                                 exec_get_addr:
40951 00006ECA 26C51D                   lds     BX,[ES:DI]            ; Get ra/sa of entry
40952 00006ECD 8CD8                     mov     AX,DS                 ; Relocate address of item
40953
40954                                   ; MSDOS 6.0
40955                                 ;;;;;;      add     AX,SI  ; MSDOS 3.3
40956                                   ;add    ax,[bp-28]
40957 00006ECF 0346E4                   add     AX,Exec_DMA_Save
40958
40959 00006ED2 8ED8                     mov     DS,AX
40960 00006ED4 0137                     add     [BX],SI
40961 00006ED6 83C704                   add     DI,4
40962 00006ED9 4A                       dec     DX
40963 00006EDA E2EA                     loop    exec_reloc_one        ; End of internal buffer?
40964
40965                                   ; We've exhausted a single buffer's worth. Read in the next
40966                                   ; piece of the relocation table.
40967
40968 00006EDC 06                       push    ES
40969 00006EDD 1F                       pop     DS
40970 00006EDE EBCE                     jmp     short exec_read_reloc
40971
40972                                 Exec_Set_PDBJ:
40973                                   ; MSDOS 6.0
40974
40975                                   ; We now determine if this is a buggy exe packed file and if
40976                                   ; so we patch in the right code. Note that fixexepatch will
40977                                   ; point to a ret if dos loads low. The load segment as
40978                                   ; determined above will be in exec_dma_save
40979
40980 00006EE0 06                       push    es
40981 00006EE1 50                       push    ax                    ; M030
40982 00006EE2 51                       push    cx                    ; M030
40983                                   ;mov    es,[bp-28]
40984 00006EE3 8E46E4                   mov     es,Exec_DMA_Save
40985 00006EE6 36A1[C50E]               mov     ax,[ss:exec_init_CS]  ; M030
40986 00006EEA 368B0E[C30E]             mov     cx,[ss:exec_init_IP]  ; M030
40987 00006EEF 36FF16[670D]             call    word [ss:FixExePatch]
40988
40989                                   ; 30/11/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
40990                                   ; (MSDOS 5.0 MSDOS.SYS does not contain 'Rational386Patch')
40991                                   ;call   word [ss:Rational386PatchPtr]
40992
40993                                   ; 08/03/2024 - Retro DOS v5.0
40994                                   ; (MSDOS 6.22 MSDOS.SYS - DOSCODE:9E6Fh)
40995                                   ; (PCDOS 7.1 IBMDOS.COM - DOSCODE:0B096h)
40996 00006EF4 36FF16[3E13]             call    word [ss:Rational386PatchPtr]
40997
40998                                   ; 08/03/2024
40999                                   ; (Windows ME IO.SYS - BIOSCODE:976Fh)
41000                                   ;call   Rational386Patch
41001
41002 00006EF9 59                       pop     cx                    ; M030
41003 00006EFA 58                       pop     ax                    ; M030
41004 00006EFB 07                       pop     es
41005
41006 00006EFC E9DD00                   jmp     Exec_Set_PDB
41007
41008                                 Exec_No_Memj:
41009 00006EFF E939FE                   jmp     Exec_No_Mem
41010
41011                                   ; we have a .COM file. First, determine if we are merely
41012                                   ; loading an overlay.
41013
41014                                 Exec_Com_File:
41015                                   ;test   byte [bp-5],2
41016 00006F02 F646FB02                 test    Exec_Func,exec_func_overlay
41017 00006F06 742D                     jz      short Exec_Alloc_Com_File
41018                                   ;lds    si,[bp-4]
41019 00006F08 C576FC                   lds     SI,Exec_Blk           ; get arg block
41020 00006F0B AD                       lodsw                         ; get load address
41021                                   ;mov    [bp-20],ax
41022 00006F0C 8946EC                   mov     Exec_DMA,AX
41023 00006F0F B8FFFF                   mov     AX,0FFFFh
41024 00006F12 EB63                     jmp     short Exec_Read_Block ; read it all!
41025
41026                                 Exec_Chk_Com_Mem:
41027                                   ; MSDOS 6.0                   ; M063 - Start
41028 00006F14 36A0[0203]               mov     al,[ss:AllocMethod]   ; save current alloc method in ax
41029 00006F18 368A1E[8400]             mov     bl,[ss:ALLOCMSAVE]
41030 00006F1D 36881E[0203]             mov     [ss:AllocMethod],bl   ; restore original allocmethod
```

```
41031 00006F22 F6C340                   test    bl,HIGH_ONLY ; 40h      ; Q: was the HIGH_ONLY bit already set
41032 00006F25 75D8                     jnz     short Exec_No_Memj      ; Y: no space in UMBs. Quit
41033                                                                   ; N: continue
41034
41035 00006F27 A840                     test    al,HIGH_ONLY            ; Q: did we set the HIGH_ONLY bit
41036 00006F29 74D4                     jz      short Exec_No_Memj      ; N: no memory
41037
41038                                   ;mov    ax,[bp-18]
41039 00006F2B 8B46EE                   mov     ax,Exec_Load_Block      ; M047: ax = block we just allocated
41040 00006F2E 31DB                     xor     bx,bx                   ; M047: bx => free arena
41041 00006F30 E87102                   call    ChangeOwner             ; M047: free this block
41042
41043 00006F33 EB0E                     jmp     short Exec_Norm_Com_Alloc
41044                                                                   ; M063 - End
41045
41046                                   ; We must allocate the max possible size block (ick!)
41047                                   ; and set up CS=DS=ES=SS=PDB pointer, IP=100, SP=max
41048                                   ; size of block.
41049
41050                                 Exec_Alloc_Com_File:
41051                                     ; MSDOS 6.0                     ; M000 -start
41052 00006F35 36F606[0203]80           test    byte [ss:AllocMethod],HIGH_FIRST ; 80h
41053                                                                   ; Q: is the alloc strat high_first
41054 00006F3B 7406                     jz      short Exec_Norm_Com_Alloc ; N: normal allocate
41055                                                                   ; Y: set high_only bit
41056 00006F3D 36800E[0203]40           or      byte [ss:AllocMethod],HIGH_ONLY ; 40h
41057                                                                   ; M000 - end
41058                                 Exec_Norm_Com_Alloc:                    ; M000
41059                                     ; MSDOS 3.3 (& MSDOS 6.0)
41060 00006F43 BBFFFF                   mov     BX,0FFFFh
41061                                   ;invoke $Alloc                  ; largest piece available as error
41062 00006F46 E8C503                   call    _$ALLOC
41063 00006F49 09DB                     or      BX,BX
41064                                   ; MSDOS 6.0
41065 00006F4B 74C7                     jz      short Exec_Chk_Com_Mem; M000
41066                                   ; MSDOS 3.3
41067                                   ;jz     short Exec_No_Memj
41068
41069                                   ;mov    [bp-16],bx
41070 00006F4D 895EF0                   mov     Exec_Size,BX            ; save size of allocation block
41071 00006F50 53                       push    BX
41072                                   ;invoke $ALLOC                  ; largest piece available
41073 00006F51 E8BA03                   call    _$ALLOC
41074 00006F54 5B                       pop     BX                      ; get size of block...
41075                                   ;mov    [bp-18],ax
41076 00006F55 8946EE                   mov     Exec_Load_Block,AX
41077
41078 00006F58 83C010                   add     AX,10h                  ; increment for header
41079                                   ;mov    [bp-20],ax
41080 00006F5B 8946EC                   mov     Exec_DMA,AX
41081
41082 00006F5E 31C0                     xor     AX,AX                   ; presume 64K read...
41083 00006F60 81FB0010                 cmp     BX,1000h                ; 64k or more in block?
41084 00006F64 730E                     jae     short Exec_Read_Com     ; yes, read only 64k
41085
41086 00006F66 89D8                     mov     AX,BX                   ; convert size to bytes
41087 00006F68 B104                     mov     CL,4
41088 00006F6A D3E0                     shl     AX,CL
41089                                   ; 17/12/2022
41090                                   ; 30/11/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
41091                                   ;                       (MSDOS 5.0, MSDOS.SYS compatibility)
41092                                   ; MSDOS 5.0
41093                                   ;cmp    AX,100h
41094                                   ; 02/06/2019 - Retro DOS v4.0
41095                                   ; MSDOS 6.0
41096                                       ; 17/12/2022
41097 00006F6C 3D0002                   cmp     AX,200h                   ; enough memory for PSP and stack?
41098 00006F6F 76A3                     jbe     short Exec_Chk_Com_Mem; M000: jump if not
41099                                   ;;jbe   short Exec_No_Memj      ; M000: jump if not
41100                                   ;; Retro DOS v3.0 modification (on MSDOS 6.0 code) -03/08/2018-
41101                                   ;;jbe   short Exec_Chk_Com_Mem; M000: jump if not
41102                                   ;jbe    short Exec_No_Memj      ; M000: jump if not
41103
41104                                                                   ; M047: size of the block is < 64K
41105 00006F71 2D0001                   sub     ax,100h                 ; M047: reserve 256 bytes for stack
41106
41107                                 Exec_Read_Com:
41108                                     ; MSDOS 3.3 (& MSDOS 6.0)
41109 00006F74 2D0001                   sub     AX,100h                 ; remember size of psp
41110                                 Exec_Read_Block:
41111 00006F77 50                       push    AX                      ; save number to read
41112                                   ;mov    bx,[bp-8]
41113 00006F78 8B5EF8                   mov     BX,Exec_FH              ; of com file
41114 00006F7B 31C9                     xor     CX,CX                   ; but seek to 0:0
41115 00006F7D 31C0                     xor     AX,AX                   ; seek relative to beginning
41116                                   ;mov    DX,CX
41117                                   ; 08/03/2024
41118 00006F7F 99                       cwd
41119                                   ;invoke $Lseek                  ; back to beginning of file
41120 00006F80 E85309                   call    _$LSEEK
41121 00006F83 59                       pop     CX                      ; number to read
41122                                   ;mov    ds,[bp-20]
41123 00006F84 8E5EEC                   mov     DS,Exec_DMA
41124 00006F87 31D2                     xor     DX,DX
41125 00006F89 51                       push    CX
41126 00006F8A E8E201                   call    ExecRead
41127 00006F8D 5E                       pop     SI                      ; get number of bytes to read
41128 00006F8E 7303                     jnc     short OkRead
41129 00006F90 E9ACFD                   jmp     Exec_Bad_File
41130
41131                                   ; 10/09/2018
41132                                 OkRead:
41133 00006F93 39F0                     cmp     AX,SI                   ; did we read them all?
41134                                   ; MSDOS 6.0
41135                                   ;jz     short Exec_Chk_Com_Mem; M00: exactly the wrong number...no
41136                                   ; MSDOS 3.3
41137                                   ;;jz    short Exec_No_Memj      ; M00: exactly the wrong number...
41138 00006F95 7503                     jne     short OkRead2
41139 00006F97 E97AFF                   jmp     Exec_Chk_Com_Mem
41140
41141                                 OkRead2:
41142                                   ; MSDOS 6.0
41143 00006F9A 368A1E[8400]             mov     bl,[ss:ALLOCMSAVE]      ; M063
41144 00006F9F 36881E[0203]             mov     [ss:AllocMethod],bl     ; M063: restore alloc method
41145
41146                                   ; MSDOS 3.3 (& MSDOS 6.0)
41147                                   ;test   byte [bp-5],2
41148 00006FA4 F646FB02                 test    Exec_Func,exec_func_overlay
41149 00006FA8 7532                     jnz     short Exec_Set_PDB      ; no starto, chumo!
41150
41151                                   ;mov    ax,[bp-20]
41152 00006FAA 8B46EC                   mov     AX,Exec_DMA
41153 00006FAD 83E810                   sub     AX,10h
41154 00006FB0 36A3[C50E]               mov     [SS:exec_init_CS],AX
```

```
                                    mov     word [SS:exec_init_IP],100h ; initial IP is 100h

                                    ; SI is AT MOST FF00h. Add FE to account for PSP - word
                                    ; of 0 on stack.

                                    add     SI,0FEh                 ; make room for stack

                                    ; MSDOS 6.0
                                    cmp     si,0FFFEh               ; M047: Q: was there >= 64K available
                                    je      short Exec_St_Ok        ; M047: Y: stack is fine
                                    add     si,100h                 ; M047: N: add the xtra 100h for stack

                                Exec_St_Ok:
                                    ; MSDOS 3.3 (& MSDOS 6.0)
                                    mov     [SS:exec_init_SP],SI    ; max value for read is also SP!;smr;SS Override
                                    mov     [SS:exec_init_SS],AX                                ;smr;SS Override
                                    mov     DS,AX
                                    mov     WORD [SI],0             ; 0 for return

                                    ; MSDOS 6.0

                                    ; M068
                                    ;
                                    ; We now determine if this is a Copy Protected App. If so the
                                    ; A20OFF_COUNT is set to 6. Note that ChkCopyProt will point to
                                    ; a ret if DOS is loaded low. Also DS contains the load segment.

                                    call    word [ss:ChkCopyProt]

                                Exec_Set_PDB:
                                    ; MSDOS 3.3 (& MSDOS 6.0)
                                    ;mov    bx,[bp-8]
                                    mov     BX,Exec_FH              ; we are finished with the file.
                                    call    Exec_Dealloc
                                    push    BP
                                    ;invoke $Close                  ; release the jfn
                                    call    _$CLOSE
                                    pop     BP
                                    call    Exec_Alloc
                                    ;test   byte [bp-5],2
                                    test    Exec_Func,exec_func_overlay
                                    jz      short Exec_Build_Header

                                    ; MSDOS 6.0
                                    call    Scan_Execname
                                    call    Scan_Special_Entries
                                ;SR;
                                ;The current lie strategy uses the PSP to store the lie version. However,
                                ;device drivers are loaded as overlays and have no PSP. To handle them, we
                                ;use the Sysinit flag provided by the BIOS as part of a structure pointed at
                                ;by BiosDataPtr. If this flag is set, the overlay call has been issued from
                                ;Sysinit and therefore must be a device driver load. We then get the lie
                                ;version for this driver and put it into the Sysinit PSP. When the driver
                                ;issues the version check, it gets the lie version until the next overlay
                                ;call is issued.

                                    cmp     byte [ss:DriverLoad],0;was Sysinit processing done?
                                    je      short norm_ovl          ;yes, no special handling
                                    push    si
                                    push    es
                                    les     si,[ss:BiosDataPtr]     ;get ptr to BIOS data block

                                    ; (es:si points to 'SysinitPresent' address/flag in retrodos4.s)
                                    cmp     byte [es:si],0          ;in Sysinit?
                                    je      short sysinit_done      ;no, Sysinit is finished

                                    mov     es,[ss:CurrentPDB]      ;es = current PSP (Sysinit PSP)
                                    push    word [ss:SPECIAL_VERSION]
                                    ;pop    word [es:40h]  ; 08/03/2024
                                    pop     word [es:PDB.Version] ;store lie version in Sysinit PSP
                                                            ;;; PDB.VERSION
                                    jmp     short setver_done
                                sysinit_done:
                                    mov     byte [ss:DriverLoad],0;Sysinit done,special handling off
                                setver_done:
                                    pop     es
                                    pop     si
                                norm_ovl:
                                    ;leave
                                    mov     sp,bp
                                    pop     bp

                                    ;transfer SYS_RET_OK            ; overlay load -> done
                                    jmp     SYS_RET_OK

                                Exec_Build_Header:
                                    ;mov    dx,[bp-18]
                                    mov     DX,Exec_Load_Block      ; assign the space to the process
                                    ;mov    si,1
                                    mov     SI,ARENA.OWNER          ; pointer to owner field
                                    ;mov    ax,[bp-14]
                                    mov     AX,Exec_Environ         ; get environ pointer
                                    or      AX,AX
                                    jz      short No_Owner          ; no environment

                                    dec     AX                      ; point to header
                                    mov     DS,AX
                                    mov     [SI],DX                 ; assign ownership
                                No_Owner:
                                    ;mov    ax,[bp-18]
                                    ;mov    AX,Exec_Load_Block      ; get load block pointer
                                    ; 07/12/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
                                    ; 17/12/2022
                                    mov     ax,dx ; 06/06/2019
                                    ;mov    ax,Exec_Load_Block      ; get load block pointer

                                    dec     AX
                                    mov     DS,AX                   ; point to header
                                    mov     [SI],DX                 ; assign ownership

                                    ; MSDOS 6.0
                                    push    DS                      ;AN000;MS. make ES=DS
                                    pop     ES                      ;AN000;MS.
                                    ;mov    di,8
                                    mov     DI,ARENA.NAME           ;AN000;MS. ES:DI points to destination
                                    call    Scan_Execname           ;AN007;MS. parse execname
                                                                    ;          ds:si->name, cx=name length
                                    push    CX                      ;AN007;;MS. save for fake version
                                    push    SI                      ;AN007;;MS. save for fake version

                                MoveName:                           ;AN000;
                                    lodsb                           ;AN000;;MS. get char
                                    cmp     AL,'.'                  ;AN000;;MS. is '.', may be name.exe
```

```
                                    jz      short Mem_Done          ;AN000;;MS. no, move to header
                                                                    ;AN000;
                                    stosb                           ;AN000;;MS. move char
                                                                    ; MSKK bug fix - limit length copied
                                    cmp     di,16 ; ARENAHEADERSIZE         ; end of memory arena block?
                                    jae     short Mem_Done          ; jump if so
                                    ;
                                    loop    MoveName                ;AN000;;MS. continue
                                Mem_Done:                           ;AN000;
                                    xor     AL,AL                   ;AN000;;MS. make ASCIIZ
                                    ;cmp    di,16
                                    cmp     DI,ARENAHEADERSIZE ; 16 ;AN000;MS. if not all filled
                                    jae     short Fill8             ;AN000;MS.

                                    stosb                           ;AN000;MS.

                                Fill8:                                      ;AN000;
                                    pop     SI                      ;AN007;MS. ds:si -> file name
                                    pop     CX                      ;AN007;MS.

                                    call    Scan_Special_Entries    ;AN007;MS.

                                    ; MSDOS 3.3 (& MSDOS 6.0)
                                    push    DX
                                    ;mov    si,[bp-16]
                                    mov     SI,Exec_Size
                                    add     SI,DX
                                    ;Invoke $Dup_PDB                ; ES is now PDB
                                    call    _$DUP_PDB
                                    pop     DX

                                    ;push   word [bp-14]
                                    push    Exec_Environ
                                    ;pop    WORD [ES:2Ch]
                                    pop     word [ES:PDB.ENVIRON]

                                    ; MSDOS 6.0                     ; *** Added for DOS 5.00
                                                                    ; version number in PSP
                                    push    word [ss:SPECIAL_VERSION] ; Set the DOS version number to
                                    pop     word [ES:PDB.Version] ; to be used for this application
                                            ; PDB.VERSION

                                    ; MSDOS 3.3 (& MSDOS 6.0)     ; set up proper command line stuff
                                    ;lds    si,[bp-4]
                                    lds     SI,Exec_Blk             ; get the block
                                    push    DS                      ; save its location
                                    push    SI
                                    ;lds    si,[si+6]
                                    lds     SI,[SI+EXEC0.5C_FCB]    ; get the 5c fcb

                                    ; DS points to user space 5C FCB

                                    mov     CX,12                   ; copy drive, name and ext
                                    push    CX
                                    mov     DI,5Ch
                                    mov     BL,[SI]
                                    rep     movsb

                                    ; DI = 5Ch + 12 = 5Ch + 0Ch = 68h

                                    ;xor    AX,AX                   ; zero extent, etc for CPM
                                    xchg    ax,cx   ; 08/03/2024
                                    stosw
                                    stosw

                                    ; DI = 5Ch + 12 + 4 = 5Ch + 10h = 6Ch

                                    pop     CX
                                    pop     SI                      ; get block
                                    pop     DS
                                    push    DS                      ; save (again)
                                    push    SI
                                    ;lds    si,[si+0Ah]
                                    lds     SI,[SI+EXEC0.6C_FCB]    ; get 6C FCB

                                    ; DS points to user space 6C FCB

                                    mov     BH,[SI]                 ; do same as above
                                    rep     movsb
                                    stosw
                                    stosw
                                    pop     SI                      ; get block (last time)
                                    pop     DS
                                    ;ld     si,[si+2]
                                    lds     SI,[SI+EXEC0.COM_LINE]; command line

                                    ; DS points to user space 80 command line

                                    or      CL,80h
                                    mov     DI,CX
                                    rep     movsb                   ; Wham!

                                    ; Process BX into default AX (validity of drive specs on args).
                                    ; We no longer care about DS:SI.

                                    dec     CL                      ; get 0FFh in CL
                                    mov     AL,BH
                                    xor     BH,BH
                                    ;invoke GetVisDrv
                                    call    GetVisDrv
                                    jnc     short Exec_BL

                                    mov     BH,CL

                                Exec_BL:
                                    mov     AL,BL
                                    xor     BL,BL
                                    ;invoke GetVisDrv
                                    call    GetVisDrv
                                    jnc     short Exec_Set_Return

                                    mov     BL,CL

                                Exec_Set_Return:
                                    ;invoke Get_User_Stack                  ; get his return address
                                    call    Get_User_Stack

                                ; 08/03/2024
                                %if 0
                                    ;push   word [si+14h]
                                    push    word [SI+user_env.user_CS]    ; suck out the CS and IP
                                    ;push   word [si+12h]
                                    push    word [SI+user_env.user_IP]
                                    ;push   word [si+14h]
```

```
                                    push    word [SI+user_env.user_CS]    ; suck out the CS and IP
                                    ;push   word [si+12h]
                                    push    word [SI+user_env.user_IP]
                                    ;pop    word [ES:0Ah]
                                    pop     WORD [ES:PDB.EXIT]
                                    ;pop    word [ES:0Ch]
                                    pop     WORD [ES:PDB.EXIT+2]
                                %else
                                    ; 07/03/2024 (PCDOS 7.1 IBMDOS.COM)
                                    ;;;
                                    ;lds    ax,[si+12h]
000070CE C54412                     lds     ax,[SI+user_env.user_IP] ; suck out the CS and IP
000070D1 1E                         push    ds
000070D2 50                         push    ax
                                    ;mov    [es:0Ah],ax
000070D3 26A30A00                   mov     [ES:PDB.EXIT],ax
                                    ;mov    [es:0Ch],ds
000070D7 268C1E0C00                 mov     [ES:PDB.EXIT+2],ds
                                    ;;;
                                %endif

000070DC 31C0                       xor     AX,AX
000070DE 8ED8                       mov     DS,AX
                                                                ; save them where we can get them
                                                                ; later when the child exits.
                                    ;pop    word [88h]
000070E0 8F068800                   pop     word [addr_int_terminate] ; 22h*4
                                    ;pop    word [90h]
000070E4 8F068A00                   pop     word [addr_int_terminate+2] ; (22h*4)+2

000070E8 36C706[2C03]8000           mov     WORD [SS:DMAADD],80h  ; SS Override
000070EF 368E1E[3003]               mov     DS,[SS:CurrentPDB]    ; SS Override
000070F4 368C1E[2E03]               mov     [SS:DMAADD+2],DS      ; SS Override

                                    ;test   byte [bp-5],1
000070F9 F646FB01                   test    Exec_Func,exec_func_no_execute
000070FD 7427                       jz      short exec_go

000070FF 36C536[BF0E]               lds     SI,[SS:exec_init_SP]  ; get stack SS Override
                                    ;les    di,[bp-4]
00007104 C47EFC                     les     DI,Exec_Blk           ; and block for return
                                    ;mov    [es:di+10h],ds
00007107 268C5D10                   mov     [ES:DI+EXEC1.SS],DS   ; return SS

0000710B 4E                         dec     SI                    ; 'push' default AX
0000710C 4E                         dec     SI
0000710D 891C                       mov     [SI],BX               ; save default AX reg
                                    ;mov    [es:di+0Eh], si
0000710F 2689750E                   mov     [ES:DI+EXEC1.SP],SI   ; return 'SP'

00007113 36C506[C30E]               lds     AX,[SS:exec_init_IP]  ; SS Override
                                    ;mov    [es:di+14h],ds
00007118 268C5D14                   mov     [ES:DI+EXEC1.CS],DS   ; initial entry stuff
                                    ;mov    [es:di+12h],ax
0000711C 26894512                   mov     [ES:DI+EXEC1.IP],AX

                                    ;leave
00007120 89EC                       mov     sp,bp
00007122 5D                         pop     bp

                                    ;transfer SYS_RET_OK
00007123 E94895                     jmp     SYS_RET_OK

                                exec_go:
00007126 36C536[C30E]               lds     SI,[SS:exec_init_IP]  ; get entry point SS Override
0000712B 36C43E[BF0E]               les     DI,[SS:exec_init_SP]  ; new stack SS Override
00007130 8CC0                       mov     AX,ES

                                    ; MSDOS 6.0
00007132 36803E[660D]00             cmp     byte [SS:DosHasHMA],0 ; Q: is dos in HMA (M021)
00007138 741A                       je      short Xfer_To_User    ; N: transfer control to user

0000713A 1E                         push    ds                    ; Y: control must go to low mem stub

0000713B 2E8E1E[0700]               mov     ds,[cs:DosDSeg]               ;  where we disable a20 and Xfer
                                                                  ;  control to user
00007140 800E[8600]04               or      byte [DOS_FLAG],EXECA20OFF ; M068:
                                                                  ; M004: Set bit to signal int 21
                                                                  ; ah = 25 & ah= 49. See dossym.inc
                                                                  ; under TAG M003 & M009 for
                                                                  ; explanation
00007145 8916[6300]                 mov     [A20OFF_PSP],dx               ; M068: set the PSP for which A20 is
                                                                  ; M068: going to be turned OFF.

00007149 8CD8                       mov     ax,ds                 ; ax = segment of low mem stub
0000714B 1F                         pop     ds

0000714C 50                         push    ax                    ; ret far into the low mem stub
0000714D B8[2410]                   mov     ax,disa20_xfer
00007150 50                         push    ax
00007151 8CC0                       mov     AX,ES                 ; restore ax
00007153 CB                         retf

                                Xfer_To_User:
                                    ; DS:SI points to entry point
                                    ; AX:DI points to initial stack
                                    ; DX has PDB pointer
                                    ; BX has initial AX value

00007154 FA                         cli
                                    ; 15/08/2018
00007155 36C606[2103]00             mov     BYTE [SS:INDOS],0     ; SS Override
                                    ; 01/07/2024
0000715B 36C606[B812]00             mov     byte [ss:INDOS_FLAG],0

00007161 8ED0                       mov     SS,AX                 ; set up user's stack
00007163 89FC                       mov     SP,DI                 ; and SP
00007165 FB                         sti

00007166 1E                         push    DS                    ; fake long call to entry
00007167 56                         push    SI
00007168 8EC2                       mov     ES,DX                 ; set up proper seg registers
0000716A 8EDA                       mov     DS,DX
0000716C 89D8                       mov     AX,BX                 ; set up proper AX

0000716E CB                         retf

                                ; 04/08/2018 - Retro DOS v3.0

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

                                ExecRead:
```

```
41527 0000716F E81600                   CALL    Exec_Dealloc
41528                                   ;mov    bx,[bp-8]
41529 00007172 8B5EF8                   MOV     bx,Exec_FH
41530
41531 00007175 55                       PUSH    BP
41532 00007176 E8FB06                   call    _$READ
41533 00007179 5D                       POP     BP
41534
41535                                   ;CALL   Exec_Alloc
41536                                   ;retn
41537                                   ; 18/12/2022
41538                                   ;jmp    short Exec_Alloc
41539
41540                           ; 18/12/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
41541
41542                           ;--------------------------------------------------------------------------
41543                           ;
41544                           ;--------------------------------------------------------------------------
41545
41546                           Exec_Alloc:
41547 0000717A 53                       push    BX
41548                                   ;mov    BX,[CS:CurrentPDB]  ; MSDOS 3.3
41549                                   ; 20/05/2019 - Retro DOS v4.0
41550                                   ; MSDOS 6.0
41551 0000717B 368B1E[3003]             mov     bx,[SS:CurrentPDB]  ; SS Override
41552 00007180 E81000                   call    ChangeOwners
41553 00007183 E889A7                   call    LCritMEM
41554 00007186 5B                       pop     BX
41555 00007187 C3                       retn
41556
41557                           ;--------------------------------------------------------------------------
41558                           ;
41559                           ;--------------------------------------------------------------------------
41560
41561                           Exec_Dealloc:
41562 00007188 53                       push    BX
41563                                   ;mov    bx,0
41564 00007189 29DB                     sub     BX,BX           ; (bx) = ARENA_OWNER_SYSTEM
41565 0000718B E854A7                   call    ECritMEM
41566 0000718E E80200                   call    ChangeOwners
41567 00007191 5B                       pop     BX
41568 00007192 C3                       retn
41569
41570                           ; 18/12/2022
41571                           %if 0
41572                           ;--------------------------------------------------------------------------
41573                           ;
41574                           ;--------------------------------------------------------------------------
41575
41576                           Exec_Alloc:
41577                               push    BX
41578                               ;mov    BX,[CS:CurrentPDB]  ; MSDOS 3.3
41579                               ; 20/05/2019 - Retro DOS v4.0
41580                               ; MSDOS 6.0
41581                               mov     bx,[SS:CurrentPDB]  ; SS Override
41582                               call    ChangeOwners
41583                               call    LCritMEM
41584                               pop     BX
41585                               retn
41586
41587                           %endif
41588
41589                           ;--------------------------------------------------------------------------
41590                           ;
41591                           ;--------------------------------------------------------------------------
41592
41593                           ChangeOwners:
41594 00007193 9C                       pushf
41595 00007194 50                       push    AX
41596                                   ;mov    ax,[bp-14]
41597 00007195 8B46F2                   mov     AX,Exec_Environ
41598 00007198 E80900                   call    ChangeOwner
41599                                   ;mov    ax,[bp-18]
41600 0000719B 8B46EE                   mov     AX,Exec_Load_Block
41601 0000719E E80300                   call    ChangeOwner
41602 000071A1 58                       pop     AX
41603 000071A2 9D                       popf
41604                           chgown_retn:
41605 000071A3 C3                       retn
41606
41607                           ;--------------------------------------------------------------------------
41608                           ;
41609                           ;--------------------------------------------------------------------------
41610
41611                           ChangeOwner:
41612 000071A4 09C0                     or      AX,AX                   ; is area allocated?
41613 000071A6 74FB                     jz      short chgown_retn       ; no, do nothing
41614 000071A8 48                       dec     AX
41615 000071A9 1E                       push    DS
41616 000071AA 8ED8                     mov     DS,AX
41617 000071AC 891E0100                 mov     [ARENA.OWNER],BX
41618 000071B0 1F                       pop     DS
41619 000071B1 C3                       retn
41620
41621                           ;--------------------------------------------------------------------------
41622                           ;
41623                           ;--------------------------------------------------------------------------
41624
41625                           ; 20/05/2019 - Retro DOS v4.0
41626
41627                               ; MSDOS 6.0
41628                           Scan_Execname:
41629                               ;lds    si,[bp-26] ; 08/03/2024
41630 000071B2 C576E6                   lds     SI,ExecName             ; DS:SI points to name
41631                           Scan_Execname1:                             ; M028
41632                           Save_Begin:                             ;
41633 000071B5 89F1                     mov     CX,SI                   ; CX= starting addr
41634                           Scan0:                                    ;
41635 000071B7 AC                       lodsb                           ; get char
41636
41637 000071B8 3C3A                     cmp     AL,':'                  ; is ':' , may be A:name
41638 000071BA 74F9                     jz      short Save_Begin        ; yes, save si
41639 000071BC 3C5C                     cmp     AL,'\'                    ; is '\', may be A:\name
41640 000071BE 74F5                     jz      short Save_Begin        ; yes, save si
41641 000071C0 3C00                     cmp     AL,0                    ; is end of name
41642 000071C2 75F3                     jnz     short Scan0             ; no, continue scanning
41643 000071C4 29CE                     sub     SI,CX                   ; get name's length
41644 000071C6 87F1                     xchg    SI,CX                   ; cx= length, si= starting addr
41645
41646 000071C8 C3                       retn
41647
41648                           ;--------------------------------------------------------------------------
41649                           ;
41650                           ;--------------------------------------------------------------------------
```

```
                                ; 20/05/2019 - Retro DOS v4.0
                                ; 01/12/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
                                ; DOSCODE:A0EDh (MSDOS 5.0, MSDOS.SYS)
                                ; 09/03/2024 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMDOS.COM)
                                ; DOSCODE:B370h (PCDOS 7.1, IBMDOS.COM)
                                    ; MSDOS 6.0
                                Scan_Special_Entries:
000071C9 49                         dec     CX                      ; cx= name length
                                ;M060       mov     DI,[Special_Entries]  ; es:di -> addr of special entries
                                                                    ;reset to current version
                                    ;mov    word [ss:SPECIAL_VERSION],1406h
                                                                    ; (MSDOS 6.21, MSDOS.SYS, DOSCODE:A14Eh)
                                    ;mov    word [ss:SPECIAL_VERSION],5
                                                                    ; (MSDOS 5.0, MSDOS.SYS, DOSCODE:A0EEh)
                                                                    ; 5 for Retro DOS 4.0 (01/12/2022, MSDOS 5.0)
000071CA 36C706[BB0E]070A           mov     word [ss:SPECIAL_VERSION],(MINOR_VERSION<<8)+MAJOR_VERSION
                                                                    ; 0005h for Retro DOS v4.1 (MSDOS 5.0)
                                                                    ; 24/09/2023
                                                                    ; 1606h for Retro DOS v4.2 (MSDOS 6.22)
                                                                    ; 09/03/2024
                                                                    ; 0A07h for Retro DOS v5.0 (PCDOS 7.1)
                                                                    ; (0008h for Windows ME)
                                ;***call    Reset_Version
                                ;M060       push    SS
                                ;M060       pop     ES
000071D1 36C43E[5D00]               les     DI,[SS:UU_IFS_DOS_CALL]     ;M060; ES:DI --> Table in SETVER.SYS
000071D6 8CC0                       mov     AX,ES                   ;M060; First do a NULL ptr check to
000071D8 09F8                       or      AX,DI                   ;M060; be sure the table exists
000071DA 7427                       jz      short End_List          ;M060; If ZR then no table
                                GetEntries:
000071DC 268A05                     mov     AL,[ES:DI]              ; end of list
000071DF 08C0                       or      AL,AL
000071E1 7420                       jz      short End_List          ; yes
000071E3 36893E[0E06]               mov     [ss:TEMP_VAR2],DI       ; save di
000071E8 38C8                       cmp     AL,CL                   ; same length ?
000071EA 751B                       jnz     short SkipOne           ; no
000071EC 47                         inc     DI                      ; es:di -> special name
000071ED 51                         push    CX                      ; save length and name addr
000071EE 56                         push    SI
                                ; M050 - BEGIN
000071EF 50                         push    ax                      ; save len
                                sse_next_char:
000071F0 AC                         lodsb
000071F1 E8A3EB                     call    UCase
000071F4 AE                         scasb
000071F5 750D                       jne     short Not_Matched
000071F7 E2F7                       loop    sse_next_char
                                ;   repz    cmpsb                   ; same name ?
                                ;   jnz     short Not_Matched       ; no
                                    ; 09/03/2024 - PCDOS 7.1 IBMDOS.COM
                                    ;pop    ax                      ; take len off the stack
                                ; M050 - END
000071F9 268B05                     mov     AX,[ES:DI]              ; get special version
000071FC 36A3[BB0E]                 mov     [ss:SPECIAL_VERSION],AX       ; save it
                                ;***mov     AL,[ES:DI+2]            ; get fake count
                                ;***mov     [ss:FAKE_COUNT],AL      ; save it
                                    ; 09/03/2024 - PCDOS 7.1 IBMDOS.COM
00007200 58                         pop     ax                      ; take len off the stack
00007201 5E                         pop     SI
00007202 59                         pop     CX
                                    ; 18/12/2022
                                    ;jmp    SHORT End_List
                                    ; 18/12/2022
                                End_List:
00007203 C3                         retn
                                Not_Matched:
00007204 58                         pop     ax                      ; get len from stack ; M050
00007205 5E                         pop     SI                      ; restore si,cx
00007206 59                         pop     CX
                                SkipOne:
00007207 368B3E[0E06]               mov     DI,[ss:TEMP_VAR2]       ; restore old di use SS Override
0000720C 30E4                       xor     AH,AH                   ; position to next entry
0000720E 01C7                       add     DI,AX
00007210 83C703                     add     DI,3                    ; DI -> next entry length
                                ;***add     DI,4                    ; DI -> next entry length
00007213 EBC7                       jmp     short GetEntries
                                    ; 18/12/2022
                                ;End_List:
                                    ;retn
                                ; 04/08/2018 - Retro DOS v3.0
                                ; IBMDOS.COM (MSDOS 3.3, 1987) - Offset 633Dh
                                ;--------------------------------------------------------------------------
                                ;SUBTTL Terminate and stay resident handler
                                ;
                                ; Input:    DX is an offset from CurrentPDB at which to
                                ;       truncate the current block.
                                ;
                                ; output:   The current block is truncated (expanded) to be [DX+15]/16
                                ;       paragraphs long. An exit is simulated via resetting CurrentPDB
                                ;       and restoring the vectors.
                                ;
                                ;--------------------------------------------------------------------------
```

```
                                    ; 20/05/2019 - Retro DOS v4.0
                                    ; DOSCODE:A19Bh (MSDOS 6.21, MSDOS.SYS)

                                    ; 01/12/2022 - Retro DOS v4.0 (Modified MSDOS5.0 MSDOS.SYS)
                                    ; DOSCODE:A13Bh (MSDOS 5.0, MSDOS.SYS)

                                    ; 09/03/2024 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMDOS.COM)
                                    ; PCDOS 7.1 IBMDOS.COM - DOSCODE:0B3BCh

                                _$KEEP_PROCESS:
00007215 50                         push    AX                      ; keep exit code around
                                    ;mov    byte [SS:EXIT_TYPE],3
00007216 36C606[7C05]03             mov     BYTE [SS:EXIT_TYPE],EXIT_KEEP_PROCESS
0000721C 368E06[3003]               mov     ES,[SS:CurrentPDB]
00007221 83FA06                     cmp     DX,6h                   ; keep enough space around for system
00007224 7303                       jae     short Keep_Shrink       ; info

00007226 BA0600                     mov     DX,6h

                                Keep_Shrink:
00007229 89D3                       mov     BX,DX
0000722B 53                         push    BX
0000722C 06                         push    ES
0000722D E82F02                     call    _$SETBLOCK              ; ignore return codes.
00007230 1F                         pop     DS
00007231 5B                         pop     BX
00007232 7207                       jc      short Keep_Done                 ; failed on modification

00007234 8CD8                       mov     AX,DS
00007236 01D8                       add     AX,BX
                                    ;mov    [2],ax
00007238 A30200                     mov     [PDB.BLOCK_LEN],AX      ;PBUGBUG

                                Keep_Done:
0000723B 58                         pop     AX
0000723C EB26                       jmp     SHORT exit_inner        ; and let abort take care of the rest

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

                                STAY_RESIDENT:
                                    ;mov    ax,3100h
0000723E B80031                     mov     AX,(KEEP_PROCESS<<8)+0 ; Lower part is return code;PBUGBUG
00007241 83C20F                     add     DX,15
00007244 D1DA                       rcr     DX,1
00007246 B103                       mov     CL,3
00007248 D3EA                       shr     DX,CL

0000724A E9A490                     jmp     COMMAND

                                ;--------------------------------------------------------------------------
                                ;SUBTTL $EXIT - return to parent process
                                ;   Assembler usage:
                                ;       MOV     AL, code
                                ;       MOV     AH, Exit
                                ;       INT     int_command
                                ;   Error return:
                                ;       None.
                                ;
                                ;--------------------------------------------------------------------------

                                    ; 20/05/2019 - Retro DOS v4.0
                                    ; DOSCODE:A1D3h (MSDOS 6.21, MSDOS.SYS)

                                    ; 01/12/2022 - Retro DOS v4.0 (Modified MSDOS5.0 MSDOS.SYS)
                                    ; DOSCODE:A173h (MSDOS 5.0, MSDOS.SYS)

                                    ; 09/03/2024 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMDOS.COM)
                                    ; PCDOS 7.1 IBMDOS.COM - DOSCODE:0B3F4h
                                _$EXIT:
                                    ; 04/08/2018 - Retro DOS v3.0
                                    ; IBMDOSDOS.COM (MSDOS 3.3, 1987) - Offset 6375h
0000724D 30E4                       xor     AH,AH
0000724F 368626[4D03]               xchg    AH,[SS:DidCTRLC]
00007254 08E4                       or      AH,AH
                                    ;mov    BYTE [SS:EXIT_TYPE],0
00007256 36C606[7C05]00             mov     BYTE [SS:EXIT_TYPE],EXIT_TERMINATE
0000725C 7406                       jz      short exit_inner
                                    ;mov    BYTE [SS:EXIT_TYPE],1
0000725E 36C606[7C05]01             mov     BYTE [SS:EXIT_TYPE],EXIT_CTRL_C

                                    ;entry  Exit_inner
                                exit_inner:
00007264 E81092                     call    Get_User_Stack          ;PBUGBUG

00007267 36FF36[3003]               push    word [ss:CurrentPDB]
                                    ;pop    word [si+14h]
0000726C 8F4414                     pop     word [SI+user_env.user_CS] ;PBUGBUG
0000726F EB08                       jmp     short abort_inner

                                ;BREAK <$ABORT -- Terminate a process>
                                ;--------------------------------------------------------------------------
                                ; Inputs:
                                ;   user_CS:00 must point to valid program header block
                                ; Function:
                                ;   Restore terminate and Cntrl-C addresses, flush buffers and transfer
                                ;   to the terminate address
                                ; Returns:
                                ;   TO THE TERMINATE ADDRESS
                                ;--------------------------------------------------------------------------

                                _$ABORT:
00007271 30C0                       xor     AL,AL
                                    ;mov    byte [SS:EXIT_TYPE],0
                                    ;mov    byte [SS:EXIT_TYPE],AL ; = 0
00007273 36C606[7C05]00             mov     byte [SS:EXIT_TYPE],EXIT_ABORT

                                    ; abort_inner must have AL set as the exit code! The exit type
                                    ; is retrieved from exit_type. Also, the PDB at user_CS needs
                                    ; to be correct as the one that is terminating.

                                abort_inner:
00007279 368A26[7C05]               mov     AH,[SS:EXIT_TYPE]
0000727E 36A3[3403]                 mov     [SS:exit_code],AX
00007282 E8F291                     call    Get_User_Stack

                                    ;mov    ds,[si+14h]
00007285 8E5C14                     mov     DS,[SI+user_env.user_CS] ; set up old interrupts ;PBUGBUG
00007288 31C0                       xor     AX,AX
0000728A 8EC0                       mov     ES,AX
                                    ;mov    si,10
0000728C BE0A00                     mov     SI,SAVEXIT
                                    ;mov    di,88h
```

```
0000728F BF8800                     mov     DI,addr_int_terminate
00007292 A5                         movsw
00007293 A5                         movsw
00007294 A5                         movsw
00007295 A5                         movsw
00007296 A5                         movsw
00007297 A5                         movsw
00007298 E954EF                     jmp     reset_environment

                                ;--------------------------------------------------------------------------
                                ;
                                ; fixexepatch will point to this is DOS loads low.
                                ;
                                ;--------------------------------------------------------------------------
                                ; MSDOS 6.0

                                ; 29/04/2019 - Retro DOS v4.0
                                ; DOSCODE:A221h (MSDOS 6.21, MSDOS.SYS)

                                ; 01/12/2022 - Retro DOS v4.0 (Modified MSDOS5.0 MSDOS.SYS)
                                ; DOSCODE:A1C1h (MSDOS 5.0, MSDOS.SYS)

                                    ; 09/03/2024 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMDOS.COM)
                                    ; PCDOS 7.1 IBMDOS.COM - DOSCODE:0B442h

                                RetExePatch: ; proc near

0000729B C3                         retn

                                ;============================================================================
                                ; ALLOC.ASM, MSDOS 6.0, 1991
                                ;============================================================================
                                ; 04/08/2018 - Retro DOS v3.0
                                ; 14/05/2019 - Retro DOS v4.0
                                ; 10/03/2024 - Retro DOS v5.0

                                ;   TITLE ALLOC.ASM - memory arena manager      NAME Alloc

                                ;**
                                ;   Microsoft Confidential
                                ;   Copyright (C) Microsoft Corporation 1991
                                ;   All Rights Reserved.
                                ;
                                ;   Memory related system calls and low level routines for MSDOS 2.X.
                                ;   I/O specs are defined in DISPATCH.
                                ;
                                ;   $ALLOC
                                ;   $SETBLOCK
                                ;   $DEALLOC
                                ;   $AllocOper
                                ;   arena_free_process
                                ;   arena_next
                                ;   check_signature
                                ;   Coalesce
                                ;
                                ;   Modification history:
                                ;
                                ;       Created: ARR 30 March 1983
                                ;
                                ;       Revision: M000 - added support for allocing UMBs. 7/9/90
                                ;                 M003 - added support for link/unlink UMBs from
                                ;                        DOS arena chain. 7/18/90
                                ;                 M009 - Added error returns invalid function and
                                ;                        arena trashed in set link state call.
                                ;                 M010 - Release UMB arenas allocated to current PDB
                                ;                        if UMB_HEAD is initialized.
                                ;
                                ;                 M016 - MACE utilities mkeyrate.com version 1.0
                                ;                        support. Please see under M009 in
                                ;                        ..\inc\dossym.inc. 8/31/90.
                                ;
                                ;                 M061 - In GetLastArena, if linking in UMBs check to make
                                ;                        sure that umb_head arena is valid and also make
                                ;                        sure that the previous arena is pointing to
                                ;                        umb_head.
                                ;
                                ;                 M064 - allow HIGH_ONLY bit to be set by a call to
                                ;                        set allloc strategy.
                                ;                        use STRAT_MASK to mask out bits 6 & 7 of
                                ;                        bx in AllocSetStrat.
                                ;
                                ;                 M068 - use a count value (A20OFF_COUNT) rather than
                                ;                        a bit to indicate to dos dispatcher to turn
                                ;                        a20 off before iret. See M016.
                                ;

                                ;   BREAK   <memory allocation utility routines>


                                ; 15/04/2018 - Retro DOS v2.0
                                ;--------------------------------------------------------------------------
                                ; xenix memory calls for MSDOS
                                ;
                                ; CAUTION: The following routines rely on the fact that arena_signature and
                                ; arena_owner_system are all equal to zero and are contained in DI.
                                ;
                                ;INCLUDE DOSSEG.ASM

                                ;CODE       SEGMENT BYTE PUBLIC  'CODE'
                                ;       ASSUME  SS:DOSGROUP,CS:DOSGROUP

                                ;.xlist
                                ;.xcref
                                ;INCLUDE DOSSYM.ASM
                                ;INCLUDE DEVSYM.ASM
                                ;.cref
                                ;.list

                                ;TITLE ALLOC.ASM - memory arena manager
                                ;NAME Alloc

                                ;SUBTTL memory allocation utility routines
                                ;PAGE
                                ;
                                ; arena data
                                ;
                                ;       i_need  arena_head,WORD         ; seg address of start of arena
                                ;       i_need  CurrentPDB,WORD         ; current process data block addr
                                ;       i_need  FirstArena,WORD         ; first free block found
                                ;       i_need  BestArena,WORD          ; best free block found
                                ;       i_need  LastArena,WORD          ; last free block found
                                ;       i_need  AllocMethod,BYTE        ; how to alloc first(best)last

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

```
42023
42024                                   ; 10/03/2024 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMDOS.COM)
42025                                   ; PCDOS 7.1 IBMDOS.COM - DOSCODE:0B443h
42026                                   ;;;
42027                                 test_umb_flag:
42028 0000729C 36F606[8900]01           test   byte [ss:UMBFLAG],LINKSTATE ; 1 ; Q: are umb's linked
42029 000072A2 C3                       retn                    ; ZF=1 -> N: scan from arena_head
42030                                                           ; ZF=0 -> Y: start_arena = umb_head
42031                                   ;;;
42032
42033                                 ;** Arena_Free_Process
42034                                 ;-----------------------------------------------------------------------------
42035                                 ;   Free all arena blocks allocated to a process
42036                                 ;
42037                                 ;   ENTRY   (bx) = PID of process
42038                                 ;   EXIT    none
42039                                 ;   USES    ????? BUGBUG
42040                                 ;-----------------------------------------------------------------------------
42041
42042                                   ; 01/12/2022 - Retro DOS v4.0 (Modified MSDOS5.0 MSDOS.SYS)
42043                                   ; DOSCODE:A1C2h (MSDOS 5.0, MSDOS.SYS)
42044
42045                                   ; 10/03/2024 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMDOS.COM)
42046                                   ; PCDOS 7.1 IBMDOS.COM - DOSCODE:0B44Ah
42047
42048                                 arena_free_process:
42049                                   ; 14/05/2019 - Retro DOS v4.0
42050                                   ; 04/08/2018 - Retro DOS v3.0
42051 000072A3 36A1[2400]                 MOV AX,[SS:arena_head]
42052                                 arena_free_process_start:
42053 000072A7 BF0000                   MOV     DI,ARENA.SIGNATURE ; 0
42054                                   ;MOV    AX,[SS:arena_head] ; 15/04/2018
42055 000072AA E82F00                       CALL       check_signature         ; ES <- AX, check for valid block
42056
42057                                 arena_free_process_loop:
42058                                       ;retc
42059 000072AD 7225                       JC  SHORT AFP_RETN ; Retro DOS v2.0 - 05/03/2018
42060 000072AF 06                       PUSH    ES
42061 000072B0 1F                           POP     DS
42062                                   ;cmp    [1],bx
42063 000072B1 391E0100                     CMP     [ARENA.OWNER],BX  ; is block owned by pid?
42064 000072B5 7504                         JNZ     SHORT arena_free_next    ; no, skip to next
42065                                   ;mov    [1],di
42066 000072B7 893E0100                     MOV     [ARENA.OWNER],DI  ; yes... free him
42067
42068                                 arena_free_next:
42069                                   ;cmp    byte [di],5Ah ;'Z'
42070 000072BB 803D5A                       CMP     BYTE [DI],arena_signature_end
42071                                                                 ; end of road, Jack?
42072                                       ;retz                     ; never come back no more
42073                                   ;JZ     SHORT AFP_RETN  ; MSDOS 3.3 (& MSDOS 2.11)
42074                                   ; 14/05/2019
42075                                   ; MSDOS 6.0
42076 000072BE 7405                     jz      short arena_chk_umbs
42077
42078 000072C0 E81200                   CALL    arena_next              ; next item in ES/AX carry set if trash
42079 000072C3 EBE8                         JMP     SHORT arena_free_process_loop
42080
42081                                   ; MSDOS 6.0
42082                                 arena_chk_umbs:                         ; M010 - Start
42083                                   ; 20/05/2019
42084 000072C5 36A1[8C00]               mov     ax,[ss:UMB_HEAD]      ; ax = umb_head
42085 000072C9 83F8FF                   cmp     ax,0FFFFh             ; Q: is umb_head initialized
42086 000072CC 741D                     je      short ret_label               ; N: we're done
42087
42088 000072CE 8CDF                     mov     di,ds                 ; di = last arena
42089 000072D0 39C7                     cmp     di,ax                 ; Q: is last arena above umb_head
42090                                   ;jae    short ret_label               ; Y: we've scanned umbs also. done.
42091                                   ;jmp    short arena_free_process_start
42092                                                                 ; M010 - End
42093                                   ; 10/03/2024 (PCDOS 7.1 IBMDOS.COM)
42094 000072D2 72D3                     jb      short arena_free_process_start
42095
42096                                   ; 10/03/2024
42097                                 AFP_RETN:
42098 000072D4 C3                       RETN
42099
42100                                 ;   BREAK   <Arena Helper Routines>
42101
42102                                 ;** Arena_Next - Find Next item in Arena
42103                                 ;-----------------------------------------------------------------------------
42104                                 ;   ENTRY   DS - pointer to block head
42105                                 ;           (di) = 0
42106                                 ;   EXIT    AX,ES - pointers to next head
42107                                 ;           'C' set iff arena damaged
42108                                 ;-----------------------------------------------------------------------------
42109
42110                                 arena_next:
42111 000072D5 8CD8                         MOV     AX,DS                   ; AX <- current block
42112 000072D7 03060300                     ADD     AX,[ARENA.SIZE]          ; AX <- AX + current block length
42113 000072DB 40                           INC     AX                      ; remember that header!
42114
42115                                 ;       fall into check_signature and return
42116                                 ;
42117                                 ;       CALL    check_signature         ; ES <- AX, carry set if error
42118                                 ;       RETN
42119
42120                                 ;** Check_Signature - Check Memory Block Signature
42121                                 ;-----------------------------------------------------------------------------
42122                                 ;   ENTRY   (AX) = address of block header
42123                                 ;           (di) = 0
42124                                 ;   EXIT    ES = AX
42125                                 ;           'C' clear if signature good
42126                                 ;           'C' set if signature bad
42127                                 ;   USES    ES, Flags
42128                                 ;-----------------------------------------------------------------------------
42129
42130                                 check_signature:
42131
42132 000072DC 8EC0                       MOV     ES,AX                   ; ES <- AX
42133                                   ;cmp    byte [es:di],4Dh ; 'M'
42134 000072DE 26803D4D                     CMP     BYTE [ES:DI],arena_signature_normal
42135                                                                 ; IF next signature = not_end THEN
42136 000072E2 7407                         JZ      SHORT check_signature_ok ;   GOTO ok
42137                                   ;cmp    byte [es:di],5Ah ; 'Z'
42138 000072E4 26803D5A                     CMP     BYTE [ES:DI],arena_signature_end
42139                                                                 ; IF next signature = end then
42140 000072E8 7401                         JZ      SHORT check_signature_ok ;   GOTO ok
42141 000072EA F9                           STC                             ; set error
42142                                 ret_label: ; MSDOS 6.0
42143                                 ;AFP_RETN:  ; 10/03/2024
42144                                   ; Retro DOS v2.0 - 05/03/2018
42145                                 check_signature_ok:
42146                                 COALESCE_RETN:
```

```
000072EB C3                         RETN

                                ;** Coalesce - Combine free blocks ahead with current block
                                ;----------------------------------------------------------------------------
                                ;   Coalesce adds the block following the argument to the argument block,
                                ;   iff 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.
                                ;
                                ;   ENTRY   (ds) = pointer to the head of a free block
                                ;           (di) = 0
                                ;   EXIT    'C' clear if OK
                                ;             (ds) unchanged, this block updated
                                ;             (ax) = address of next block, IFF not at end
                                ;           'C' set if arena trashed
                                ;   USES    (cx)
                                ;----------------------------------------------------------------------------

                                Coalesce:
                                    ;cmp    byte [di],5Ah ; 'Z'
000072EC 803D5A                     CMP     BYTE [DI],arena_signature_end
                                                                    ; IF current signature = END THEN
                                        ;retz                       ;   GOTO ok
000072EF 74FA                       jz  short COALESCE_RETN
000072F1 E8E1FF                     CALL    arena_next              ; ES, AX <- next block, Carry set if error
                                        ;retc                       ; IF no error THEN GOTO check
000072F4 72F5                       jc      short COALESCE_RETN

                                coalesce_check:
                                    ;cmp    [es:1],di
000072F6 26393E0100                     CMP     [ES:ARENA.OWNER],DI
                                        ;retnz                      ; IF next block isnt free THEN return
000072FB 75EE                           JNZ SHORT COALESCE_RETN
                                    ;mov    cx,[ES:3]
000072FD 268B0E0300                 MOV     CX,[ES:ARENA.SIZE]    ; CX <- next block size
00007302 41                             INC     CX                  ; CX <- CX + 1 (for header size)
                                        ;ADD       [3],CX
00007303 010E0300                   ADD     [ARENA.SIZE],CX         ; current size <- current size + CX
00007307 268A0D                         MOV     CL,[ES:DI]          ; move up signature
0000730A 880D                           MOV     [DI],CL
0000730C EBDE                           JMP     SHORT Coalesce       ; try again

                                ; 04/08/2018 - Retro DOS v3.0
                                ; IBMDOS.COM (MSDOS 3.3, 1987) - Offset 641Fh

                                ;   BREAK  <$Alloc - allocate space in memory>

                                ; MSDOS 6.0
                                ;----------------------------------------------------------------------------
                                ;** $Alloc - Allocate Memory Space
                                ;
                                ;   $Alloc services the INT21 that allocates memory space to a program.
                                ;   Alloc returns a pointer to a free block of memory that
                                ;   has the requested size in paragraphs.
                                ;
                                ;   If the allocation strategy is HIGH_FIRST or HIGH_ONLY memory is
                                ;   scanned from umb_head if not from arena_head. If the strategy is
                                ;   HIGH_FIRST the scan is continued from arena_head if a block of
                                ;   appropriate size is not found in the UMBs. If the strategy is
                                ;   HIGH_FIRST+HIGH_ONLY only the UMBs are scanned for memory.
                                ;
                                ;   In either case if bit 0 of UmbFlag is not initialized then the scan
                                ;   starts from arena_head.
                                ;
                                ;   Assembler usage:
                                ;           MOV     BX,size
                                ;           MOV     AH,Alloc
                                ;           INT     21h
                                ;
                                ;   BUGBUG - a lot can be done to improve performance. We can set marks
                                ;   so that we start searching the arena at it's first non-trivial free
                                ;   block, we can peephole the code, etc. (We can move some subr calls
                                ;   inline, etc.) I assume that this is called rarely and that the arena
                                ;   doesn't have too many memory objects in it beyond the first free one.
                                ;   verify that this is true; if so, this can stay as is
                                ;
                                ;   ENTRY   (bx) = requested size, in bytes
                                ;           (DS) = (ES) = DOSGROUP
                                ;   EXIT    'C' clear if memory allocated
                                ;             (ax:0) = address of requested memory
                                ;           'C' set if request failed
                                ;             (AX) = error_not_enough_memory
                                ;               (bx) = max size we could have allocated
                                ;             (ax) = error_arena_trashed
                                ;   USES    All
                                ;----------------------------------------------------------------------------

                                ; MSDOS 2.11 (& MSDOS 3.3)
                                ;----------------------------------------------------------------------------
                                ;SUBTTL $Alloc - allocate space in memory
                                ;
                                ;   Assembler usage:
                                ;           MOV     BX,size
                                ;           MOV     AH,Alloc
                                ;           INT     21h
                                ;         AX:0 is pointer to allocated memory
                                ;         BX is max size if not enough memory
                                ;
                                ;   Description:
                                ;           Alloc returns  a  pointer  to  a  free  block of
                                ;       memory that has the requested  size  in  paragraphs.
                                ;
                                ;   Error return:
                                ;           AX = error_not_enough_memory
                                ;              = error_arena_trashed
                                ;----------------------------------------------------------------------------

                                ; DOSCODE:A28Eh (MSDOS 6.21, MSDOS.SYS)

                                ; 01/12/2022 - Retro DOS v4.0 (Modified MSDOS5.0 MSDOS.SYS)
                                ; DOSCODE:A22Eh (MSDOS 5.0, MSDOS.SYS)

                                    ; 10/03/2024 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMDOS.COM)
                                    ; PCDOS 7.1 IBMDOS.COM - DOSCODE:0B4B5h

                                _$ALLOC:
                                    ; 25/05/2019 (Procedure has been checked and confirmed)
                                    ; 14/05/2019 - Retro DOS v4.0
                                    ; 04/08/2018 - Retro DOS v3.0
                                    ;EnterCrit critMem
0000730E E8D1A5                     call    ECritMEM ; MSDOS 3.3 & MSDOS 6.0

                                ; 17/12/2022
                                ; 01/12/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
```

```
                                ;%if 0
                                    ; 14/05/2019
00007311 16                         push    ss
00007312 1F                         pop     ds

                                    ; MSDOS 6.0
                                    ;mov    ax,[ss:arena_head]
                                    ;mov    [ss:START_ARENA],ax   ; assume LOW_FIRST

00007313 A1[2400]                   mov     ax,[arena_head]
00007316 A3[8E00]                   mov     [START_ARENA],ax

                                    ;test   byte [ss:AllocMethod],HIGH_FIRST+HIGH_ONLY
00007319 F606[0203]C0               test    byte [AllocMethod],HIGH_FIRST+HIGH_ONLY
                                                                  ; Q: should we start scanning from
                                                                  ;    UMB's
0000731E 740B                       jz      short norm_alloc      ; N: scan from arena_head

                                ; 10/03/2024
                                %if 0
                                    ;;cmp   word [ss:UMB_HEAD],-1 ; Q: Has umb_head been initialized
                                    ;cmp    word [UMB_HEAD],-1
                                    ;je     short norm_alloc      ; N: scan from arena_head

                                    ;test   byte [ss:UMBFLAG],LINKSTATE ; Q: are umb's linked
                                    test    byte [UMBFLAG],LINKSTATE ; 1
                                    jz      short norm_alloc      ; N: scan from arena_head
                                %else
                                    ; 10/03/2024 (PCDOS 7.1 IBMDOS.COM)
                                    ;;;
00007320 E879FF                     call    test_umb_flag
00007323 7406                       jz       short norm_alloc
                                    ;;;
                                %endif

                                    ;mov    ax,[ss:UMB_HEAD]
                                    ;mov    [ss:START_ARENA],ax   ; start_arena = umb_head
00007325 A1[8C00]                   mov     ax,[UMB_HEAD]
00007328 A3[8E00]                   mov     [START_ARENA],ax
                                                                  ; M000 - end
                                norm_alloc:
0000732B 31C0                           XOR     AX,AX
0000732D 89C7                           MOV     DI,AX
                                    ; 15/03/2018
                                        ;MOV       [SS:FirstArena],AX    ; init the options
                                        ;MOV       [SS:BestArena],AX
                                        ;MOV       [SS:LastArena],AX
                                    ; 14/05/2019
0000732F A3[4003]                   MOV     [FirstArena],AX              ; init the options
00007332 A3[4203]                       MOV [BestArena],AX
00007335 A3[4403]                       MOV [LastArena],AX
00007338 50                             PUSH    AX                      ; alloc_max <- 0
                                    ; 04/08/2018
                                start_scan:
                                    ;MOV    AX,[SS:arena_head]    ; AX <- beginning of arena
                                    ;MOV    AX,[arena_head]

                                    ; 14/05/2019
                                    ; MSDOS 6.0
                                    ;mov    ax,[SS:START_ARENA]   ; M000: AX <- beginning of arena
00007339 A1[8E00]                   mov     ax,[START_ARENA]

                                    ; 27/09/2023 (BugFix) (*)
                                    ; ( jump from 'alloc_chk' (ds<>ss, ax = [SS:START_ARENA]))
                                start_scan_x:

0000733C E89DFF                     CALL    check_signature         ; ES <- AX, carry set if error
0000733F 7233                           JC      SHORT alloc_err         ; IF error THEN GOTO err

                                ;%endif

                                ; 17/12/2022
                                %if 0
                                    ; 01/12/2022 (MSDOS 5.0 MSDOS.SYS compatibility)

                                    ; MSDOS 6.0
                                    mov     ax,[ss:arena_head]
                                    mov     [ss:START_ARENA],ax   ; assume LOW_FIRST

                                    test    byte [ss:AllocMethod],HIGH_FIRST+HIGH_ONLY
                                                                  ; Q: should we start scanning from
                                                                  ;    UMB's
                                    jz      short norm_alloc      ; N: scan from arena_head

                                    ;cmp    word [ss:UMB_HEAD],-1 ; Q: Has umb_head been initialized
                                    ;je     short norm_alloc      ; N: scan from arena_head

                                    test    byte [ss:UMBFLAG],LINKSTATE ; Q: are umb's linked
                                    jz      short norm_alloc      ; N: scan from arena_head

                                    mov     ax,[ss:UMB_HEAD]
                                    mov     [ss:START_ARENA],ax   ; start_arena = umb_head
                                                                  ; M000 - end
                                norm_alloc:
                                        XOR     AX,AX
                                        MOV     DI,AX
                                    ; 15/03/2018
                                    MOV     [SS:FirstArena],AX    ; init the options
                                    MOV     [SS:BestArena],AX
                                    MOV     [SS:LastArena],AX
                                        PUSH    AX                      ; alloc_max <- 0
                                    ; 04/08/2018
                                start_scan:
                                    ;MOV    AX,[SS:arena_head]    ; AX <- beginning of arena
                                    ; 14/05/2019
                                    ; MSDOS 6.0
                                    mov     ax,[SS:START_ARENA]   ; M000: AX <- beginning of arena
                                    CALL    check_signature         ; ES <- AX, carry set if error
                                        JC      SHORT alloc_err         ; IF error THEN GOTO err
                                %endif

                                alloc_scan:
00007341 06                             PUSH    ES
00007342 1F                             POP     DS                      ; DS <- ES
00007343 393E0100                       CMP     [ARENA.OWNER],DI ; 0
00007347 7466                           JZ      SHORT alloc_free  ; IF current block is free THEN examine

                                alloc_next:

                                ; 10/03/2024
                                %if 0
                                    ; MSDOS 6.0                   ; M000 - start
                                    test    byte [ss:UMBFLAG],LINKSTATE ; Q: are umb's linked
                                    jz      short norm_strat      ; N: see if we reached last arena
```

```
                                %else
                                    ; 10/03/2024 (PCDOS 7.1 IBMDOS.COM)
                                    ;;;
00007349 E850FF                     call    test_umb_flag
0000734C 741C                       jz      short norm_strat
                                    ;;;
                                %endif
0000734E 36F606[0203]80             test    byte [ss:AllocMethod],HIGH_FIRST
                                                                    ; Q: is alloc strategy high_first
00007354 7414                       jz      short norm_strat        ; N: see if we reached last arena
00007356 36A1[8E00]                 mov     ax,[ss:START_ARENA]
0000735A 363B06[2400]               cmp     ax,[ss:arena_head]      ; Q: did we start scan from
                                                                    ;    arena_head
0000735F 7509                       jne     short norm_strat        ; N: see if we reached last arena
00007361 8CD8                       mov     ax,ds                   ; ax = current block
00007363 363B06[8C00]               cmp     ax,[ss:UMB_HEAD]        ; Q: check against umb_head
00007368 EB03                       jmp     short alloc_chk_end

                                norm_strat:
                                    ;cmp    byte [di],5Ah ; 'Z'
0000736A 803D5A                         CMP     BYTE [DI],arena_signature_end
                                                                        ; IF current block is last THEN
                                alloc_chk_end:
0000736D 740E                           JZ      SHORT alloc_end          ;   GOTO end
0000736F E863FF                         CALL    arena_next              ; AX, ES <- next block, Carry set if error
00007372 73CD                           JNC     SHORT alloc_scan  ; IF no error THEN GOTO scan

                                alloc_err:
00007374 58                             POP     AX

                                alloc_trashed:
                                    ;LeaveCrit critMem
00007375 E897A5                     call    LCritMEM ; MSDOS 3.3 & MSDOS 6.0

                                alloc_trashed2:    ; 10/03/2024 - Retro DOS v5.0 (PCDOS 7.1) -jmp from GetLastArena-
                                    ;error  error_arena_trashed
                                    ;mov    al,7
00007378 B007                       MOV     AL,error_arena_trashed
                                alloc_errj:
0000737A E9FB92                     JMP     SYS_RET_ERR

                                alloc_end:
                                    ; 18/05/2019
0000737D 36833E[4003]00                 CMP WORD [SS:FirstArena],0
00007383 7403                       jz      short alloc_chk
00007385 E98400                     jmp     alloc_do_split

                                alloc_chk:
                                    ; MSDOS 6.0
00007388 36A1[2400]                 mov     ax,[ss:arena_head]
0000738C 363B06[8E00]               cmp     ax,[ss:START_ARENA]    ; Q: started scanning from arena_head
00007391 740E                       je      short alloc_fail       ; Y: not enough memory
                                                                    ; N:
                                                                    ; Q: is the alloc strat HIGH_ONLY
00007393 36F606[0203]40             test    byte [ss:AllocMethod],HIGH_ONLY
00007399 7506                       jnz     short alloc_fail       ; Y: return size of largest UMB

0000739B 36A3[8E00]                 mov     [ss:START_ARENA],ax    ; N: start scanning from arena_head
                                    ; 27/09/2023 (*)
0000739F EB9B                       jmp     short start_scan_x ; (*) ; (BugFix)
                                    ;jmp    short start_scan
                                                                    ; M000 - end
                                alloc_fail:
                                        ;invoke Get_User_Stack
000073A1 E8D390                         CALL        Get_User_Stack
000073A4 5B                         POP     BX
                                        ;MOV        [SI].user_BX,BX
                                    ;MOV    [SI+2],BX
000073A5 895C02                     mov     [SI+user_env.user_BX],bx
                                    ;LeaveCrit critMem
000073A8 E864A5                     call    LCritMEM ; MSDOS 3.3 & MSDOS 6.0
                                    ;error  error_not_enough_memory
                                    ;mov    al,8
000073AB B008                       MOV     AL,error_not_enough_memory
                                    ; 01/12/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
000073AD EBCB                       jmp     short alloc_errj
                                    ;JMP    SYS_RET_ERR

                                alloc_free:
000073AF E83AFF                         CALL    Coalesce            ; add following free block to current
000073B2 72C0                           JC  SHORT alloc_err                 ; IF error THEN GOTO err
000073B4 8B0E0300                       MOV     CX,[ARENA.SIZE]
000073B8 5A                             POP     DX                  ; check for max found size
000073B9 39D1                           CMP     CX,DX
000073BB 7602                           JNA     SHORT alloc_test
000073BD 89CA                           MOV     DX,CX

                                alloc_test:
000073BF 52                             PUSH    DX
000073C0 39CB                           CMP     BX,CX                   ; IF BX > size of current block THEN
000073C2 7785                       JA      SHORT alloc_next     ;   GOTO next

                                    ; 15/03/2018
000073C4 36833E[4003]00                 CMP     WORD [SS:FirstArena],0
000073CA 7505                       JNZ     SHORT alloc_best
000073CC 368C1E[4003]                   MOV     [SS:FirstArena],DS      ; save first one found
                                alloc_best:
000073D1 36833E[4203]00                 CMP     WORD [SS:BestArena],0
000073D7 740E                           JZ      SHORT alloc_make_best   ; initial best
000073D9 06                             PUSH        ES
000073DA 368E06[4203]                   MOV     ES,[SS:BestArena]
000073DF 26390E0300                     CMP     [ES:ARENA.SIZE],CX      ; is size of best larger than found?
000073E4 07                             POP ES
000073E5 7605                           JBE     SHORT alloc_last
                                alloc_make_best:
000073E7 368C1E[4203]                   MOV     [SS:BestArena],DS ; assign best
                                alloc_last:
000073EC 368C1E[4403]                   MOV     [SS:LastArena],DS       ; assign last
000073F1 E955FF                         JMP     alloc_next
                                ;
                                ; split the block high
                                ;
                                alloc_do_split_high:
000073F4 368E1E[4403]                   MOV     DS,[SS:LastArena]
000073F9 8B0E0300                       MOV     CX,[ARENA.SIZE]
000073FD 29D9                           SUB     CX,BX
000073FF 8CDA                           MOV     DX,DS
00007401 7449                           JE      SHORT alloc_set_owner   ; sizes are equal, no split
00007403 01CA                           ADD     DX,CX                   ; point to next block
00007405 8EC2                           MOV     ES,DX                   ; no decrement!
00007407 49                             DEC     CX
00007408 87D9                           XCHG    BX,CX                   ; bx has size of lower block
0000740A EB2B                           JMP     SHORT alloc_set_sizes   ; cx has upper (requested) size
                                ;
```

```
                                ; we have scanned memory and have found all appropriate blocks
                                ; check for the type of allocation desired; first and best are identical
                                ; last must be split high
                                ;
                                alloc_do_split:

                                ; 17/12/2022
                                ; 01/12/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
                                ;%if 0
                                    ; 14/05/2019
                                    ; MSDOS 6.0                   ; M000 - start
                                    ;xor    cx,cx
0000740C 368A0E[0203]               mov     cl,[ss:AllocMethod]
                                    ;and    cx,STRAT_MASK ; 0FF3Fh; mask off bit 7
00007411 80E13F                     and     cl,3Fh
                                    ;cmp    cx,BEST_FIT ; 1              ; Q; is the alloc strategy best_fit
00007414 80F901                     cmp     cl,BEST_FIT
00007417 77DB                       ja      short alloc_do_split_high
                                ;%endif

                                    ; 17/12/2022
                                    ; 01/12/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
                                    ; MSDOS 6.0 & MSDOS 5.0
                                    ;xor    cx,cx
                                    ;mov    cl,[ss:AllocMethod]
                                    ;and    cx,STRAT_MASK ; 0FF3Fh; mask off bit 7
                                    ;cmp    cx,BEST_FIT ; 1              ; Q; is the alloc strategy best_fit
                                    ;ja     short alloc_do_split_high

                                    ; 15/03/2018
                                        ;;CMP      BYTE [SS:AllocMethod], 1
                                    ; 04/08/2018
                                    ;CMP    BYTE [SS:AllocMethod],BEST_FIT
                                        ;JA SHORT alloc_do_split_high

00007419 368E1E[4003]               MOV     DS,[SS:FirstArena]
0000741E 7205                       JB      SHORT alloc_get_size
00007420 368E1E[4203]               MOV     DS,[SS:BestArena]

                                alloc_get_size:
00007425 8B0E0300                       MOV     CX,[ARENA.SIZE]
00007429 29D9                           SUB     CX,BX                   ; get room left over
0000742B 8CD8                           MOV     AX,DS
0000742D 89C2                           MOV     DX,AX                   ; save for owner setting
0000742F 741B                           JE      SHORT alloc_set_owner    ; IF BX = size THEN (don't split)
00007431 01D8                           ADD     AX,BX
00007433 40                             INC     AX                      ; remember the header
00007434 8EC0                           MOV     ES,AX                   ; ES <- DS + BX (new header location)
00007436 49                             DEC     CX                      ; CX <- size of split block
                                alloc_set_sizes:
00007437 891E0300                       MOV     [ARENA.SIZE],BX          ; current size <- BX
0000743B 26890E0300                     MOV     [ES:ARENA.SIZE],CX      ; split size <- CX
                                    ;mov    bl,4Dh ; 'M'
00007440 B34D                           MOV     BL,arena_signature_normal
00007442 861D                           XCHG    BL,[DI]                 ; current signature <- 4D
00007444 26881D                         MOV     [ES:DI],BL        ; new block sig <- old block sig
00007447 26893E0100                     MOV     [ES:ARENA.OWNER],DI

                                alloc_set_owner:
0000744C 8EDA                           MOV     DS,DX
0000744E 36A1[3003]                     MOV     AX,[SS:CurrentPDB] ; 15/03/2018
00007452 A30100                         MOV     [ARENA.OWNER],AX
00007455 8CD8                           MOV     AX,DS
00007457 40                             INC     AX
00007458 5B                             POP     BX
                                    ;LeaveCrit critMem
00007459 E8B3A4                     call    LCritMEM ; MSDOS 3.3 & MSDOS 6.0

                                    ; 01/12/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
                                alloc_ok:
                                        ;transfer SYS_RET_OK
0000745C E90F92                     JMP     SYS_RET_OK

                                ;   BREAK $SETBLOCK - change size of an allocated block (if possible)

                                ; MSDOS 6.0
                                ;--------------------------------------------------------------------------
                                ;** $SETBLOCK - Change size of an Allocated Block
                                ;
                                ;   Setblock changes the size of an allocated block. First, we coalesce
                                ;   any following free space onto this block; then we try to trim the
                                ;   block down to the size requested.
                                ;
                                ;   Note that if the guy wants to grow the block but that growth fails,
                                ;   we still go ahead and coalesce any trailing free blocks onto it.
                                ;   Thus the maximum-size-possible value that we return has already
                                ;   been allocated! This is a bug, dare we fix it? BUGBUG
                                ;
                                ;   NOTE - $SETBLOCK is in bed with $ALLOC and jumps into $ALLOC to
                                ;          finish it's work. For this reason we build the allocsf
                                ;          structure on the frame, to make us compatible with $ALLOCs
                                ;          code.
                                ;
                                ;   ENTRY   (es) = segment of old block
                                ;           (bx) = newsize
                                ;           (ah) = SETBLOCK
                                ;
                                ;   EXIT    'C' clear if OK
                                ;           'C' set if error
                                ;             (ax) = error_invalid_block
                                ;                  = error_arena_trashed
                                ;                  = error_not_enough_memory
                                ;                  = error_invalid_function
                                ;             (bx) = maximum size possible, iff (ax) = error_not_enough_memory
                                ;   USES    ???? BUGBUG
                                ;--------------------------------------------------------------------------

                                ; MSDOS 2.11 (& MSDOS 3.3)
                                ;--------------------------------------------------------------------------
                                ;SUBTTL $SETBLOCK - change size of an allocated block (if possible)
                                ;
                                ;   Assembler usage:
                                ;           MOV     ES,block
                                ;           MOV     BX,newsize
                                ;           MOV     AH,setblock
                                ;           INT     21h
                                ;         if setblock fails for growing, BX will have the maximum
                                ;         size possible
                                ;   Error return:
                                ;           AX = error_invalid_block
                                ;              = error_arena_trashed
                                ;              = error_not_enough_memory
                                ;              = error_invalid_function
                                ;--------------------------------------------------------------------------
```

```
42643
42644                                   _$SETBLOCK:
42645                                       ; 04/08/2018 - Retro DOS v3.0
42646                                       ;EnterCrit   critMem
42647 0000745F E880A4                     call   ECritMEM ; MSDOS 3.3 & MSDOS 6.0
42648
42649 00007462 BF0000                     MOV     DI,ARENA.SIGNATURE
42650 00007465 8CC0                           MOV     AX,ES
42651 00007467 48                             DEC     AX
42652 00007468 E871FE                         CALL    check_signature
42653 0000746B 7303                           JNC     SHORT setblock_grab
42654
42655                                   setblock_bad:
42656 0000746D E905FF                         JMP     alloc_trashed
42657
42658                                   setblock_grab:
42659 00007470 8ED8                           MOV     DS,AX
42660 00007472 E877FE                         CALL    Coalesce
42661 00007475 72F6                           JC      SHORT setblock_bad
42662 00007477 8B0E0300                       MOV     CX,[ARENA.SIZE]
42663 0000747B 51                             PUSH    CX
42664 0000747C 39CB                           CMP     BX,CX
42665 0000747E 76A5                           JBE     SHORT alloc_get_size
42666 00007480 E91EFF                         JMP     alloc_fail
42667
42668                                   ;   BREAK $DEALLOC - free previously allocated piece of memory
42669
42670                                   ; MSDOS 6.0
42671                                   ;--------------------------------------------------------------------------
42672                                   ;** $DEALLOC - Free Heap Memory
42673                                   ;
42674                                   ;   ENTRY   (es) = address of item
42675                                   ;
42676                                   ;   EXIT    'C' clear of OK
42677                                   ;           'C' set if error
42678                                   ;             (AX) = error_invalid_block
42679                                   ;   USES    ???? BUGBUG
42680
42681                                   ; MSDOS 2.11 (& MSDOS 3.3)
42682                                   ;--------------------------------------------------------------------------
42683                                   ;SUBTTL $DEALLOC - free previously allocated piece of memory
42684                                   ;
42685                                   ;   Assembler usage:
42686                                   ;           MOV     ES,block
42687                                   ;           MOV     AH,dealloc
42688                                   ;           INT     21h
42689                                   ;
42690                                   ;   Error return:
42691                                   ;           AX = error_invalid_block
42692                                   ;              = error_arena_trashed
42693                                   ;--------------------------------------------------------------------------
42694
42695                                       ; 01/12/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
42696                                       ; 10/03/2024 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMDOS.COM)
42697                                   _$DEALLOC:
42698                                       ; 14/05/2019 - Retro DOS v4.0
42699                                       ; 04/08/2018 - Retro DOS v3.0
42700                                       ;EnterCrit   critMem
42701 00007483 E85CA4                     call   ECritMEM ; MSDOS 3.3 & MSDOS 6.0
42702
42703                                       ; MSDOS 6.0                     ; M016, M068 - Start
42704 00007486 36F606[8600]04               test   byte [ss:DOS_FLAG],EXECA20OFF
42705                                                                       ; Q: was the previous call an int 21
42706                                                                       ;    exec call
42707 0000748C 740D                         jz     short deallocate       ; N: continue
42708 0000748E 36803E[8500]00               cmp    byte [ss:A20OFF_COUNT], 0 ; Q: is count 0
42709 00007494 7505                         jne    short deallocate       ; N: continue
42710                                       ;mov   byte [ss:A20OFF_COUNT], 1 ; Y: set count to 1
42711                                       ; 25/09/2023
42712 00007496 36FE06[8500]                 inc    byte [ss:A20OFF_COUNT]
42713                                   deallocate:                         ; M016, M068 - End
42714 0000749B BF0000                     MOV     DI,ARENA.SIGNATURE ; = 0
42715 0000749E 8CC0                           MOV     AX,ES
42716 000074A0 48                             DEC     AX
42717 000074A1 E838FE                         CALL    check_signature
42718 000074A4 720A                           JC      SHORT dealloc_err
42719 000074A6 26893E0100                     MOV     [ES:ARENA.OWNER],DI
42720                                       ;LeaveCrit critMem
42721 000074AB E861A4                     call    LCritMEM ; MSDOS 3.3 & MSDOS 6.0
42722                                     ; 01/12/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
42723                                         ;transfer SYS_RET_OK
42724                                   dealloc_ok:
42725 000074AE EBAC                       jmp    short alloc_ok
42726                                     ;JMP    SYS_RET_OK
42727
42728                                   dealloc_err:
42729                                       ;LeaveCrit critMem
42730 000074B0 E85CA4                     call    LCritMEM ; MSDOS 3.3 & MSDOS 6.0
42731                                         ;error     error_invalid_block
42732                                     ;mov   al,9
42733 000074B3 B009                       MOV    AL,error_invalid_block
42734                                     ; 01/12/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
42735                                   dealloc_errj:
42736                                   AllocOperErrj:     ; 17/12/2022
42737 000074B5 E9C091                     JMP    SYS_RET_ERR
42738
42739                                   ;   BREAK $AllocOper - get/set allocation mechanism
42740
42741                                   ; MSDOS 6.0
42742                                   ;--------------------------------------------------------------------------
42743                                   ;** $AllocOper - Get/Set Allocation Mechanism
42744                                   ;
42745                                   ;   Assembler usage:
42746                                   ;           MOV     AH,AllocOper
42747                                   ;           MOV     BX,method
42748                                   ;           MOV     AL,func
42749                                   ;           INT     21h
42750                                   ;
42751                                   ;   ENTRY
42752                                   ;           (al) = 0
42753                                   ;             Get allocation Strategy in (ax)
42754                                   ;
42755                                   ;           (al) = 1, (bx) = method = zw0000xy
42756                                   ;             Set allocation strategy.
42757                                   ;              w  = 1  => HIGH_ONLY
42758                                   ;              z  = 1  => HIGH_FIRST
42759                                   ;              xy = 00 => FIRST_FIT
42760                                   ;                 = 01 => BEST_FIT
42761                                   ;                 = 10 => LAST_FIT
42762                                   ;
42763                                   ;           (al) = 2
42764                                   ;             Get UMB link state in (al)
42765                                   ;
42766                                   ;           (al) = 3
```

```
42767                                  ;            Set UMB link state
42768                                  ;             (bx) = 0 => Unlink UMBs
42769                                  ;             (bx) = 1 => Link UMBs
42770                                  ;
42771                                  ;
42772                                  ;   EXIT    'C' clear if OK
42773                                  ;
42774                                  ;            if (al) = 0
42775                                  ;             (ax) = existing method
42776                                  ;            if (al) = 1
42777                                  ;             Sets allocation strategy
42778                                  ;            if (al) = 2
42779                                  ;             (al) = 0 => UMBs not linked
42780                                  ;             (al) = 1 => UMBs linked in
42781                                  ;            if (al) = 3
42782                                  ;             Links/Unlinks the UMBs into DOS chain
42783                                  ;
42784                                  ;           'C' set if error
42785                                  ;             AX = error_invalid_function
42786                                  ;
42787                                  ;   Rev. M000 - added support for HIGH_FIRST in (al) = 1. 7/9/90
42788                                  ;   Rev. M003 - added functions (al) = 2 and (al) = 3. 7/18/90
42789                                  ;   Rev. M009 - (al) = 3 will return 'invalid function' in ax if
42790                                  ;               umbhead has'nt been initialized by sysinit and 'trashed
42791                                  ;               arena' if an arena sig is damaged.
42792                                  ;--------------------------------------------------------------------------
42793                                  
42794                                  ; MSDOS 2.11 (& MSDOS 3.3)
42795                                  ;--------------------------------------------------------------------------
42796                                  ;SUBTTL $AllocOper - get/set allocation mechanism
42797                                  ;
42798                                  ;   Assembler usage:
42799                                  ;           MOV     AH,AllocOper
42800                                  ;           MOV     BX,method
42801                                  ;           MOV     AL,func
42802                                  ;           INT     21h
42803                                  ;
42804                                  ;   Error return:
42805                                  ;            AX = error_invalid_function
42806                                  ;--------------------------------------------------------------------------
42807                                  
42808                                      ; 01/12/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
42809                                      ; 10/03/2024 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMDOS.COM)
42810                                  _$ALLOCOPER:
42811                                      ; 14/05/2019 - Retro DOS v4.0
42812                                      ; MSDOS 6.0
42813 000074B8 08C0                      or      al,al ; 0
42814 000074BA 7411                      jz      short AllocGetStrat
42815                                    ; 17/12/2022
42816                                    ;cmp    al,1
42817                                    ;jz     short AllocSetStrat
42818                                  
42819                                    ; 01/12/2022
42820                                    ;cmp    al, 2
42821                                    ;jb     short AllocSetStrat
42822                                    ;ja     short AllocSetLink
42823                                    ;;jmp   short AllocGetLink
42824                                  ;AllocGetLink:
42825                                      ; MSDOS 6.0
42826                                    ;mov    al,[ss:UMBFLAG]              ; return link state in al
42827                                    ;and    al,LINKSTATE
42828                                    ;;transfer SYS_RET_OK
42829                                    ;jmp    SYS_RET_OK
42830                                  
42831 000074BC 3C02                      cmp     al,2
42832                                    ; 17/12/2022
42833 000074BE 7216                      jb      short AllocSetStrat ; al = 1
42834 000074C0 7425                      je      short AllocGetLink
42835                                  
42836                                    ;cmp    al,2
42837                                    ;jz     short AllocGetLink
42838 000074C2 3C03                      cmp     al,3
42839 000074C4 7429                      jz      short AllocSetLink
42840                                  
42841                                    ; 15/04/2018
42842                                    ;CMP    AL,1
42843                                        ;JB SHORT AllocOperGet
42844                                        ;JZ SHORT AllocOperSet
42845                                  
42846                                  AllocOperError:
42847                                  
42848                                  ; 10/03/2024
42849                                  %if 0
42850                                      ; 04/08/2018 - Retro DOS v3.0
42851                                      ; MSDOS 3.3 (& MSDOS 6.0)     ; Extended Error Locus
42852                                      ;mov    byte [ss:EXTERR_LOCUS],5
42853                                          MOV     byte [SS:EXTERR_LOCUS],errLOC_Mem
42854                                      ;error  error_invalid_function
42855                                  %else
42856                                      ; 10/03/2024 - Retro DOS v5.0
42857                                      ; (PCDOS 7.1 IBMDOS.COM)
42858                                      ;;;
42859 000074C6 E8279E                    call    set_exerr_locus_mem
42860                                      ;;;
42861                                  %endif
42862                                      ;mov    al,1
42863 000074C9 B001                      MOV     AL,error_invalid_function
42864                                      ; 17/12/2022
42865                                  ;AllocOperErrj:
42866                                      ;JMP    SYS_RET_ERR
42867                                      ; 01/12/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
42868                                      ;jmp    short dealloc_errj
42869                                      ; 17/12/2022
42870 000074CB EBE8                      jmp     short AllocOperErrj
42871                                  
42872                                  ; 10/03/2024 - Retro DOS v5.0
42873                                  ; (PCDOS 7.1 IBMDOS.COM)
42874                                  %if 0
42875                                  AllocArenaError:
42876                                      ; MSDOS 6.0
42877                                      MOV     byte [SS:EXTERR_LOCUS],errLOC_Mem
42878                                                                       ; M009: Extended Error Locus
42879                                      ;error  error_arena_trashed   ; M009:
42880                                      ;mov    al,7
42881                                      MOV     AL,error_arena_trashed
42882                                      ;JMP    SYS_RET_ERR
42883                                      jmp     short AllocOperErrj ; 17/12/2022
42884                                  %endif
42885                                  
42886                                  AllocGetStrat:
42887                                      ; MSDOS 6.0
42888                                  AllocOperGet:
42889 000074CD 36A0[0203]                    MOV     AL,[SS:AllocMethod]
42890 000074D1 30E4                          XOR     AH,AH
```

```
                                    ; 01/12/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
                                    ;transfer SYS_RET_OK
                                AllocOperOk:
                                    ; 17/12/2022
                                    ;jmp    short dealloc_ok
000074D3 E99891                     JMP     SYS_RET_OK

                                AllocSetStrat:
                                    ; 14/05/2019
                                    ; MSDOS 6.0
000074D6 53                         push    bx                      ; M000 - start
                                    ; 01/12/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
                                    ;and    bx,STRAT_MASK ; 0FF3Fh; M064: mask off bit 6 & 7
                                    ; 17/12/2022
000074D7 80E33F                     and     bl,3Fh
000074DA 83FB02                     cmp     bx,2                    ; BX must be 0-2
                                    ;cmp    bl,2
000074DD 5B                         pop     bx                      ; M000 - end
000074DE 77E6                       ja      short AllocOperError

                                AllocOperSet:
000074E0 36881E[0203]                   MOV     [SS:AllocMethod],BL
                                    ; 01/12/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
                                    ;transfer SYS_RET_OK
                                AllocOperOkj:
000074E5 EBEC                       jmp     short AllocOperOk
                                    ;JMP    SYS_RET_OK

                                AllocGetLink:
                                    ; MSDOS 6.0
000074E7 36A0[8900]                 mov     al,[ss:UMBFLAG]             ; return link state in al
                                    ;and    al,1
000074EB 2401                       and     al,LINKSTATE
                                    ; 01/12/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
                                    ;transfer SYS_RET_OK
                                AllocOperOkj2:
                                    ; 17/12/2022
000074ED EBE4                       jmp     short AllocOperOk
                                    ;jmp    short AllocOperOkj
                                    ;;JMP   SYS_RET_OK

                                AllocSetLink:
                                    ; MSDOS 6.0                     ; M009 - start
000074EF 368B0E[8C00]               mov     cx,[ss:UMB_HEAD]        ; cx = umb_head

                                ; 10/03/2024
                                %if 0
                                    cmp     cx,0FFFFh               ; Q: has umb_head been initialized
                                    je      short AllocOperError    ; N: error
                                                                    ; Y: continue
                                                                    ; M009 - end
                                    cmp     bx,1
                                    jb      short UnlinkUmbs ; 0
                                    jz      short LinkUmbs   ; 1
                                %else
                                    ; 10/03/2024 - Retro DOS v5.0
                                    ; (PCDOS 7.1 IBMDOS.COM)
                                    ;;;
000074F4 41                         inc     cx                      ; Q: has umb_head been initialized
000074F5 74CF                       jz      short AllocOperError    ; -1 ; N: error
000074F7 49                         dec     cx
000074F8 4B                         dec     bx
000074F9 7418                       jz      short LinkUmbs ; 1
000074FB 43                         inc     bx
                                    ;jz     short UnlinkUmbs ; 0
                                    ;;;
                                %endif
                                    ;jmp    short AllocOperError
                                    ; 10/03/2024
000074FC 75C8                       jnz     short AllocOperError ; > 1

                                UnlinkUmbs:

                                ; 10/03/2024
                                %if 0
                                    ;test   byte [ss:UMBFLAG],1
                                    test    byte [ss:UMBFLAG],LINKSTATE ; Q: umbs unlinked?
                                    jz      short unlinked          ; Y: return

                                    call    GetLastArena            ; get arena before umb_head in DS
                                    jc      short AllocArenaError ; M009: arena trashed
                                %else
                                    ; 10/03/2024 - Retro DOS v5.0
                                    ; (PCDOS 7.1 IBMDOS.COM)
                                    ;;;
000074FE E89BFD                     call    test_umb_flag  ; test byte [ss:UMBFLAG],LINKSTATE
                                                           ; Q: umbs unlinked?
00007501 740E                       jz      short unlinked ; Y: return
00007503 E82200                     call    GetLastArena   ; get arena before umb_head in DS
                                    ;;;
                                %endif
                                                                    ; make it last
00007506 C60600005A                 mov     byte [0],arena_signature_end

                                    ;and    byte [ss:UMBFLAG],0FEh
0000750B 368026[8900]FE             and     byte [ss:UMBFLAG],~LINKSTATE ; indicate unlink'd state in umbflag

                                unlinked:
                                    ; 01/12/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
                                    ;transfer SYS_RET_OK
                                    ; 17/12/2022
00007511 EBC0                       jmp     short AllocOperOk
                                    ;jmp    short AllocOperOkj2
                                    ;;JMP   SYS_RET_OK

                                LinkUmbs:

                                ; 10/03/2024
                                %if 0
                                    test    byte [ss:UMBFLAG],LINKSTATE ; Q: umbs linked?
                                    jnz     short linked            ; Y: return

                                    call    GetLastArena            ; get arena before umb_head
                                    jc      short AllocArenaError ; M009: arena trashed

                                                                    ; make it normal. M061: ds points to
                                %else                                       ; arena before umb_head
                                    ; 10/03/2024 - Retro DOS v5.0
                                    ; (PCDOS 7.1 IBMDOS.COM)
                                    ;;;
00007513 E886FD                     call    test_umb_flag  ; Q: umbs linked?
00007516 750E                       jnz     short linked   ; Y: return
00007518 E80D00                     call    GetLastArena   ; get arena before umb_head
                                    ;;;
```

```
                                %endif
0000751B C60600004D                 mov     byte [0],arena_signature_normal

00007520 36800E[8900]01             or      byte [ss:UMBFLAG],LINKSTATE ; indicate link'd state in umbflag
                                linked:
                                    ; 01/12/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
                                    ;transfer SYS_RET_OK
                                    ; 17/12/2022
00007526 EBAB                       jmp     short AllocOperOk
                                    ;jmp    short unlinked
                                    ;;JMP   SYS_RET_OK

                                ; MSDOS 6.0
                                ;--------------------------------------------------------------------------
                                ; Procedure Name : GetLastArena          -  M003
                                ;
                                ; Inputs     : cx = umb_head
                                ;
                                ;
                                ; Outputs    : If UMBs are linked
                                ;                   ES = umb_head
                                ;                   DS = arena before umb_head
                                ;              else
                                ;                   DS = last arena
                                ;                   ES = next arena. will be umb_head if NC.
                                ;
                                ;              CY if error
                                ;
                                ; Uses              : DS, ES, DI, BX
                                ;--------------------------------------------------------------------------

                                ; 14/05/2019 - Retro DOS v4.0
                                ; DOSCODE:A4D6h (MSDOS 6.21, MSDOS.SYS)

                                ; 01/12/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
                                ; DOSCODE:A476h (MSDOS 5.0, MSDOS.SYS)

                                    ; 10/03/2024 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMDOS.COM)
                                    ; PCDOS 7.1 IBMDOS.COM - DOSCODE:0B6D9h

                                    ; (MSDOS 6.22 MSDOS.SYS - DOSCODE:0A4D6h)
                                    ; (Windows ME IO.SYS - BIOSCODE:9F25h)

                                GetLastArena:
00007528 50                         push    ax                  ; save ax

00007529 36A1[2400]                 mov     ax,[ss:arena_head]
0000752D 8EC0                       mov     es,ax               ; es = arena_head
0000752F 31FF                       xor     di,di

00007531 26803D5A                   cmp     byte [es:di],arena_signature_end
                                                                ; Q: is this the last arena
00007535 7416                       je      short GLA_done      ; Y: return last arena in ES

                                GLA_next:
00007537 8ED8                       mov     ds,ax
00007539 E899FD                     call    arena_next          ; ax, es -> next arena
                                    ; 01/12/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
                                    ;jc     short GLA_err
                                    ; 17/12/2022
0000753C 7222                       jc      short GLA_err2

                                ; 10/03/2024
                                %if 0
                                    test    byte [ss:UMBFLAG],LINKSTATE ; Q: are UMBs linked
                                    jnz     short GLA_chkumb    ; Y: terminating condition is
                                                                ;    umb_head
                                                                ; N: terminating condition is 05Ah
                                %else
                                    ; 10/03/2024 (PCDOS 7.1 IBMDOS.COM)
                                    ;;;
0000753E E85BFD                     call    test_umb_flag
00007541 7506                       jnz     short GLA_chkumb
                                    ;;;
                                %endif
00007543 26803D5A                   cmp     byte [es:di],arena_signature_end
                                                                ; Q: is this the last arena
00007547 EB02                       jmp     short GLA_@f
                                GLA_chkumb:
00007549 39C8                       cmp     ax,cx               ; Q: is this umb_head
                                GLA_@f:
0000754B 75EA                       jne     short GLA_next      ; N: get next arena

                                GLA_done:

                                ; 10/03/2024
                                %if 0
                                                                ; M061 - Start
                                    test    byte [ss:UMBFLAG],LINKSTATE ; Q: are UMBs linked
                                    jnz     short GLA_ret       ; Y: we're done
                                                                ; N: let us confirm that the next
                                                                ;    arena is umb_head
                                %else
                                    ; 10/03/2024 (PCDOS 7.1 IBMDOS.COM)
                                    ;;;
0000754D E84CFD                     call    test_umb_flag
00007550 750B                       jnz     short GLA_ret ; cf=0
                                    ;;;
                                %endif
00007552 8ED8                       mov     ds,ax
00007554 E87EFD                     call    arena_next          ; ax, es -> next arena
                                    ;jc     short GLA_err
00007557 7207                       jc      short GLA_err2
00007559 39C8                       cmp     ax,cx               ; Q: is this umb_head
0000755B 7502                       jne     short GLA_err       ; N: error
                                                                ; M061 - End
                                GLA_ret:
                                    ; 17/12/2022
                                    ;clc
                                    ; 01/12/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
                                    ;clc
0000755D 58                         pop     ax                  ; M061
0000755E C3                         retn                        ; M061

                                GLA_err:
0000755F F9                         stc                         ; M061
                                GLA_err2:
00007560 58                         pop     ax

                                    ; 10/03/2024 - Retro DOS v5.0
                                    ; (PCDOS 7.1 IBMDOS.COM)
                                    ;;;
00007561 58                         pop     ax      ; return address to _$ALLOCOPER (the caller)
00007562 E88B9D                     call    set_exerr_locus_mem
```

```
00007565 E910FE                     jmp     alloc_trashed2 ; (error return from _$ALLOCOPER)
                                    ;;;

                                    ; 10/03/2024
                                    ;retn

                                ;============================================================================
                                ; SRVCALL.ASM, MSDOS 6.0, 1991
                                ;============================================================================
                                ; 04/08/2018 - Retro DOS v3.0

                                ;   TITLE SRVCALL - Server DOS call
                                ;   NAME  SRVCALL

                                ;** SRVCALL.ASM - Server DOS call functions
                                ;
                                ;
                                ;   $ServerCall
                                ;
                                ;   Modification history:
                                ;
                                ;       Created: ARR 08 August 1983

                                ;AsmVars <Installed>

                                ;include dpl.asm

                                ;Installed = TRUE

                                ; 29/04/2019 - Retro DOS v4.0 (MSDOS 6.0, MSDOS 6.21)
                                ; ---------------------------------------------------------------------------
                                ; 01/12/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)

                                ;BREAK <ServerCall -- Server DOS call>

                                ; DOSCODE:A517h (MSDOS 6.21, MSDOS.SYS)
                                ; DOSCODE:A4B7h (MSDOS 5.0, MSDOS.SYS)

                                ; 10/03/2024
                                ; DOSCODE:B71Ah (PCDOS 7.1, IBMDOS.COM) ; Retro DOS v5.0
                                ; DOSCODE:A517h (MSDOS 6.22, MSDOS.SYS) ; Retro DOS v4.2
                                ; BIOSCODE:9F66h (Windows ME, IO.SYS)

                                ;hkn; TABLE SEGMENT
                                ;Public SRVC001S,SRVC001E
                                ;SRVC001S label byte

                                SRVC001S:

00007568 [6C75]                 SERVERTAB:  dw     SERVER_DISP
0000756A [BB75]                 SERVERLEAVE:       dw     SERVERRETURN
0000756C 0B                     SERVER_DISP:       db     (SERVER_DISP_END-SERVER_DISP-1)/2 ; = 11
0000756D [2176]                             dw     SRV_CALL       ; 0
0000756F [BC75]                             dw     COMMIT_ALL     ; 1
00007571 [F275]                             dw     CLOSE_NAME     ; 2
00007573 [FB75]                             dw     CLOSE_UID      ; 3
00007575 [0276]                             dw     CLOSE_UID_PID  ; 4
00007577 [0976]                             dw     GET_LIST       ; 5
00007579 [6276]                             dw     GET_DOS_DATA   ; 6
0000757B [8676]                             dw     SPOOL_OPER     ; 7
0000757D [8676]                             dw     SPOOL_OPER     ; 8
0000757F [8676]                             dw     SPOOL_OPER     ; 9
00007581 [9276]                             dw     _$SetExtendedError  ; 10

                                SERVER_DISP_END:  ;  LABEL BYTE

                                ;SRVC001E label byte

                                SRVC001E:

                                ;hkn; TABLE ENDS

                                ;--------------------------------------------------------------------------
                                ;
                                ; Procedure Name : $ServerCall
                                ;
                                ; Inputs:
                                ;   DS:DX -> DPL  (except calls 7,8,9)
                                ; Function:
                                ;   AL=0    Server DOS call
                                ;   AL=1    Commit All files
                                ;   AL=2    Close file by name (SHARING LOADED ONLY) DS:DX in DPL -> name
                                ;   AL=3    Close all files for DPL_UID
                                ;   AL=4    Close all files for DPL_UID/PID_PID
                                ;   AL=5    Get open file list entry
                                ;               IN: BX File Index
                                ;                   CX User Index
                                ;               OUT:ES:DI -> Name
                                ;                   BX = UID
                                ;               CX = # locked blocks held by this UID
                                ;   AL=6    Get DOS data area
                                ;               OUT: DS:SI -> Start
                                ;                   CX size in bytes of swap if indos
                                ;                   DX size in bytes of swap always
                                ;   AL=7    Get truncate flag
                                ;   AL=8    Set truncate flag
                                ;   AL=9    Close all spool files
                                ;   AL=10   SetExtendedError
                                ;
                                ;--------------------------------------------------------------------------

                                    ; 10/03/2024 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMDOS.COM)
                                    ; PCDOS 7.1 IBMDOS.COM - DOSCODE:0B735h

                                _$ServerCall:
                                    ; 13/11/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
                                    ; DOSCODE:A4D2h (MSDOS 5.0 MSDOS.SYS)
                                    ; 10/06/2019
                                    ; 29/04/2019 - Retro DOS v4.0
                                    ; DOSCODE:A532h (MSDOS 6.21 MSDOS.SYS)

                                    ; 05/08/2018 - Retro DOS v3.0
                                    ; IBMDOS.COM (MSDOS 3.3, 1987) - Offset 657Bh
00007583 3C07                       CMP     AL,7
00007585 7204                       JB      short SET_STUFF
00007587 3C09                       CMP     AL,9
00007589 761A                       JBE     short NO_SET_ID              ; No DPL on calls 7,8,9
                                SET_STUFF:
0000758B 89D6                       MOV     SI,DX                   ; Point to DPL with DS:SI
                                    ;mov    bx,[si+12h]
0000758D 8B5C12                     MOV     BX,[SI+DPL.UID]

                                    ; MSDOS 6.0
                                ;SR;
```

```
                                ; WIN386 updates the USER_ID itself. If WIN386 is present we skip the updating
                                ; of USER_ID

00007590 36F606[5B0F]01             test    byte [SS:IsWin386],1
00007596 7505                       jnz     short skip_win386

                                ;hkn; SS override for user_id and proc_id
                                    ; 15/08/2018
00007598 36891E[3E03]               MOV     [SS:USER_ID],BX             ; Set UID

                                skip_win386:
0000759D 8B5C14                     MOV     BX,[SI+DPL.PID]
000075A0 36891E[3C03]               MOV     [SS:PROC_ID],BX             ; Set process ID
                                NO_SET_ID:
                                    ; 10/06/2019 - Retro DOS v4.0
000075A5 2EFF36[6A75]               PUSH    word [cs:SERVERLEAVE] ; push return address
000075AA 2EFF36[6875]               PUSH    word [cs:SERVERTAB]   ; push table address
000075AF 50                         PUSH    AX
000075B0 E84BA2                     call    TableDispatch

                                ;hkn; SS override
                                    ;;mov   byte [SS:EXETERR_LOCUS],1
                                    ;MOV    byte [SS:EXTERR_LOCUS],errLOC_Unk ; Extended Error Locus
                                    ; 01/07/2024
000075B3 E8279D                     call    set_exerr_locus_unk

                                    ;error  error_invalid_function
                                    ;mov    al,1
000075B6 B001                       MOV     AL,error_invalid_function
                                servercall_error:
000075B8 E9BD90                     JMP     SYS_RET_ERR

                                SERVERRETURN:
000075BB C3                         retn

                                ; Commit - iterate through the open file list and make sure that the
                                ; directory entries are correctly updated.
                                    ; 01/12/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
                                COMMIT_ALL:
000075BC 31DB                       XOR     BX,BX                   ; for (i=0; ThisSFT=getSFT(i); i++)
000075BE 16                         push    ss
000075BF 1F                         pop     ds
000075C0 E81FA3                     call    ECritSFT                ; Gonna scan SFT cache, lock it down
                                CommitLoop:
000075C3 53                         push    bx
000075C4 E82201                     call    SFFromSFN
000075C7 7222                       JC      short CommitDone
000075C9 26833D00                   cmp     word [es:di],0
                                    ;CMP    word [ES:DI+SF_ENTRY.sf_Ref_Count],0
                                                                        ; if (ThisSFT->refcount != 0)
000075CD 7418                       JZ      short CommitNext
                                    ;cmp    word [es:di],0FFFFh ; -1
000075CF 26833DFF                   cmp     word [ES:DI],sf_busy
                                    ;CMP    word [ES:DI+SF_ENTRY.sf_Ref_Count],sf_busy
                                                                        ; BUSY SFTs have god knows what
000075D3 7412                       JZ      short CommitNext        ;   in them.
                                    ; 17/12/2022
000075D5 26F6450680                 test    byte [ES:DI+SF_ENTRY.sf_flags+1],(sf_isnet>>8) ; 80h
                                    ;TEST   word [ES:DI+SF_ENTRY.sf_flags],sf_isnet ; 8000h
000075DA 750B                       JNZ     short CommitNext        ; Skip Network SFTs so the SERVER
                                                                        ;       doesn't deadlock
000075DC 893E[9E05]                 MOV     [THISSFT],DI
000075E0 8C06[A005]                 MOV     [THISSFT+2],ES
000075E4 E8F0C2                     call    DOS_COMMIT              ; DOSCommit ();
                                CommitNext:
000075E7 5B                         pop     bx
000075E8 43                         INC     BX
000075E9 EBD8                       JMP     short CommitLoop
                                CommitDone:
000075EB E821A3                     call    LCritSFT
000075EE 5B                         pop     bx
                                    ; 01/12/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
                                Commit_Ok:
000075EF E97C90                     jmp     SYS_RET_OK

                                CLOSE_NAME:

                                ;if installed

                                ;hkn; SS override
                                    ;call   far [ss:MFTcloN]
000075F2 36FF1E[A400]               Call    far [SS:JShare+(5*4)] ; 5 = MFTcloN
                                ;else
                                ;   Call    MFTcloN
                                ;endif

                                CheckReturns:

                                ; 10/03/2024
                                %if 0
                                    JC      short func_err
                                    ; 01/12/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
                                    ;transfer SYS_RET_OK
                                Commit_Okj:
                                    jmp     short Commit_Ok
                                    ;jmp    SYS_RET_OK
                                %else
000075F7 73F6                       jnc     short Commit_Ok
                                %endif

                                func_err:
                                    ;transfer SYS_RET_ERR
                                    ;jmp    SYS_RET_ERR
000075F9 EBBD                       jmp     short servercall_error

                                CLOSE_UID:

                                ;if installed
                                ;hkn; SS override
                                    ;call   far [ss:MFTcluU]
000075FB 36FF1E[9C00]               Call    far [SS:JShare+(3*4)] ; 3 = MTFTclu
                                ;else
                                ;   Call    MFTcluU
                                ;endif
00007600 EBF5                       JMP     short CheckReturns

                                CLOSE_UID_PID:

                                ;if installed
                                ;hkn; SS override
                                    ;call   far [ss:MFTCloseP]
00007602 36FF1E[A000]               Call    far [SS:JShare+(4*4)] ; 4 = MFTCloseP
                                ;else
```

```
43387                                   ;   Call    MFTCloseP
43388                                   ;endif
43389 00007607 EBEE                       JMP     short CheckReturns
43390
43391                                   GET_LIST:
43392
43393                                   ;if installed
43394                                   ;hkn; SS override
43395                                       ;call   far [ss:MFT_get]
43396 00007609 36FF1E[B400]               Call    far [SS:JShare+(9*4)] ; 9 = MFT_get
43397                                   ;else
43398                                   ;   Call    MFT_get
43399                                   ;endif
43400 0000760E 72E9                       JC      short func_err
43401 00007610 E8648E                     call    Get_User_Stack
43402                                     ;mov    [si+2],bx
43403 00007613 895C02                     MOV     [SI+user_env.user_BX],BX
43404                                     ;mov    [si+10],di
43405 00007616 897C0A                     MOV     [SI+user_env.user_DI],DI
43406                                     ;mov    [si+16],es
43407 00007619 8C4410                     MOV     [SI+user_env.user_ES],ES
43408                                   SetCXOK:
43409                                     ;mov    [si+4],cx
43410 0000761C 894C04                     MOV     [SI+user_env.user_CX],CX
43411                                     ; 01/12/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
43412                                     ;transfer SYS_RET_OK
43413                                   Commit_Okj2:
43414                                     ; 17/12/2022
43415 0000761F EBCE                       jmp     short Commit_Ok
43416                                     ;jmp    short Commit_Okj
43417                                     ;;jmp   SYS_RET_OK
43418
43419                                   SRV_CALL:
43420 00007621 58                         POP     AX                      ; get rid of call to $srvcall
43421 00007622 1E                         push    ds
43422 00007623 56                         push    si
43423 00007624 E8508E                     call    Get_User_Stack
43424 00007627 5F                         pop     di
43425 00007628 07                         pop     es
43426
43427                                   ; DS:SI point to stack
43428                                   ; ES:DI point to DPL
43429
43430 00007629 E8B1A1                     call    XCHGP
43431
43432                                   ; DS:SI point to DPL
43433                                   ; ES:DI point to stack
43434                                   ;
43435                                   ; We now copy the registers from DPL to save stack
43436
43437 0000762C 56                          push    si
43438 0000762D B90600                     MOV     CX,6
43439 00007630 F3A5                       REP     MOVSW                   ; Put in AX,BX,CX,DX,SI,DI
43440 00007632 47                         INC     DI
43441 00007633 47                         INC     DI                      ; Skip user_BP
43442 00007634 A5                         MOVSW                           ; DS
43443 00007635 A5                         MOVSW                           ; ES
43444 00007636 5E                         pop     si                      ; DS:SI -> DPL
43445 00007637 8B04                       mov     ax,[SI]
43446                                     ;MOV    AX,[SI+DPL.AX]
43447                                     ;mov    bx,[si+2]
43448 00007639 8B5C02                     MOV     BX,[SI+DPL.BX]
43449                                     ;mov    cx,[si+4]
43450 0000763C 8B4C04                     MOV     CX,[SI+DPL.CX]
43451                                     ;mov    dx,[si+6]
43452 0000763F 8B5406                     MOV     DX,[SI+DPL.DX]
43453                                     ;mov    di,[si+10]
43454 00007642 8B7C0A                     MOV     DI,[SI+DPL.DI]
43455                                     ;mov    es,[si+14]
43456 00007645 8E440E                     MOV     ES,[SI+DPL.ES]
43457                                     ;push   word [si+8]
43458 00007648 FF7408                     PUSH    word [SI+DPL.SI]
43459                                     ;mov    ds,[si+12]
43460 0000764B 8E5C0C                     MOV     DS,[SI+DPL.DS]
43461 0000764E 5E                         POP     SI
43462
43463                                   ;hkn; SS override for next 3
43464 0000764F 368C1E[EC05]               MOV     [SS:SAVEDS],DS
43465 00007654 36891E[EA05]               MOV     [SS:SAVEBX],BX
43466 00007659 36C606[7205]FF             MOV     byte [SS:FSHARING],-1 ; set no redirect flag
43467 0000765F E9138D                     jmp     REDISP
43468
43469                                   GET_DOS_DATA:
43470 00007662 16                         push    ss
43471 00007663 07                         pop     es
43472 00007664 BF[2003]                   MOV     DI,SWAP_START
43473 00007667 B9[FA0A]                   MOV     CX,SWAP_END
43474 0000766A BA[3A03]                   MOV     DX,SWAP_ALWAYS
43475 0000766D 29F9                       SUB     CX,DI
43476 0000766F 29FA                       SUB     DX,DI
43477 00007671 D1E9                       SHR     CX,1                    ; div by 2, remainder in carry
43478 00007673 83D100                     ADC     CX,0                    ; div by 2 + round up
43479 00007676 D1E1                       SHL     CX,1                    ; round up to 2 boundary.
43480 00007678 E8FC8D                     call    Get_User_Stack
43481                                     ;mov    [si+14],es
43482 0000767B 8C440E                     MOV     [SI+user_env.user_DS],ES
43483                                     ;mov    [si+8],di
43484 0000767E 897C08                     MOV     [SI+user_env.user_SI],DI
43485                                     ;mov    [si+6],dx
43486 00007681 895406                     MOV     [SI+user_env.user_DX],DX
43487 00007684 EB96                       JMP     short SetCXOK
43488
43489                                   SPOOL_OPER:
43490                                       ;CallInstall NETSpoolOper,MultNET,37,AX,BX
43491
43492 00007686 50                          push    ax
43493 00007687 B82511                     mov     ax,1125h
43494 0000768A CD2F                       int     2Fh     ; Multiplex - NETWORK REDIRECTOR - REDIRECTED PRINTER MODE
43495                                                     ; STACK: WORD subfunction
43496                                                     ; Return: CF set on error, AX = error code
43497                                                     ; STACK unchanged
43498 0000768C 5B                         pop     bx
43499                                     ; 17/12/2022
43500                                     ;JC     short func_err2
43501 0000768D 7390                       jnc     short Commit_Okj2
43502                                     ; 01/12/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
43503                                     ;;jmp   SYS_RET_OK
43504                                     ;jmp    short Commit_Okj2
43505
43506                                   func_err2:
43507 0000768F E9E68F                     jmp     SYS_RET_ERR
43508
43509                                   ;Break      <$SetExtendedError - set extended error for later retrieval>
43510                                   ;--------------------------------------------------------------------------
```

```
                                ;
                                ; Procedure Name : $SetExtendedError
                                ;
                                ; $SetExtendedError takes extended error information and loads it up for the
                                ; next extended error call. This is used by interrupt-level proccessors to
                                ; mask their actions.
                                ;
                                ;   Inputs: DS:SI points to DPL which contains all registers
                                ;   Outputs: none
                                ;
                                ;-----------------------------------------------------------------------------

                               _$SetExtendedError:

                               ;hkn; SS override for all variables used

00007692 8B04                      mov     ax,[si]
                                   ;MOV    AX,[SI+DPL.AX]
00007694 36A3[2403]                MOV     [SS:EXTERR],AX
                                   ;mov    ax,[si+10]
00007698 8B440A                    MOV     AX,[SI+DPL.DI]
0000769B 36A3[2803]                MOV     [SS:EXTERRPT],AX
                                   ;mov    ax,[si+14]
0000769F 8B440E                    MOV     AX,[SI+DPL.ES]
000076A2 36A3[2A03]                MOV     [SS:EXTERRPT+2],AX
                                   ;mov    ax,[si+2]
000076A6 8B4402                    MOV     AX,[SI+DPL.BX]
000076A9 36A3[2603]                MOV     [SS:EXTERR_ACTION],AX
                                   ;mov    ax,[si+4]
000076AD 8B4404                    MOV     AX,[SI+DPL.CX]
000076B0 368826[2303]              MOV     [SS:EXTERR_LOCUS],AH
000076B5 C3                        retn

                               ;=============================================================================
                               ; UTIL.ASM, MSDOS 6.0, 1991
                               ;=============================================================================
                               ; 05/08/2018 - Retro DOS v3.0
                               ; 05/05/2019 - Retro DOS v4.0
                               ; 11/03/2024 - Retro DOS v5.0

                               ;** Handle related utilities for MSDOS 2.X.
                               ;-----------------------------------------------------------------------------
                               ;   pJFNFromHandle written
                               ;   SFFromHandle   written
                               ;   SFFromSFN      written
                               ;   JFNFree        written
                               ;   SFNFree        written
                               ;
                               ;   Modification history:
                               ;
                               ;       Created: MZ 1 April 1983
                               ;-----------------------------------------------------------------------------

                               ;   BREAK  <pJFNFromHandle - return pointer to JFN table entry>

                               ;** pJFNFromHandle - Translate Handle to Pointer to JFN
                               ;-----------------------------------------------------------------------------
                               ;   pJFNFromHandle takes a file handle and turns that into a pointer to
                               ;   the JFN entry (i.e., to a byte holding the internal file handle #)
                               ;
                               ;   NOTE:
                               ;     This routine is called from $CREATE_PROCESS_DATA_BLOCK which is called
                               ;     at DOSINIT time with SS NOT DOSGROUP
                               ;
                               ;   ENTRY   (bx) = handle
                               ;   EXIT    'C' clear if ok
                               ;             (es:di) = address of JFN value
                               ;           'C' set if error
                               ;             (ax) = error code
                               ;   USES    AX, DI, ES, Flags
                               ;-----------------------------------------------------------------------------

                                   ; 02/12/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
                                   ; 11/03/2024 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMDOS.COM)

                               pJFNFromHandle:
                                   ; 05/05/2019 - Retro DOS v4.0
                                   ;getdseg <es>                  ; es -> dosdata
000076B6 2E8E06[0700]              mov     es,[cs:DosDSeg]

                                   ;MOV    ES,[cs:CurrentPDB]     ; get user process data block
000076BB 268E06[3003]              mov     es,[es:CurrentPDB]

                                   ;cmp    bx,[ES:32h]
000076C0 263B1E3200                CMP     BX,[ES:PDB.JFN_Length]; is handle greater than allocated
000076C5 7204                      JB      short pjfn10           ; no, get offset
                               ReturnCarry_inv_hndl: ; 05/08/2018 - Retro DOS v3.0
                                   ;mov    al,6
000076C7 B006                      MOV     AL,error_invalid_handle ; appropriate error
                               ReturnCarry:
000076C9 F9                        STC                             ; signal error
000076CA C3                        retn                           ; go back
                               pjfn10:
                                   ;les    di,[es:34h]
000076CB 26C43E3400                LES     DI,[ES:PDB.JFN_Pointer]       ; get pointer to beginning of table
000076D0 01DF                      ADD     DI,BX                  ; add in offset, clear 'C'
                                   ;clc
                               pJFNFromHandle_error:
000076D2 C3                        retn

                               ;BREAK <SFFromHandle - return pointer (or error) to SF entry from handle>
                               ;-----------------------------------------------------------------------------
                               ;
                               ; Procedure Name : SFFromHandle
                               ;
                               ; SFFromHandle - Given a handle, get JFN and then index into SF table
                               ;
                               ;   Input:      BX has handle
                               ;   Output:     Carry Set
                               ;                   AX has error code
                               ;               Carry Reset
                               ;                   ES:DI has pointer to SF entry
                               ;   Registers modified: If error, AX,ES, else ES:DI
                               ; NOTE:
                               ;   This routine is called from $CREATE_PROCESS_DATA_BLOCK which is called
                               ;       at DOSINIT time with SS NOT DOSGROUP
                               ;
                               ;-----------------------------------------------------------------------------

                                   ; 02/12/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
                                   ; 11/03/2024 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMDOS.COM)

                               SFFromHandle:
000076D3 E8E0FF                    CALL    pJFNFromHandle         ; get jfn pointer
```

```
                                    ;retc                       ; return if error
000076D6 72FA                       jc      short pJFNFromHandle_error
000076D8 26803DFF                   CMP     BYTE [ES:DI],-1             ; unused handle
                                    ;JNZ    short GetSF         ; nope, suck out SF
                                    ;;mov   al,6
                                    ;MOV    AL,error_invalid_handle ; appropriate error
                                    ;jmp    short ReturnCarry   ; signal it
                                    ; 17/12/2022
                                    ; 02/12/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
000076DC 74E9                       jz      short ReturnCarry_inv_hndl ; Retro DOS v3.0 modification
                                    ;JNZ    short GetSF         ; nope, suck out SF
                                    ;;mov   al,6
                                    ;MOV    AL,error_invalid_handle ; appropriate error
                                    ;jmp    short ReturnCarry   ; signal it
                                GetSF:
000076DE 53                         push    bx                  ; save handle
000076DF 268A1D                     MOV     BL,[ES:DI]          ; get SFN
000076E2 30FF                       XOR     BH,BH                 ; ignore upper half
000076E4 E80200                     CALL    SFFromSFN             ; get real sf spot
000076E7 5B                         pop     bx                  ; restore
000076E8 C3                         retn                        ; say goodbye

                                ;BREAK <SFFromSFN - index into SF table for SFN>

                                ;** SFFromSFN - Get an SF Table entry from an SFN
                                ;-----------------------------------------------------------------------------
                                ;   SFFromSfn uses an SFN to index an entry into the SF table. This
                                ;   is more than just a simple index instruction because the SF table
                                ;   can be made up of multiple pieces chained together. We follow the
                                ;   chain to the right piece and then do the index operation.
                                ;
                                ;   NOTE:
                                ;   This routine is called from SFFromHandle which is called
                                ;       at DOSINIT time with SS NOT DOSGROUP
                                ;
                                ;   ENTRY  BX has SF index
                                ;   EXIT   'C' clear if OK
                                ;            ES:DI points to SF entry
                                ;          'C' set if index too large
                                ;   USES   BX, DI, ES
                                ;-----------------------------------------------------------------------------

                                    ; 02/12/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
                                    ; 11/03/2024 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMDOS.COM)

                                SFFromSFN:
                                    ; 05/05/2019 - Retro DOS v4.0
                                    ;getdseg <es>               ; es -> dosdata
000076E9 2E8E06[0700]               mov     es,[cs:DosDSeg]

                                    ;LES    DI,[CS:SFT_ADDR]    ; (es:di) = start of SFT table
000076EE 26C43E[2A00]               les     di,[es:SFT_ADDR]
                                sfsfn5:
                                    ;cmp    bx,[es:di+4]
000076F3 263B5D04                   CMP     BX,[ES:DI+SFT.SFCount]; is handle in this table?
000076F7 720E                       JB      short sfsfn7        ; yes, go grab it
                                    ;sub    bx,[es:di+4]
000076F9 262B5D04                   SUB     BX,[ES:DI+SFT.SFCount]
000076FD 26C43D                     les     di,[es:di] ; 14/08/2018
                                    ;LES    DI,[ES:DI+SFT.SFLink] ; get next table segment
00007700 83FFFF                     CMP     DI,-1                 ; end of tables?
00007703 75EE                       JNZ     short sfsfn5        ; no, try again
00007705 F9                         STC
00007706 C3                         retn                        ; return with error, not found
                                sfsfn7:
00007707 50                         push    ax
                                    ;;mov   ax,53 ; MSDOS 3.3
                                    ;mov    ax,59 ; MSDOS 5.0 to PCDOS 7.1 (to Win ME) ; 11/03/2024
                                    ;MOV    AX,SF_ENTRY.size    ; put it in a nice place

                                    ; 17/12/2022
00007708 B03B                       mov     al,SF_ENTRY.size ; 28/05/2019
                                    ; 07/12/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
                                    ;mov    ax,SF_ENTRY.size ; 59

0000770A F6E3                       MUL     BL                  ; (ax) = offset into this SF block
0000770C 01C7                       ADD     DI,AX               ; add base of SF block
0000770E 58                         pop     ax
                                    ;add    di,6
0000770F 83C706                     ADD     DI,SFT.SFTable      ; offset into structure, 'C' cleared
00007712 C3                         retn                        ; return with 'C' clear

                                ;   BREAK <JFNFree - return a jfn pointer if one is free>

                                ;** JFNFree - Find a Free JFN Slot
                                ;-----------------------------------------------------------------------------
                                ;   JFNFree scansthrough the JFN table and returnsa pointer to a free slot
                                ;
                                ;   ENTRY  (ss) = DOSDATA
                                ;   EXIT   'C' clear if OK
                                ;            (bx) = new handle
                                ;            (es:di) = pointer to JFN slot
                                ;          'C' set if error
                                ;            (al) = error code
                                ;   USES   bx, di, es, flags
                                ;-----------------------------------------------------------------------------

                                JFNFree:
00007713 31DB                       XOR     BX,BX               ; (bx) = initial JFN to try
                                jfnf1:
00007715 E89EFF                     CALL    pJFNFromHandle      ; get the appropriate handle
00007718 7209                       JC      short jfnf5         ; no more handles
0000771A 26803DFF                   CMP     BYTE [ES:DI],-1             ; free?
0000771E 7405                       je      short jfnfx         ; yes, carry is clear
00007720 43                         INC     BX                    ; no, next handle
00007721 EBF2                       JMP     short jfnf1         ; and try again

                                    ; Error. 'C' set
                                jfnf5:
                                    ;mov    al,4
00007723 B004                       MOV     AL,error_too_many_open_files
                                jfnfx:
00007725 C3                         retn                        ; bye

                                ;   BREAK <SFNFree - Allocate a free SFN>

                                ;** SFNFree - Allocate a Free SFN/SFT
                                ;-----------------------------------------------------------------------------
                                ;   SFNFree scans through the sf table looking for a free entry
                                ;   If it finds one it partially allocates it by setting SFT_REF_COUNT = -1
                                ;
                                ;   The problem is that we want to mark the SFT busy so that other threads
                                ;   can't allocate the SFT before we're finished marking it up.  However,
                                ;   we can't just mark it busy because we may get blown out of our open
```

```
43759                                   ;   by INT24 and leave the thing orphaned.      To solve this we mark it
43760                                   ;   "allocation in progress" by setting SFT_REF_COUNT = -1.  If we see
43761                                   ;   an SFT with this value we look to see if it belongs to this user
43762                                   ;   and process.  If it does belong to us then it must be an orphan
43763                                   ;   and we reclaim it.
43764                                   ;
43765                                   ;   BUGBUG - improve the performance. I guess it's smaller to call SFFromSFN
43766                                   ;           over and over, but we could at least set a high water mark...
43767                                   ;           cause an N^2 loop calling slow SFFromSFN is real slow, too slow
43768                                   ;           even though this is not a frequently called routine - jgl
43769                                   ;
43770                                   ;   ENTRY   (ss) = DOSDATA
43771                                   ;   EXIT    'C' clear if no error
43772                                   ;             (bx) = SFN
43773                                   ;             (es:di) = pointer to SFT
43774                                   ;             es:[di].SFT_REF_COUNT = -1
43775                                   ;           'C' set if error
43776                                   ;             (al) = error code
43777                                   ;   USES    bx, di, es, Flags
43778                                   ;--------------------------------------------------------------------------
43779
43780                                       ; 02/12/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
43781                                       ; DOSCODE:A682h (MSDOS 5.0 MSDOS.SYS)
43782
43783                                       ; 11/03/2024 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMDOS.COM)
43784                                       ; PCDOS 7.1 IBMDOS.COM - DOSCODE:0B8DEh
43785
43786                                   SFNFree:
43787                                       ; 12/08/2018
43788                                       ; 05/08/2018 - Retro DOS v3.0
43789                                       ;
43790                                       ; MSDOS 6.0
43791 00007726 50                          push    ax
43792 00007727 31DB                        xor     bx,bx                   ; (bx) = SFN to consider
43793                                   sfnf5:
43794 00007729 53                          push    bx
43795 0000772A E8BCFF                      call    SFFromSFN               ; get the potential handle
43796 0000772D 5B                          pop     bx
43797 0000772E 723A                        jc      short sfnf95            ; no more free SFNs
43798 00007730 26833D00                    cmp     word [ES:DI],0
43799                                      ;cmp    word [ES:DI+SF_ENTRY.sf_Ref_Count],0 ; free?
43800 00007734 741D                        je      short sfnf20                    ; yep, got one
43801
43802                                      ;cmp    word [es:di],0FFFFh ; -1
43803 00007736 26833DFF                    cmp     word [ES:DI],sf_busy
43804                                      ;cmp    word [ES:DI+SF_ENTRY.sf_ref_count],sf_busy
43805 0000773A 7403                        je      short sfnf10            ; special busy mark
43806                                   sfnf7:
43807 0000773C 43                          inc     bx                      ; try the next one
43808 0000773D EBEA                        jmp     short sfnf5
43809
43810                                   ;   The SFT has the special "busy" mark; if it belongs to us then
43811                                   ;   it was abandoned during a earlier call and we can use it.
43812                                   ;
43813                                   ;   (bx)    = SFN
43814                                   ;   (es:di) = pointer to SFT
43815                                   ;   (TOS)   = caller's (ax)
43816
43817                                   sfnf10:
43818 0000773F 36A1[3E03]                  mov     ax,[SS:USER_ID]
43819                                      ;cmp    [es:di+2Fh],ax
43820 00007743 2639452F                    cmp     [ES:DI+SF_ENTRY.sf_UID],ax
43821 00007747 75F3                        jnz     short sfnf7             ; not ours
43822 00007749 36A1[3C03]                  mov     ax,[SS:PROC_ID]
43823                                      ;cmp    [es:di+31h],ax
43824 0000774D 26394531                    cmp     [ES:DI+SF_ENTRY.sf_PID],ax
43825 00007751 75E9                        jnz     short sfnf7             ; can't use this one, try the next
43826
43827                                   ;   We have an SFT to allocate
43828                                   ;
43829                                   ;   (bx)    = SFN
43830                                   ;   (es:di) = pointer to SFT
43831                                   ;   (TOS)   = caller's (ax)
43832
43833                                   sfnf20:
43834                                       ; cf = 0 ;; Retro DOS v3.0
43835
43836                                       ;mov    word [es:di],0FFFFh
43837 00007753 26C705FFFF                  mov     word [ES:DI],sf_busy
43838                                                                       ; make sure that this is allocated
43839                                       ;mov    word [ES:DI+SF_ENTRY.sf_ref_count],sf_busy
43840
43841 00007758 36A1[3E03]                  mov     ax,[SS:USER_ID]
43842                                      ;mov    [es:di+2Fh],ax
43843 0000775C 2689452F                    mov     [ES:DI+SF_ENTRY.sf_UID],ax
43844 00007760 36A1[3C03]                  mov     ax,[SS:PROC_ID]
43845                                      ;mov    [es:di+31h],ax
43846 00007764 26894531                    mov     [ES:DI+SF_ENTRY.sf_PID],ax
43847                                   sfnf21: ;; Retro DOS v3.0
43848                                       ; 02/12/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
43849                                       ;pop    ax
43850                                       ;;clc
43851                                       ;retn                           ; return with no error
43852                                       ; 17/12/2022
43853 00007768 58                          pop     ax
43854                                      ;clc
43855 00007769 C3                          retn
43856
43857                                   ;** Error - no more free SFNs
43858                                   ;
43859                                   ;   'C' set
43860                                   ;   (TOS) = saved ax
43861
43862                                   sfnf95:
43863 0000776A 58                          pop     ax
43864
43865                                   ; 11/03/2024
43866                                   %if 0
43867                                       ;mov    al,4
43868                                       mov     al,error_too_many_open_files
43869                                       retn                            ; return with 'C' and error
43870                                   %else
43871                                       ; 11/03/2024 (PCDOS 7.1 IBMDOS.COM)
43872                                       ;;;
43873 0000776B EBB6                        jmp     short jfnf5
43874                                      ;;;
43875                                   %endif
43876
43877                                   ;============================================================================
43878                                   ; HANDLE.ASM, MSDOS 6.0, 1991
43879                                   ;============================================================================
43880                                   ; 13/07/2018 - Retro DOS v3.0
43881                                   ; 20/05/2019 - Retro DOS v4.0
43882                                   ; 11/03/2024 - Retro DOS v5.0
```

```
                                ; DOSCODE:A72Bh (MSDOS 6.21, MSDOS.SYS)

                                ;   BREAK <$Close - return a handle to the system>
                                ;--------------------------------------------------------------------------
                                ;
                                ;** $Close - Close a file Handle
                                ;
                                ;   BUGBUG - close gets called a LOT with invalid handles - sizzle that
                                ;           path
                                ;
                                ;   Assembler usage:
                                ;       MOV     BX, handle
                                ;       MOV     AH, Close
                                ;       INT     int_command
                                ;
                                ;   ENTRY   (bx) = handle
                                ;   EXIT    <normal INT21 return convention>
                                ;   USES    all
                                ;
                                ;--------------------------------------------------------------------------

                                ; 02/12/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
                                ; DOSCODE:A6CBh (MSDOS 5.0 MSDOS.SYS)

                                ; 11/03/2024 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMDOS.COM)
                                ; PCDOS 7.1 IBMDOS.COM - DOSCODE:0B926h

                                ; (Windows ME IO.SYS - BIOSCODE:0A145h)
                                ; (MSDOS 6.22 MSDOS.SYS - DOSCODE:0A72Bh)

                                _$CLOSE:
                                ;   Grab the SFT pointer from the JFN.

0000776D E8F602                     call    CheckOwner              ; get system file entry
00007770 722B                       jc      short CloseError        ; error return
00007772 16                         push    ss
00007773 1F                         pop     ds                      ; For DOS_CLOSE
00007774 893E[9E05]                 MOV     [THISSFT],DI            ; save offset of pointer
00007778 8C06[A005]                 MOV     [THISSFT+2],ES          ; save segment value

                                ; DS:SI point to JFN table entry.
                                ; ES:DI point to SFT
                                ;
                                ; We now examine the user's JFN entry; If the file was a 70-mode file (network
                                ; FCB, we examine the ref count on the SFT; if it was 1, we free the JFN.
                                ; If the file was not a net FCB, we free the JFN too.

                                    ;CMP    word [ES:DI+SF_ENTRY.sf_ref_count],1
0000777C 26833D01                   cmp     word [ES:DI],1          ; will the SFT become free?
00007780 740A                       jz      short FreeJFN           ; yes, free JFN anyway.
                                    ;mov    al,[ES:DI+2]
00007782 268A4502                   MOV     AL,[ES:DI+SF_ENTRY.sf_mode]
                                    ;and    al,0F0h
00007786 24F0                       AND     AL,SHARING_MASK
                                    ;cmp    al,70h
00007788 3C70                       CMP     AL,SHARING_NET_FCB
0000778A 7407                       JZ      short PostFree          ; 70-mode and big ref count => free it

                                ; The JFN must be freed. Get the pointer to it and replace the contents with
                                ; -1.

                                FreeJFN:
0000778C E827FF                     call    pJFNFromHandle          ; d = pJFN (handle);
0000778F 26C605FF                   MOV     BYTE [ES:DI],0FFh       ; release the JFN
                                PostFree:

                                ; ThisSFT is correctly set, we have DS = DOSDATA. Looks OK for a DOS_CLOSE!

00007793 E892BF                     call    DOS_CLOSE

                                ; DOS_Close may return an error. If we see such an error, we report it but
                                ; the JFN stays closed because DOS_Close always frees the SFT!

00007796 7205                       JC      short CloseError
                                    ;mov    ah,3Eh
00007798 B43E                       MOV     AH,CLOSE                ; MZ Bogus multiplan fix
                                    ; 02/12/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
                                CloseOk:
0000779A E9D18E                     jmp     SYS_RET_OK
                                CloseError:
                                CommitError:        ; 11/03/2024
0000779D E9D88E                     jmp     SYS_RET_ERR

                                ;   BREAK <$Commit - commit the file>
                                ;--------------------------------------------------------------------------
                                ;
                                ;** $Commit - Commit a File
                                ;
                                ;   $Commit "commits" a file to disk - all of it's buffers are
                                ;   flushed out. BUGBUG - I'm pretty sure that $Commit doesn't update
                                ;   the directory entry, etc., so this commit is pretty useless. check
                                ;   and fix this!! jgl
                                ;
                                ;   Assembler usage:
                                ;       MOV     BX, handle
                                ;       MOV     AH, Commit
                                ;       INT     int_command
                                ;
                                ;   ENTRY   (bx) = handle
                                ;   EXIT    none
                                ;   USES    all
                                ;;--------------------------------------------------------------------------

                                _$COMMIT:
                                ;   Grab the SFT pointer from the JFN.

000077A0 E8C302                     call    CheckOwner              ; get system file entry
                                    ;JC     short CommitError       ; error return
                                    ; 11/03/2024
000077A3 72F8                       jc      short CommitError

000077A5 16                         push    ss
000077A6 1F                         pop     ds                      ; For DOS_COMMIT
000077A7 893E[9E05]                 MOV     [THISSFT],DI            ; save offset of pointer
000077AB 8C06[A005]                 MOV     [THISSFT+2],ES          ; save segment value

                                ;   ThisSFT is correctly set, we have DS = DOSDATA. Looks OK for a DOS_COMMIT
                                ;
                                ;   ES:DI point to SFT

000077AF E825C1                     call    DOS_COMMIT
000077B2 72E9                       JC      short CommitError
                                    ; 07/12/2022
```

```
                                    ;jc     short CloseError
                                    ;mov    ah,68h
000077B4 B468                       MOV     AH,COMMIT
                                    ; 02/12/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
                                    ;jmp    SYS_RET_OK
                                CommitOk:
000077B6 EBE2                       jmp     short CloseOk

                                ; 11/03/2024
                                ;CommitError:
                                ;   ; 07/12/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
                                ;   ;jmp    SYS_RET_ERR
                                ;   jmp     short CloseError

                                ;   BREAK <$ExtHandle - extend handle count>

                                ;** $ExtHandle - Extend Handle Count
                                ;-----------------------------------------------------------------------
                                ;   Assembler usage:
                                ;       MOV     BX, Number of Opens Allowed (MAX=65534;66535 is
                                ;       MOV     AX, 6700H                   reserved to mark SFT
                                ;       INT     int_command                 busy )
                                ;
                                ;   ENTRY   (bx) = new number of handles
                                ;   EXIT    'C' clear if OK
                                ;           'C' set iff err
                                ;             (ax) = error code
                                ;                    AX = error_not_enough_memory
                                ;                         error_too_many_open_files
                                ;   USES    all
                                ;-----------------------------------------------------------------------

                                    ; 11/03/2024 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMDOS.COM)

                                _$ExtHandle:
000077B8 31ED                       XOR     BP,BP                   ; 0: enlarge  1: shrink  2:psp
                                    ;cmp    bx,20
000077BA 83FB14                     CMP     BX,FILPERPROC
000077BD 7303                       JAE     short exth2             ; Don't set less than FilPerProc no
000077BF BB1400                     MOV     BX,FILPERPROC
                                exth2:
000077C2 368E06[3003]               MOV     ES,[ss:CurrentPDB]      ; get user process data block;smr;SS Override
                                    ;mov    cx,[ES:32h]
000077C7 268B0E3200                 MOV     CX,[ES:PDB.JFN_Length]; get number of handle allowed
000077CC 39CB                       CMP     BX,CX                   ; the requested == current
                                    ;JE     short ok_done           ; yes and exit
                                    ; 11/03/2024
000077CE 74CA                       je      short CloseOk
000077D0 771E                       JA      short larger            ; go allocate new table

                                ;   We're going to shrink the # of handles available

                                    ;MOV    BP,1                    ; shrink
                                    ; 11/03/2024
000077D2 45                         inc     bp

                                    ;mov    ds,[ES:36h]
000077D3 268E1E3600                 MOV     DS,[ES:PDB.JFN_Pointer+2] ;
000077D8 89DE                       MOV     SI,BX                   ;
000077DA 29D9                       SUB     CX,BX                   ; get difference

                                ;   BUGBUG - code a SCASB here, should be a bit smaller
                                chck_handles:
000077DC 803CFF                     CMP     BYTE [SI],-1            ; scan through handles to ensure close
000077DF 7539                       JNZ     short too_many_files    ; status
000077E1 46                         INC     SI
000077E2 E2F8                       LOOP    chck_handles

000077E4 83FB14                     CMP     BX,FILPERPROC           ; = 20
000077E7 7707                       JA      short larger            ; no

                                    ;MOV    BP,2                    ; psp
                                    ; 11/03/2024
000077E9 45                         inc     bp
                                    ;mov    di,24
000077EA BF1800                     MOV     DI,PDB.JFN_TABLE        ; es:di -> jfn table in psp
000077ED 53                         PUSH    BX
000077EE EB1B                       JMP     short movhandl

                                larger:
                                    ;CMP    BX,-1                   ; 65535 is not allowed
                                    ;JZ     short invalid_func      ; 10/08/2018
                                    ; 11/03/2024 (PCDOS 7.1 IBMDOS.COM)
                                    ;;;
000077F0 43                         inc     bx
000077F1 747D                       jz      short invalid_func
000077F3 4B                         dec     bx
                                    ;;;
000077F4 F8                         CLC
000077F5 53                         PUSH    BX                      ; save requested number
000077F6 83C30F                     ADD     BX,0FH                  ; adjust to paragraph boundary
000077F9 B104                       MOV     CL,4
                                    ;ror    bx,cl                   ; MSDOS 3.3
000077FB D3DB                       RCR     BX,CL                   ; DOS 4.00 fix        ;AC000;
                                    ;AND    BX,1FFFH                ; clear most 3 bits
                                    ; 01/07/2024
000077FD 80E71F                     and     bh,1Fh

00007800 55                         PUSH    BP
00007801 E80AFB                     call    _$ALLOC                 ; allocate memory
00007804 5D                         POP     BP
00007805 7264                       JC      short no_memory             ; not enough memory

00007807 8EC0                       MOV     ES,AX                   ; es:di points to new table memory
00007809 31FF                       XOR     DI,DI
                                movhandl:
0000780B 368E1E[3003]               MOV     DS,[ss:CurrentPDB]      ; get user PDB address;smr;SS Override

00007810 F7C50300                   test    BP,3                    ; enlarge ?
00007814 7409                       JZ      short enlarge           ; yes
00007816 59                         POP     CX                      ; cx = the amount you shrink
00007817 51                         PUSH    CX
00007818 EB09                       JMP     short copy_hand

                                ;   Done.  'C' clear

                                ; 17/12/2022
                                ;ok_done:
                                ;   ; 02/12/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
                                ;   ;jmp    short CommitOk
                                ;   ; 17/12/2022
                                ;   jmp     SYS_RET_OK

                                too_many_files:
```

```
                                    ;mov    al,4
0000781A B004                       MOV     AL,error_too_many_open_files
                                    ; 02/12/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
                                    ;jmp    SYS_RET_ERR
                                CommitErrorj:
                                    ;jmp    short CommitError
                                    ; 17/12/2022
0000781C E9598E                     jmp     SYS_RET_ERR

                                ; 11/03/2024
                                ; 17/12/2022
                                ;ok_done:
                                ;   ; 02/12/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
                                ;   ;jmp    short CommitOk
                                ;   ; 17/12/2022
                                ;   jmp     SYS_RET_OK

                                enlarge:
                                    ;mov    cx,[32h]
0000781F 8B0E3200                   MOV     CX,[PDB.JFN_Length]   ; get number of old handles
                                copy_hand:
00007823 89CA                       MOV     DX,CX
                                    ;lds    si,[34h]
00007825 C5363400                   LDS     SI,[PDB.JFN_Pointer]  ; get old table pointer
00007829 F3A4                       REP     MOVSB                 ; copy information to new table
0000782B 59                         POP     CX                    ; get new number of handles
0000782C 51                         PUSH    CX                    ; save it again
0000782D 29D1                       SUB     CX,DX                 ; get the difference
0000782F B0FF                       MOV     AL,-1                 ; set availability to handles
00007831 F3AA                       REP     STOSB
00007833 368E1E[3003]               MOV     DS,[ss:CurrentPDB]    ; get user process data block;smr;SS Override
                                    ;cmp    word [34h],0
00007838 833E340000                 CMP     WORD [PDB.JFN_Pointer],0 ; check if original table pointer
0000783D 750D                       JNZ     short update_info     ; yes, go update PDB entries
0000783F 55                         PUSH    BP
00007840 1E                         PUSH    DS                    ; save old table segment
00007841 06                         PUSH    ES                    ; save new table segment
00007842 8E063600                   MOV     ES,[PDB.JFN_Pointer+2]; get old table segment
00007846 E83AFC                     call    _$DEALLOC             ; deallocate old table memory
00007849 07                         POP     ES                    ; restore new table segment
0000784A 1F                         POP     DS                    ; restore old table segment
0000784B 5D                         POP     BP

                                update_info:
0000784C F7C50200                   test    BP,2                  ; psp?
00007850 7408                       JZ      short non_psp         ; no
                                    ;mov    word [34h],18h ; 24
00007852 C70634001800               MOV     WORD [PDB.JFN_Pointer],PDB.JFN_TABLE ; restore
00007858 EB06                       JMP     short final
                                non_psp:
                                    ;mov    word [34h],0
0000785A C70634000000               MOV     WORD [PDB.JFN_Pointer],0 ; new table pointer offset always 0
                                final:
                                    ;mov    [36h],es
00007860 8C063600                   MOV     [PDB.JFN_Pointer+2],ES; update table pointer segment
                                    ;pop    word [32h]
00007864 8F063200                   POP     word [PDB.JFN_Length] ; restore new number of handles
                                    ; 11/03/2024
                                ok_done:
                                    ; 02/12/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
00007868 E9038E                     jmp     SYS_RET_OK
                                ;ok_done_j:
                                ;   jmp     short ok_done

                                no_memory:
0000786B 5B                         POP     BX                    ; clean stack
                                    ;mov    al,8
0000786C B008                       MOV     AL,error_not_enough_memory
                                    ; 02/12/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
                                    ;jmp    SYS_RET_ERR
                                CommitErrorj2:
0000786E EBAC                       jmp     short CommitErrorj

                                invalid_func:
                                    ;mov    al,1
00007870 B001                       MOV     AL,error_invalid_function
                                    ; 02/12/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
                                    ;jmp    SYS_RET_ERR
                                CommitErrorj3:
                                    ;jmp    short CommitErrorj2
                                    ; 17/12/2022
00007872 EBA8                       jmp     short CommitErrorj

                                ; 20/05/2019 - Retro DOS v4.0
                                ; DOSCODE:A83Ah (MSDOS 6.21, MSDOS.SYS)

                                ; 02/12/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
                                ; DOSCODE:A7DAh (MSDOS 5.0 MSDOS.SYS)

                                ;   BREAK <$READ - Read from a file handle>
                                ;--------------------------------------------------------------------------
                                ;
                                ;** $Read - Read from a File Handle
                                ;
                                ;   Assembler usage:
                                ;
                                ;   LDS     DX, buf
                                ;   MOV     CX, count
                                ;   MOV     BX, handle
                                ;   MOV     AH, Read
                                ;   INT     int_command
                                ;     AX has number of bytes read
                                ;
                                ;   ENTRY   (bx) = file handle
                                ;           (cx) = byte count
                                ;           (ds:dx) = buffer address
                                ;   EXIT    Through system call return so that to user:
                                ;             'C' clear if OK
                                ;               (ax) = bytes read
                                ;             'C' set if error
                                ;               (ax) = error code
                                ;
                                ;--------------------------------------------------------------------------

                                    ; 12/03/2024 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMDOS.COM)
                                    ; PCDOS 7.1 IBMDOS.COM - DOSCODE:0BA2Eh

                                    ; (MSDOS 6.22 MSDOS.SYS - DOSCODE:0A83Ah)
                                    ; (Windows ME IO.SYS - BIOSCODE:0A256h)

                                _$READ:
00007874 BE[773B]                   MOV     SI,DOS_READ
                                ReadDo:
00007877 E83CFE                     call    pJFNFromHandle
```

```
                                    JC      short ReadError

                                    MOV     AL,[ES:DI]
                                    call    CheckOwner              ; get the handle
                                    JNC     short ReadSetup                 ; no errors do the operation

                                ;   Have an error. 'C' set

                                ReadError:
                                    ; 02/12/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
                                    ;;jmp   SYS_RET_ERR              ; go to error traps
                                    ;jmp    short CommitErrorj3
                                    ; 17/12/2022
                                    jmp     SYS_RET_ERR

                                ReadSetup:
                                    MOV     [ss:THISSFT],DI                 ; save offset of pointer;smr;SS Override
                                    MOV     [ss:THISSFT+2],ES       ; save segment value  ;smr;SS Override
                                    ; 20/05/2019 - Retro DOS v4.0
                                    ; MSDOS 6.0
                                ;; Extended Open
                                    ;test   byte [es:di+3],20h
                                    test    byte [ES:DI+SF_ENTRY.sf_mode+1],(INT_24_ERROR>>8)
                                                                            ;AN000;;EO. need i24
                                    JZ      short needi24                   ;AN000;;EO. yes
                                    OR      byte [ss:EXTOPEN_ON],EXT_OPEN_I24_OFF ; 2
                                                                            ;AN000;;EO. set it off;smr;SS Override
                                needi24:                                    ;AN000;

                                ; 12/03/2024
                                %if 0

                                ;; Extended Open
                                    push    word [SS:DMAADD]
                                    push    word [SS:DMAADD+2]      ;smr;SS Override

                                ;;;;;       BAD SPOT FOR 286!!! SEGMENT ARITHMETIC!!!

                                    ; 26/07/2019

                                    ; 02/12/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
                                    ;
                                    ; (It is not necessary to call 'Align_Buffer' proc here/below because
                                    ; there is not another caller; it is better to put the code in this proc
                                    ; here instead of calling it as a subroutine; but I have modified code
                                    ; here for MSDOS 5.0 MSDOS.SYS address compatibility)

                                    ; MSDOS 6.0
                                    CALL    Align_Buffer            ;AN000;MS. align user's buffer

                                    ; 02/12/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
                                    ; MSDOS 3.3
                                    ;MOV    BX,DX                   ; copy offset
                                    ;push   cx                      ; don't stomp on count
                                    ;MOV    CL,4                    ; bits to shift bytes->para
                                    ;SHR    BX,CL                   ; get number of paragraphs
                                    ;pop    cx                      ; get count back
                                    ;MOV    AX,DS                   ; get original segment
                                    ;ADD    AX,BX                   ; get new segment
                                    ;MOV    DS,AX                   ; in seg register
                                    ;AND    DX,0Fh                  ; normalize offset
                                    ;MOV    [ss:DMAADD],DX          ; use user DX as offset       ;smr;SS Override
                                    ;MOV    [ss:DMAADD+2],DS        ; use user DS as segment for DMA
                                                                            ;smr;SS Override
                                %else
                                    ; 12/03/2024 (PCDOS 7.1 IBMDOS.COM)
                                    ;;;
                                    mov     ax,ds                   ; original segment
                                    lds     bx,[ss:DMAADD]
                                    push    bx
                                    push    ds
                                    mov     bx,dx
                                    shr     bx,1
                                    shr     bx,1
                                    shr     bx,1
                                    shr     bx,1
                                    add     ax,bx                   ; new segment
                                    and     dx,0Fh                  ; normalize offset
                                    ;mov    [ss:DMAADD],dx          ; use user DX as offset
                                    ; 23/03/2024
                                    mov     [ss:DMAADD+2],ax        ; use user DS as segment for DMA
                                    ;;;

                                %endif

                                ;;;;;       END BAD SPOT FOR 286!!! SEGMENT ARITHMETIC!!!

                                    push    ss                      ; go for DOS addressability
                                    pop     ds

                                    ; 12/03/2024 - Retro DOS v5.0
                                    ;;;
                                    mov     [DMAADD],dx
                                    ;;;

                                    CALL    SI ; DOS_READ           ; indirect call to operation

                                    pop     word [DMAADD+2]
                                    pop     word [DMAADD]
                                    ;JNC    short READ_OK           ;AN002;
                                    ;JMP    short ReadError                 ;AN002; if error, say bye bye
                                    ; 17/12/2022
                                    jc      short ReadError
                                    ; 02/12/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
                                    ;jnc    short READ_OK           ;AN002;
                                    ;jmp    short ReadError

                                READ_OK:
                                    MOV     AX,CX                   ; get correct return in correct reg
                                Read_Okj:
                                    ; 02/12/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
                                    ;;jmp   SYS_RET_OK              ; successful return
                                    ;jmp    short ok_done_j
                                    ; 17/12/2022
                                    jmp     SYS_RET_OK

                                ; 13/07/2018 - Retro DOS v3.0

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

                                ;   Input: DS:DX points to user's buffer addr
                                ;   Function: rearrange segment and offset for READ/WRITE buffer
                                ;   Output: [DMAADD] set
```

```
                                ; 12/03/2024
                                %if 0

                                ; 20/05/2019 - Retro DOS v4.0
                                ; 26/07/2019
                                ;   ; MSDOS 6.0
                                ;Align_Buffer:
                                ;   MOV     BX,DX                   ; copy offset
                                ;   push    cx                      ; don't stomp on count
                                ;   MOV     CL,4                    ; bits to shift bytes->para
                                ;   SHR     BX,CL                   ; get number of paragraphs
                                ;   pop     cx                      ; get count back
                                ;   MOV     AX,DS                   ; get original segment
                                ;   ADD     AX,BX                   ; get new segment
                                ;   MOV     DS,AX                   ; in seg register
                                ;   AND     DX,0Fh                  ; normalize offset
                                ;   MOV     [ss:DMAADD],DX          ; use user DX as offset       ;smr;SS Override
                                ;   MOV     [ss:DMAADD+2],DS        ; use user DS as segment for DMA
                                ;                                           ;smr;SS Override
                                ;   retn

                                ; 02/12/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
                                Align_Buffer:
                                    MOV     BX,DX                   ; copy offset
                                    push    cx                      ; don't stomp on count
                                    MOV     CL,4                    ; bits to shift bytes->para
                                    SHR     BX,CL                   ; get number of paragraphs
                                    pop     cx                      ; get count back
                                    MOV     AX,DS                   ; get original segment
                                    ADD     AX,BX                   ; get new segment
                                    MOV     DS,AX                   ; in seg register
                                    AND     DX,0Fh                  ; normalize offset
                                    MOV     [ss:DMAADD],DX          ; use user DX as offset       ;smr;SS Override
                                    MOV     [ss:DMAADD+2],DS        ; use user DS as segment for DMA
                                                                            ;smr;SS Override
                                    retn
                                %endif

                                ; 20/05/2019 - Retro DOS v4.0
                                ; DOSCODE:A8A0h (MSDOS 6.21, MSDOS.SYS)

                                ; 02/12/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
                                ; DOSCODE:A840h (MSDOS 5.0 MSDOS.SYS)

                                ; 12/03/2024 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMDOS.COM)
                                ; PCDOS 7.1 IBMDOS.COM - DOSCODE:0BA8Ch)

                                ; (MSDOS 6.22 MSDOS.SYS - DOSCODE:0A8A0h)
                                ; (Windows ME IO.SYS - BIOSCODE:0A2B9h)

                                ;BREAK <$WRITE - write to a file handle>
                                ;-----------------------------------------------------------------------------
                                ;
                                ;   Assembler usage:
                                ;       LDS     DX, buf
                                ;       MOV     CX, count
                                ;       MOV     BX, handle
                                ;       MOV     AH, Write
                                ;       INT     int_command
                                ;     AX has number of bytes written
                                ;   Errors:
                                ;       AX = write_invalid_handle
                                ;          = write_access_denied
                                ;
                                ;   Returns in register AX
                                ;
                                ;-----------------------------------------------------------------------------

                                _$WRITE:
000078D1 BE[783D]                   MOV     SI,DOS_WRITE
000078D4 EBA1                       JMP     short ReadDo

                                ;BREAK <$LSEEK - move r/w pointer>
                                ;-----------------------------------------------------------------------------
                                ;
                                ;   Assembler usage:
                                ;       MOV     DX, offsetlow
                                ;       MOV     CX, offsethigh
                                ;       MOV     BX, handle
                                ;       MOV     AL, method
                                ;       MOV     AH, LSeek
                                ;       INT     int_command
                                ;     DX:AX has the new location of the pointer
                                ;   Error returns:
                                ;       AX = error_invalid_handle
                                ;          = error_invalid_function
                                ;   Returns in registers DX:AX
                                ;
                                ;-----------------------------------------------------------------------------

                                ; 21/05/2019 - Retro DOS v4.0
                                ; DOSCODE:A8A5h (MSDOS 6.21, MSDOS.SYS)

                                ; 02/12/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
                                ; DOSCODE:A845h (MSDOS 5.0 MSDOS.SYS)

                                    ; 12/03/2024 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMDOS.COM)
                                    ; PCDOS 7.1 IBMDOS.COM - DOSCODE:0BA91h

                                _$LSEEK:
000078D6 E88D01                     call    CheckOwner              ; get system file entry

                                    ; 17/12/2022
                                ;LSeekError:
                                    ;JNC    short CHKOWN_OK                 ;AN002;
                                    ;JMP    short ReadError                 ;AN002; error return
                                    ; 17/12/2022
                                    ; 02/06/2019
000078D9 72A9                       jc      short ReadError
                                    ; 02/12/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
                                    ;JNC    short CHKOWN_OK                 ;AN002;
                                    ;JMP    short ReadError                 ;AN002; error return

                                CHKOWN_OK:
                                                                        ;AN002;
000078DB 3C02                       CMP     AL,2                    ; is the seek value correct?
000078DD 7607                       JBE     short LSeekDisp                 ; yes, go dispatch

                                ; 12/03/2024
                                %if 0
                                    ;mov    byte [ss:EXTERR_LOCUS],1
                                    MOV     byte [ss:EXTERR_LOCUS],errLOC_Unk ; Extended Error Locus
```

```
                                                ;smr;SS Override
                                %else
                                    ; 12/03/2024 (PCDOS 7.1 IBMDOS.COM)
                                    ;;;
000078DF E8FB99                     call    set_exerr_locus_unk   ; Extended Error Locus
                                    ;;;
                                %endif
                                    ;mov    al,1
000078E2 B001                       mov     al,error_invalid_function ; invalid method
                                    ; 02/12/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
                                LSeekError2:
000078E4 EB9E                       jmp     short ReadError

                                LSeekDisp:
000078E6 3C01                       CMP     AL,1                  ; best way to dispatch; check middle
000078E8 720A                       JB      short LSeekStore      ; just store CX:DX
000078EA 771B                       JA      short LSeekEOF        ; seek from end of file
                                    ;add    dx,[es:di+21]
000078EC 26035515                   ADD     DX,[ES:DI+SF_ENTRY.sf_position]
                                    ;adc    cx,[es:di+23]
000078F0 26134D17                   ADC     CX,[ES:DI+SF_ENTRY.sf_position+2]
                                LSeekStore:
000078F4 89C8                       MOV     AX,CX                 ; AX:DX
000078F6 92                         XCHG    AX,DX                 ; DX:AX is the correct value
                                LSeekSetpos:
                                    ;mov    [es:di+21],ax
000078F7 26894515                   MOV     [ES:DI+SF_ENTRY.sf_position],AX
                                    ;mov    [es:di+23],dx
000078FB 26895517                   MOV     [ES:DI+SF_ENTRY.sf_position+2],DX
000078FF E8758B                     call    Get_User_Stack
                                    ;mov    [si+6],dx
00007902 895406                     MOV     [SI+user_env.user_DX],DX ; return DX:AX
                                    ;jmp    SYS_RET_OK            ; successful return
                                    ; 25/06/2019
                                    ;jmp    SYS_RET_OK_clc
                                    ; 02/12/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
                                    ;jmp    SYS_RET_OK_clc
                                LSeekOk:
00007905 EBC7                       jmp     short Read_Okj

                                LSeekEOF:
                                    ;;test  word [es:di+5],8000h
                                    ;test   word [ES:DI+SF_ENTRY.sf_flags],sf_isnet
                                    ; 21/05/2019 - Retro DOS v4.0
00007907 26F6450680                 test    byte [ES:DI+SF_ENTRY.sf_flags+1],(sf_isnet>>8)
0000790C 750A                       JNZ     short Check_LSeek_Mode; Is Net
                                LOCAL_LSeek:
                                    ;add    dx,[es:di+17]
0000790E 26035511                   ADD     DX,[ES:DI+SF_ENTRY.sf_size]
                                    ;adc    cx,[es:di+19]
00007912 26134D13                   ADC     CX,[ES:DI+SF_ENTRY.sf_size+2]
00007916 EBDC                       JMP     short LSeekStore      ; go and set the position

                                Check_LSeek_Mode:
                                    ;;test  word [es:di+2],8000h
                                    ;test   word [ES:DI+SF_ENTRY.sf_mode],sf_isFCB
                                    ; 21/05/2019
00007918 26F6450380                 test    byte [ES:DI+SF_ENTRY.sf_mode+1],(sf_isFCB>>8)
0000791D 75EF                       JNZ     short LOCAL_LSeek     ; FCB treated like local file
                                    ;mov    ax,[es:di+2]
0000791F 268B4502                   MOV     AX,[ES:DI+SF_ENTRY.sf_mode]
                                    ;and    ax,0F0h
00007923 25F000                     AND     AX,SHARING_MASK
                                    ;cmp    ax,40h
00007926 83F840                     CMP     AX,SHARING_DENY_NONE
00007929 7405                       JZ      short NET_LSEEK               ; LSEEK exported in this mode
                                    ;cmp    ax,30h
0000792B 83F830                     CMP     AX,SHARING_DENY_READ
0000792E 75DE                       JNZ     short LOCAL_LSeek     ; Treated like local Lseek
                                NET_LSEEK:
                                ;   JMP     short LOCAL_LSeek
                                ; REMOVE ABOVE INSTRUCTION TO ENABLE DCR 142
                                    ;CallInstall Net_Lseek,MultNET,33
                                    ;JNC    short LSeekSetPos

00007930 B82111                     mov     ax,1121h
00007933 CD2F                       int     2Fh    ; Multiplex - NETWORK REDIRECTOR - SEEK FROM END OF REMOTE FILE
                                                   ; CX:DX = offset (in bytes) from end
                                                   ; ES:DI -> SFT, SFT DPB field -> DPB of drive with file
                                                   ; SS = DOS CS
                                                   ; Return: CF set on error
                                                   ; CF clear if successful, DX:AX = new file position
00007935 73C0                       jnb     short LSeekSetpos
                                    ; 02/12/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
                                    ;jmp    SYS_RET_ERR
                                ;LSeekError3:
                                    ; 17/12/2022
                                LSeekError:
                                    ;jmp    short LSeekError2
                                DupErr: ; 17/12/2022
00007937 E93E8D                     jmp     SYS_RET_ERR

                                ; 21/05/2019 - Retro DOS v4.0
                                ; DOSCODE:A95Bh (MSDOS 6.21, MSDOS.SYS)

                                ; 02/12/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
                                ; DOSCODE:A8FBh (MSDOS 5.0 MSDOS.SYS)

                                ;BREAK <$DUP - duplicate a jfn>
                                ;--------------------------------------------------------------------------
                                ;
                                ;   Assembler usage:
                                ;       MOV     BX, fh
                                ;       MOV     AH, Dup
                                ;       INT     int_command
                                ;     AX has the returned handle
                                ;   Errors:
                                ;       AX = dup_invalid_handle
                                ;          = dup_too_many_open_files
                                ;
                                ;--------------------------------------------------------------------------

                                    ; 12/03/2024 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMDOS.COM)
                                    ; PCDOS 7.1 IBMDOS.COM - DOSCODE:0BBEBh

                                    ; (MSDOS 6.22 MSDOS.SYS - DOSCODE:0A95Bh)
                                    ; (Windows ME IO.SYS - BIOSCODE:0A3EFh)

                                _$DUP:
0000793A 89D8                       MOV     AX,BX                 ; save away old handle in AX
0000793C E8D4FD                     call    JFNFree               ; free handle? into ES:DI, new in BX
                                DupErrorCheck:
0000793F 72F6                       JC      short DupErr          ; nope, bye
00007941 06                         push    es
```

```
00007942 57                         push    di                      ; save away SFT
00007943 5E                         pop     si                      ; into convenient place DS:SI
00007944 1F                         pop     ds
00007945 93                         XCHG    AX,BX                   ; get back old handle
00007946 E81D01                     call    CheckOwner              ; get sft in ES:DI
00007949 72EC                       JC      short DupErr            ; errors go home
0000794B E870B7                     call    DOS_Dup_Direct
0000794E E865FD                     call    pJFNFromHandle          ; get pointer
00007951 268A1D                     MOV     BL,[ES:DI]              ; get SFT number
00007954 881C                       MOV     [SI],BL                 ; stuff in new SFT
                                    ; 02/12/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
                                    ;jmp    SYS_RET_OK              ; and go home
00007956 EB5A                       jmp     short ok_ret

                                    ; 17/12/2022
                                ;DupErr:
                                    ; 02/12/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
                                    ;;jmp   SYS_RET_ERR
                                    ;jmp    short ft_error

                                ;BREAK <$DUP2 - force a dup on a particular jfn>
                                ;--------------------------------------------------------------------------
                                ;
                                ;   Assembler usage:
                                ;       MOV     BX, fh
                                ;       MOV     CX, newfh
                                ;       MOV     AH, Dup2
                                ;       INT     int_command
                                ;   Error returns:
                                ;       AX = error_invalid_handle
                                ;
                                ;--------------------------------------------------------------------------

                                _$DUP2:
00007958 53                         push    bx
00007959 51                         push    cx                      ; save source
0000795A 89CB                       MOV     BX,CX                   ; get one to close
0000795C E80EFE                     call    _$CLOSE                 ; close destination handle
0000795F 5B                         pop     bx
00007960 58                         pop     ax                      ; old in AX, new in BX
00007961 E852FD                     call    pJFNFromHandle          ; get pointer
00007964 EBD9                       JMP     short DupErrorCheck     ; check error and do dup

                                ;BREAK <FileTimes - modify write times on a handle>
                                ;--------------------------------------------------------------------------
                                ;
                                ;   Assembler usage:
                                ;       MOV AH, FileTimes (57H)
                                ;       MOV AL, func
                                ;       MOV BX, handle
                                ;   ; if AL = 1 then then next two are mandatory
                                ;       MOV CX, time
                                ;       MOV DX, date
                                ;       INT 21h
                                ;   ; if AL = 0 then CX/DX has the last write time/date
                                ;   ; for the handle.
                                ;
                                ;   AL=02           get extended attributes
                                ;      BX=handle
                                ;      CX=size of buffer (0, return max size )
                                ;      DS:SI query list (si=-1, selects all EA)
                                ;      ES:DI buffer to hold EA list
                                ;
                                ;   AL=03           get EA name list
                                ;      BX=handle
                                ;      CX=size of buffer (0, return max size )
                                ;      ES:DI buffer to hold name list
                                ;
                                ;   AL=04           set extended attributes
                                ;      BX=handle
                                ;      ES:DI buffer of EA list
                                ;
                                ;
                                ;   Error returns:
                                ;       AX = error_invalid_function
                                ;          = error_invalid_handle
                                ;
                                ;--------------------------------------------------------------------------

                                ; 21/05/2019 - Retro DOS v4.0
                                ; DOSCODE:A90Dh (MSDOS 6.21, MSDOS.SYS)

                                ; 02/12/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
                                ; DOSCODE:A8ADh (MSDOS 5.0 MSDOS.SYS)

                                    ; 12/03/2024 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMDOS.COM)
                                    ; PCDOS 7.1 IBMDOS.COM - DOSCODE:0BAF5h

                                    ; (MSDOS 6.22 MSDOS.SYS - DOSCODE:0A90Dh)
                                    ; (Windows ME IO.SYS - BIOSCODE:0A327h)

                                _$FILE_TIMES:

                                ; 12/03/2024
                                %if 0
                                    ; 13/07/2018 - Retro DOS v3.0

                                    ; MSDOS 3.3
                                    ;cmp    al,2                    ; correct subfunction ?
                                    ;jb     short ft1

                                    ;;mov   byte [ss:EXTERR_LOCUS], 1
                                    ;mov    byte [ss:EXTERR_LOCUS],errLOC_Unk ; Extended Error Locus
                                                                              ;SS Overr
                                    ;;mov   al,1
                                    ;mov    al,error_invalid_function ; give bad return
                                    ;jmp    SYS_RET_ERR

                                    ; MSDOS 6.0
                                    cmp     al,2                    ; correct subfunction ?
                                    jae     short inval_func
                                ;ft1:
                                    call    CheckOwner              ; get sft
                                    ; 17/12/2022
                                    jc      short LSeekError        ; bad handle

                                    or      al,al                   ; get time/date ?
                                    jnz     short ft_set_time

                                ;------ here we get the time & date from the sft for the user

                                    cli                             ; is this cli/sti reqd ? BUGBUG
                                    ;mov    cx,[es:di+13]
                                    mov     cx,[es:di+SF_ENTRY.sf_time] ; get the time
```

```
                            ;mov    dx,[es:di+15]
                            mov     dx,[es:di+SF_ENTRY.sf_date] ;  & date
                            sti
                            call    Get_User_Stack
                            ;mov    [si+4],cx
                            mov     [si+user_env.user_CX],cx
                            ;mov    [si+6],dx
                            mov     [si+user_env.user_DX],dx
                            jmp     short ok_ret

                        ;------ here we set the time in sft

                        ft_set_time:
                            call    ECritSFT
                            ;mov    [es:di+13],cx
                            mov     [es:di+SF_ENTRY.sf_time],cx ; drop in new time
                            ;mov    [es:di+15],dx
                            mov     [es:di+SF_ENTRY.sf_date],dx ;  and date

                            xor     ax, ax
                            call    far [ss:JShare+(14*4)] ; 14 = ShSU   ; SS Override

                        ;------ set the flags in SFT entry
                            ;and    word [es:di+5],0FFBFh
                            ; 18/12/2022
                            ;and    byte [es:di+5],0BFh
                            and     byte [es:di+SF_ENTRY.sf_flags],~devid_file_clean
                            ;and    word [es:di+SF_ENTRY.sf_flags],~devid_file_clean
                                                                    ; mark file as dirty
                            ;or     word [es:di+5],4000h
                            ; 17/12/2022
                            ;or     byte [es:di+6],40h
                            or      byte [es:di+SF_ENTRY.sf_flags+1],(sf_close_nodate>>8)
                            ;or     word [es:di+SF_ENTRY.sf_flags],sf_close_nodate
                                                                    ; ask close not to
                                                                    ;   bother about date
                                                                    ;   and time
                            call    LCritSFT

                        ok_ret:
                            ; 02/12/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
                            ; 17/12/2022
                            jmp     SYS_RET_OK
                            ;jmp    short LSeekOk

                        inval_func:
                            ;mov    byte [ss:EXTERR_LOCUS],1
                            mov     byte [ss:EXTERR_LOCUS],errLOC_Unk ; Extended Error Locus
                                                                      ;SS Overr
                            ;mov    al,1
                            mov     al,error_invalid_function ; give bad return
                            ; 02/12/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
                        ft_error:
                            ;;jmp   SYS_RET_ERR
                            ;jmp    short LSeekError3
                            ; 17/12/2022
                            jmp     short LSeekError

                        %else
                            ; 12/03/2024 - Retro DOS v5.0
                            ; PCDOS 7.1 IBMDOS.COM
                            ;;;                     ; ((Ref: Ralf Brown's Intterupt List))
00007966 3C07               cmp     al,7            ; SET CREATION DATE AND TIME
                                                    ; 6 = GET CREATION DATE AND TIME
                                                    ; 5 = SET LAST ACCESS DATE AND TIME
                            ;jnb    short LSeekError1
                            ; 12/03/2024
00007968 737B               jnb     short inval_func
0000796A 3C04               cmp     al,4            ; GET LAST ACCESS DATE AND TIME
0000796C 7304               jnb     short ft1
0000796E 3C02               cmp     al,2            ; 0 = GET FILE'S LAST-WRITTEN DATE AND TIME
                                                    ; 1 = SET FILE'S LAST-WRITTEN DATE AND TIME
                            ;jnb    short LSeekError1 ; 2,3 -> invalid
                            ; 12/03/2024
00007970 7373               jnb     short inval_func
                        ft1:
00007972 E8F100             call    CheckOwner      ; get sft
00007975 72C0               jc      short LSeekError ; bad handle

00007977 3C01               cmp     al,1
00007979 773A               ja      short ft2       ; PCDOS 7.1 (MSDOS 7) sub function

0000797B 08C0               or      al,al           ; get time/date ?
0000797D 7514               jnz     short ft_set_time ; no,set

                            ;lds    cx,[es:di+0Dh]
0000797F 26C54D0D           lds     cx,[es:di+SF_ENTRY.sf_time]
                                                    ; get the time
00007983 8CDA               mov     dx,ds           ; & date ; [es:di+SF_ENTRY.sf_date]
00007985 E8EF8A             call    Get_User_Stack
                            ;mov    [si+4],cx
00007988 894C04             mov     [si+user_env.user_CX],cx
0000798B 895406             mov     [si+6],dx
0000798E 895406             mov     [si+user_env.user_DX],dx

00007991 EB1F               jmp     short ft_ok

                        ft_set_time:
00007993 E84C9F             call    ECritSFT

                            ;mov    [es:di+0Dh],cx
00007996 26894D0D           mov     [es:di+SF_ENTRY.sf_time],cx ; drop in new time
                            ;mov    [es:di+0Fh],dx
0000799A 2689550F           mov     [es:di+SF_ENTRY.sf_date],dx ; and date

0000799E 31C0               xor     ax,ax ; 0
                            ;call   ss:ShSU
000079A0 36FF1E[C800]       call    far [ss:JShare+(14*4)] ; 14 = ShSU

                            ;;and   word [es:di+5],0FFBFh
                            ;and    word [es:di+SF_ENTRY.sf_flags],~devid_file_clean
000079A5 26806505BF         and     byte [es:di+SF_ENTRY.sf_flags],~devid_file_clean ; 0BFh

                            ;;or    word [es:di+5],4000h
                            ;or     word [es:di+SF_ENTRY.sf_flags],sf_close_nodate
000079AA 26804D0640         or      byte [es:di+SF_ENTRY.sf_flags+1],(sf_close_nodate>>8) ; 40h

000079AF E85D9F             call    LCritSFT
                        ft_ok:
                        ok_ret:             ; 12/03/2024
000079B2 E9B98C             jmp     SYS_RET_OK

                        ft2:
                            ;test   byte [es:di+5],80h
```

```
44875 000079B5 26F6450580               test    byte [es:di+SF_ENTRY.sf_flags],devid_device
44876 000079BA 7505                     jnz     short ft3       ; device
44877
44878 000079BC E8839E                   call    IsSFTNet
44879                                 ;ft3:
44880 000079BF 7415                     jz      short ft5       ; local file
44881                                 ft3:    ; 12/03/2024
44882 000079C1 A801                     test    al,1            ; 0 = GET, 1 = SET
44883 000079C3 750F                     jnz     short ft_okj    ; SET
44884
44885                                   ;mov    dx,[es:di+0Fh]
44886 000079C5 268B550F                 mov     dx,[es:di+SF_ENTRY.sf_date]
44887                                   ;;;;
44888                                 ft7:
44889                                   ; 12/03/2024
44890 000079C9 29C9                     sub     cx,cx ; 0 ; Retro DOS v5.0
44891                                                     ; (Windows ME IO.SYS BIOSCODE:A348h)
44892                                   ;;;;
44893                                 ft4:
44894 000079CB E8A98A                   call    Get_User_Stack
44895                                   ;mov    [si+4],cx
44896 000079CE 894C04                   mov     [si+user_env.user_CX],cx ; time
44897                                   ;mov    [si+6],dx
44898 000079D1 895406                   mov     [si+user_env.user_DX],dx ; date
44899                                 ft_okj:
44900 000079D4 EBDC                     jmp     short ft_ok
44901
44902                                 ft5:
44903 000079D6 16                       push    ss              ; Retrieve the directory entry for the file
44904 000079D7 1F                       pop     ds
44905 000079D8 50                       push    ax
44906 000079D9 52                       push    dx
44907 000079DA 51                       push    cx              ; ES:DI point to SFT
44908 000079DB E8BBBE                   call    DirFromSFT      ; locate a directory entry given an SFT
44909 000079DE 59                       pop     cx
44910 000079DF 5A                       pop     dx
44911 000079E0 730B                     jnc     short ft6       ; ES:DI point to entry
44912                                                           ; (DS:SI point to SFT)
44913                                                           ; (ES:BX point to buffer header)
44914 000079E2 59                       pop     cx
44915                                 ;ft_error:
44916                                   ;jmp    short LSeekError2
44917                                   ; 12/03/2024
44918                                   ;jmp    short LseekError
44919 000079E3 EB05                     jmp     short ft_error
44920
44921                                   ;;;;
44922                                   ; 12/03/2024 - Retro DOS v5.0
44923                                 inval_func:
44924 000079E5 E8F598                   call    set_exerr_locus_unk ; Extended Error Locus
44925
44926                                   ;mov    al,1
44927 000079E8 B001                     mov     al,error_invalid_function ; give bad return
44928                                 ft_error:
44929                                   ;jmp    short LSeekError
44930                                   ; 12/03/2024
44931 000079EA E98B8C                   jmp     SYS_RET_ERR
44932                                   ;;;;
44933                                 ft6:
44934 000079ED 58                       pop     ax
44935 000079EE A801                     test    al,1            ; SET ?
44936 000079F0 7512                     jnz     short ft8       ; yes
44937
44938                                   ; 12/03/2024 (ft7:, cx=0)
44939                                   ;xor    cx,cx   ; 0     ; (always) last access time = 0
44940
44941                                   ;mov    dx,[es:di+12h]
44942 000079F2 268B5512                 mov     dx,[es:di+dir_entry.dir_lstaccdate]
44943
44944 000079F6 3C06                     cmp     al,6            ; GET CREATION DATE AND TIME ?
44945 000079F8 75CF                     jne     short ft7       ; no,GET LAST ACCESS DATE AND TIME
44946
44947                                   ;mov    cx,[es:di+0Eh]
44948 000079FA 268B4D0E                 mov     cx,[es:di+dir_entry.dir_crttime]
44949                                   ;mov    dx,[es:di+10h]
44950 000079FE 268B5510                 mov     dx,[es:di+dir_entry.dir_crtdate]
44951                                 ;ft7:
44952 00007A02 EBC7                     jmp     short ft4
44953
44954                                 ft8:            ; check date (set) values
44955 00007A04 F6C21F                   test    dl,1Fh          ; DAY (1 to 31) > 0 ?
44956 00007A07 7504                     jnz     short ft9       ; yes,valid
44957
44958                                 ft_err_invd:
44959                                   ;mov    al,0Dh
44960 00007A09 B00D                     mov     al,error_invalid_data
44961                                 ft_errj:
44962 00007A0B EBDD                     jmp     short ft_error
44963
44964                                 ft9:
44965 00007A0D F7C2E001                 test    dx,1E0h         ; MONTH (1 to 12) > 0 ?
44966 00007A11 74F6                     jz      short ft_err_invd ; no,invalid
44967 00007A13 52                       push    dx
44968 00007A14 81E2E001                 and     dx,1E0h         ; isolate MONTH
44969 00007A18 81FA8001                 cmp     dx,180h         ; > 12 ?
44970 00007A1C 5A                       pop     dx
44971 00007A1D 77EA                     ja      short ft_err_invd ; yes,invalid
44972 00007A1F 3C05                     cmp     al,5            ; SET LAST ACCESS DATE AND TIME ?
44973 00007A21 7506                     jnz     short ft10      ; no,SET CREATION DATE AND TIME
44974
44975                                   ;mov    [es:di+12h],dx
44976 00007A23 26895512                 mov     [es:di+dir_entry.dir_lstaccdate],dx
44977 00007A27 EB20                     jmp     short ft11
44978
44979                                 ft10:               ; check time (set) values
44980 00007A29 89C8                     mov     ax,cx
44981 00007A2B 251FF8                   and     ax,0F81Fh       ; isolate seconds/2 and hour
44982 00007A2E 80FCB8                   cmp     ah,0B8h         ; HOUR > 23 ?
44983 00007A31 77D6                     ja      short ft_err_invd ; yes,invalid
44984 00007A33 3C1D                     cmp     al,1Dh          ; SECONDS/2 > 29 (count of 2 seconds)
44985 00007A35 77D2                     ja      short ft_err_invd ; yes,invalid
44986 00007A37 89C8                     mov     ax,cx
44987 00007A39 25E007                   and     ax,7E0h         ; isolate MINUTE
44988 00007A3C 3D6007                   cmp     ax,760h         ; MINUTE > 59 ?
44989 00007A3F 77C8                     ja      short ft_err_invd ; yes,invalid
44990
44991                                   ;mov    [es:di+10h],dx
44992 00007A41 26895510                 mov     [es:di+dir_entry.dir_crtdate],dx
44993                                   ;mov    [es:di+0Eh],cx
44994 00007A45 26894D0E                 mov     [es:di+dir_entry.dir_crttime],cx
44995                                 ft11:
44996                                   ;test   byte [es:bx+5],40h
44997 00007A49 26F6470540               test    byte [es:bx+BUFFINFO.buf_flags],buf_dirty
44998 00007A4E 7508                     jnz     short ft12
```

```
44999
45000 00007A50 E877F1                   call    INC_DIRTY_COUNT
45001
45002                                   ;or     byte [es:bx+5],40h
45003 00007A53 26804F0540               or      byte [es:bx+BUFFINFO.buf_flags],buf_dirty
45004                                   ft12:
45005 00007A58 06                       push    es
45006 00007A59 1F                       pop     ds
45007 00007A5A 89DF                     mov     di,bx           ; DS:DI - pointer to buffer
45008 00007A5C B0FF                     mov     al,0FFh         ; Drive number
45009                                                           ; -1 means do not check for drive
45010 00007A5E E886F0                   call    CHECKFLUSH
45011 00007A61 7287                     jc      short ft_error
45012                                   ;ok_ret:    ; 12/03/2024
45013 00007A63 E9088C                   jmp     SYS_RET_OK
45014                                   ;;;
45015                                   %endif
45016
45017                                   ;Break      <CheckOwner - verify ownership of handles from server>
45018                                   ;---------------------------------------------------------------------------
45019                                   ;   CheckOwner - Due to the ability of the server to close file handles for a
45020                                   ;   process without the process knowing it (delete/rename of open files, for
45021                                   ;   example), it is possible for the redirector to issue a call to a handle
45022                                   ;   that it does not rightfully own. We check here to make sure that the
45023                                   ;   issuing process is the owner of the SFT. At the same time, we do a
45024                                   ;   SFFromHandle to really make sure that the SFT is good.
45025                                   ;
45026                                   ;   ENTRY   BX has the handle
45027                                   ;           User_ID is the current user
45028                                   ;   EXIT    Carry Clear => ES:DI points to SFT
45029                                   ;           Carry Set => AX has error code
45030                                   ;   USES    none
45031                                   ;---------------------------------------------------------------------------
45032
45033                                       ; 02/12/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
45034                                       ; 21/05/2019 - Retro DOS v4.0
45035                                   CheckOwner:
45036                                       ; 13/07/2018 - Retro DOS v3.0
45037
45038 00007A66 E86AFC                   call    SFFromHandle
45039 00007A69 721B                     jc      short co_ret_label    ; retc
45040
45041 00007A6B 50                       push    ax
45042
45043                                   ; MSDOS 6.0
45044
45045                                   ;SR; WIN386 patch - Do not check for USER_ID for using handles since these
45046                                   ;SR; are shared across multiple VMs in win386.
45047
45048 00007A6C 36F606[5B0F]01           test    byte [ss:IsWin386],1 ; 02/06/2019
45049 00007A72 7404                     jz      short no_win386             ;win386 is not present
45050 00007A74 31C0                     xor     ax,ax                   ;set the zero flag
45051 00007A76 EB08                     jmp     short _skip_win386
45052
45053                                   no_win386:
45054 00007A78 36A1[3E03]               mov     ax,[SS:USER_ID]             ;smr;SS Override
45055                                   ;cmp    ax,[es:di+47]
45056 00007A7C 263B452F                 cmp     ax,[es:di+SF_ENTRY.sf_UID]
45057
45058                                   _skip_win386:
45059 00007A80 58                       pop     ax
45060
45061                                   ; 17/12/2022
45062 00007A81 7403                     jz      short co_ret_label
45063                                   ; 02/12/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
45064                                   ;jnz    short CheckOwner_err
45065                                   ;retn
45066
45067                                   CheckOwner_err:
45068                                       ;mov    al,6
45069 00007A83 B006                     mov     al,error_invalid_handle
45070 00007A85 F9                       stc
45071
45072                                   co_ret_label:
45073 00007A86 C3                       retn
45074
45075                                   ;============================================================================
45076                                   ; MACRO.ASM, MSDOS 6.0, 1991
45077                                   ;============================================================================
45078                                   ; Retro     DOS v3.0 - 11/07/2018
45079                                   ; 21/05/2019 - Retro DOS v4.0
45080                                   ; 13/03/2024 - Retro DOS v5.0
45081
45082                                   ;   TITLE   MACRO - Pathname and macro related internal routines
45083                                   ;   NAME    MACRO
45084
45085                                   ;   Microsoft Confidential
45086                                   ;   Copyright (C) Microsoft Corporation 1991
45087                                   ;   All Rights Reserved.
45088
45089                                   ;** MACRO.ASM
45090                                   ;
45091                                   ;   $AssignOper
45092                                   ;   FIND_DPB
45093                                   ;   InitCDS
45094                                   ;   $UserOper
45095                                   ;   GetVisDrv
45096                                   ;   GetThisDrv
45097                                   ;   GetCDSFromDrv
45098                                   ;
45099                                   ;   Revision history:
45100                                   ;
45101                                   ;   Created: MZ 4 April 1983
45102                                   ;            MZ 18 April 1983   Make TransFCB handle extended FCBs
45103                                   ;            AR 2 June 1983     Define/Delete macro for NET redir.
45104                                   ;            MZ 3 Nov 83       Fix InitCDS to reset length to 2
45105                                   ;            MZ 4 Nov 83       Fix NetAssign to use STRLEN only
45106                                   ;            MZ 18 Nov 83      Rewrite string processing for subtree
45107                                   ;                              aliasing.
45108                                   ;
45109                                   ;   MSDOS performs several types of name translation. First, we maintain for
45110                                   ;   each valid drive letter the text of the current directory on that drive.
45111                                   ;   For invalid drive letters, there is no current directory so we pretend to
45112                                   ;   be at the root. A current directory is either the raw local directory
45113                                   ;   (consisting of drive:\path) or a local network directory (consisting of
45114                                   ;   \\machine\path. There is a limit on the point to which a .. is allowed.
45115                                   ;
45116                                   ;   Given a path, MSDOS will transform this into a real from-the-root path
45117                                   ;   without . or .. entries. Any component that is > 8.3 is truncated to
45118                                   ;   this and all * are expanded into ?'s.
45119                                   ;
45120                                   ;   The second part of name translation involves subtree aliasing. A list of
45121                                   ;   subtree pairs is maintained by the external utility SUBST. The results of
45122                                   ;   the previous 'canonicalization' are then examined to see if any of the
```

```
45123                                   ;   subtree pairs is a prefix of the user path. If so, then this prefix is
45124                                   ;   replaced with the other subtree in the pair.
45125                                   ;
45126                                   ;   A third part involves mapping this "real" path into a "physical" path.  A
45127                                   ;   list of drive/subtree pairs are maintained by the external utility JOIN.
45128                                   ;   The output of the previous translation is examined to see if any of the
45129                                   ;   subtrees in this list are a prefix of the string. If so, then the prefix
45130                                   ;   is replaced by the appropriate drive letter. In this manner, we can
45131                                   ;   'mount' one device under another.
45132                                   ;
45133                                   ;   The final form of name translation involves the mapping of a user's
45134                                   ;   logical drive number into the internal physical drive. This is
45135                                   ;   accomplished by converting the drive number into letter:CON, performing
45136                                   ;   the above translation and then converting the character back into a drive
45137                                   ;   number.
45138                                   ;
45139                                   ;   There are two main entry points: TransPath and TransFCB. TransPath will
45140                                   ;   take a path and form the real text of the pathname with all . and ..
45141                                   ;   removed. TransFCB will translate an FCB into a path and then invoke
45142                                   ;   TransPath.
45143                                   ;
45144                                   ;   A000   version 4.00  Jan. 1988
45145
45146                                   ;Installed = TRUE
45147
45148                                   ;   I_need  ThisCDS,DWORD         ; pointer to CDS used
45149                                   ;   I_need  CDSAddr,DWORD         ; pointer to CDS table
45150                                   ;   I_need  CDSCount,BYTE         ; number of CDS entries
45151                                   ;   I_need  CurDrv,BYTE           ; current macro assignment (old
45152                                   ;                                 ; current drive)
45153                                   ;   I_need  NUMIO,BYTE           ; Number of physical drives
45154                                   ;   I_need  fSharing,BYTE         ; TRUE => no redirection allowed
45155                                   ;   I_need  DummyCDS,80h          ; buffer for dummy cds
45156                                   ;   I_need  DIFFNAM,BYTE          ; flag for MyName being set
45157                                   ;   I_need  MYNAME,16             ; machine name
45158                                   ;   I_need  MYNUM,WORD            ; machine number
45159                                   ;   I_need  DPBHEAD,DWORD         ; beginning of DPB chain
45160                                   ;   I_need  EXTERR_LOCUS,BYTE     ; Extended Error Locus
45161                                   ;   I_need  DrvErr,BYTE           ; drive error
45162
45163                                   ;BREAK <$AssignOper -- Set up a Macro>
45164                                   ;-----------------------------------------------------------------------------
45165                                   ; Inputs:
45166                                   ;   AL = 00 get assign mode                 (ReturnMode)
45167                                   ;   AL = 01 set assign mode                 (SetMode)
45168                                   ;   AL = 02 get attach list entry           (GetAsgList)
45169                                   ;   AL = 03 Define Macro (attch start)
45170                                   ;       BL = Macro type
45171                                   ;          = 0 alias
45172                                   ;          = 1 file/device
45173                                   ;          = 2 drive
45174                                   ;          = 3 Char device -> network
45175                                   ;          = 4 File device -> network
45176                                   ;       DS:SI -> ASCIZ source name
45177                                   ;       ES:DI -> ASCIZ destination name
45178                                   ;   AL = 04 Cancel Macro
45179                                   ;       DS:SI -> ASCIZ source name
45180                                   ;   AL = 05 Modified get attach list entry
45181                                   ;   AL = 06 Get ifsfunc item
45182                                   ;   AL = 07 set in_use of a drive's CDS
45183                                   ;        DL = drive number, 0=default  0=A,,
45184                                   ;   AL = 08 reset in_use of a drive's CDS
45185                                   ;        DL = drive number, 0=A, 1=B,,,
45186                                   ; Function:
45187                                   ;   Do macro stuff
45188                                   ; Returns:
45189                                   ;   Std Xenix style error return
45190                                   ;-----------------------------------------------------------------------------
45191
45192                                       ; 02/12/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
45193                                       ; 21/05/2019 - Retro DOS v4.0
45194
45195                                   _$AssignOper:
45196                                       ; MSDOS 6.0
45197 00007A87 3C07                       CMP     AL,7                    ; set in_use ?          ;AN000;
45198 00007A89 7525                       JNZ     short chk08             ; no                    ;AN000;
45199                                   srinuse:                                                  ;AN000;
45200 00007A8B 50                         PUSH    AX                      ; save al               ;AN000;
45201 00007A8C 88D0                       MOV     AL,DL                   ; AL= drive id          ;AN000;
45202 00007A8E E84B01                     CALL    GetCDSFromDrv           ; ds:si -> cds          ;AN000;
45203 00007A91 58                         POP     AX                      ;                       ;AN000;
45204 00007A92 7216                       JC      short baddrv            ; bad drive             ;AN000;
45205                                     ;cmp    word [si+45h],0
45206 00007A94 837C4500                   CMP     WORD [SI+curdir.devptr],0     ; dpb ptr =0 ?    ;AN000;
45207 00007A98 7410                       JZ      short baddrv            ;     no                ;AN000;
45208 00007A9A 3C07                       CMP     AL,7                    ; set ?                 ;AN000;
45209 00007A9C 7506                       JNZ     short resetdrv          ; no                    ;AN000;
45210                                     ;or     word [si+43h],4000h
45211                                     ; 17/12/2022
45212                                     ;or     byte [si+44h],40h
45213 00007A9E 804C4440                   or      byte [SI+curdir.flags+1],(curdir_inuse>>8)
45214                                     ;OR     word [SI+curdir.flags],curdir_inuse ; set in_use    ;AN000;
45215 00007AA2 EB19                       JMP     SHORT okdone            ;                       ;AN000;
45216                                   resetdrv:
45217                                     ;and    word [si+43h],0BFFFh                              ;AN000;
45218                                     ; 18/12/2022
45219 00007AA4 806444BF                   and     byte [SI+curdir.flags+1],0BFh ; (~curdir_inuse)>>8
45220                                     ;AND    word [SI+curdir.flags],~curdir_inuse ; reset in_use ;AN000;
45221 00007AA8 EB13                       JMP     SHORT okdone            ;                       ;AN000;
45222
45223                                     ; 17/12/2022
45224                                   baddrv:                                                   ;AN000;
45225 00007AAA B80F00                     MOV     AX,error_invalid_drive  ; error                 ;AN000;
45226
45227                                     ; 02/12/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
45228                                     ;JMP    SHORT ASS_ERR           ;                       ;AN000;
45229                                     ; 17/12/2022
45230                                     ; 21/05/2019
45231                                   ASS_ERR:
45232 00007AAD E9C88B                     jmp     SYS_RET_ERR
45233
45234                                   chk08:                                                            ;AN000;
45235 00007AB0 3C08                       CMP     AL,8                    ; reset inuse ?         ;AN000;
45236 00007AB2 74D7                       JZ      short srinuse           ; yes                   ;AN000;
45237
45238                                   ;IF       NOT INSTALLED
45239                                     ;transfer NET_ASSOPER
45240                                   ;ELSE
45241                                     ; MSDOS 3.3 (& MSDOS 6.0)
45242 00007AB4 50                         PUSH    AX
45243                                     ;mov    ax,111Eh
45244                                     ;MOV    AX,(MultNET SHL 8) OR 30
45245 00007AB5 B81E11                     mov     ax,(MultNET*256)+30
45246 00007AB8 CD2F                       int     2Fh     ; Multiplex - NETWORK REDIRECTOR - DO REDIRECTION
```

```
                                                ; SS = DOS CS
                                                ; STACK: WORD function to execute
                                                ; Return: CF set on error, AX = error code
                                                ; STACK unchanged
00007ABA 5B                             POP     BX                      ; Don't zap error code in AX
00007ABB 72F0                           JC      short ASS_ERR
                                    okdone:
00007ABD E9AE8B                         jmp     SYS_RET_OK

                                        ; 17/12/2022
                                        ; 02/12/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
                                    ;ASS_ERR:
                                        ;jmp    SYS_RET_ERR

                                      ;ENDIF

                                    ;Break <FIND_DPB - Find a DPB from a drive number>
                                    ;--------------------------------------------------------------------------
                                    ;** FIND_DPB - Find a DPB from a Drive #
                                    ;
                                    ;   ENTRY   AL has drive number A = 0
                                    ;   EXIT    'C' set
                                    ;               No DPB for this drive number
                                    ;           'C' clear
                                    ;               DS:SI points to DPB for drive
                                    ;   USES    SI, DS, Flags
                                    ;--------------------------------------------------------------------------

                                        ; 21/05/2019 - Retro DOS v4.0
                                    FIND_DPB:
00007AC0 36C536[2600]                   LDS     SI,[SS:DPBHEAD]             ;smr;SS Override
                                    fdpb5:
00007AC5 83FEFF                         CMP     SI,-1
00007AC8 7409                           JZ      short fdpb10
00007ACA 3A04                           cmp     al,[si]
                                        ;CMP    AL,[SI+DPB.DRIVE]
00007ACC 7406                           jz      short ret_label15       ; Carry clear (retz)
                                        ;;lds   si,[si+18h] ; MSDOS 3.3
                                        ;lds    si,[si+19h] ; MSDOS 6.0
00007ACE C57419                         LDS     SI,[SI+DPB.NEXT_DPB]
00007AD1 EBF2                           JMP     short fdpb5
                                    fdpb10:
00007AD3 F9                             STC
                                    ret_label15:
00007AD4 C3                             retn

                                    ;   Break <InitCDS - set up an empty CDS>
                                    ;--------------------------------------------------------------------------
                                    ;** InitCDS - Setup an Empty CDS
                                    ;
                                    ;   ENTRY   ThisCDS points to CDS
                                    ;           AL has uppercase drive letter
                                    ;   EXIT    ThisCDS is now empty
                                    ;           (ES:DI) = CDS
                                    ;           'C' set if no DPB associated with drive
                                    ;   USES    AH,ES,DI, Flags
                                    ;--------------------------------------------------------------------------

                                    ; 21/05/2019 - Retro DOS v4.0
                                    ; DOSCODE:A9FDh (MSDOS 6.21, MSDOS.SYS)

                                    ; 02/12/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
                                    ; DOSCODE:A99Dh (MSDOS 5.0, MSDOS.SYS)

                                        ; 13/03/2024 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMDOS.COM)
                                        ; PCDOS 7.1 IBMDOS.COM - DOSCODE:0BC91h

                                    InitCDS:
                                        ; 19/08/2018
                                        ; 05/08/2018 - Retro DOS v3.0
                                        ; MSDOS 6.0
00007AD5 50                             push    ax                      ; save (AL) for caller
00007AD6 36C43E[A205]                   LES     DI,[SS:THISCDS]             ; (es:di) = CDS address
                                        ;mov    word [es:di+67],0
00007ADB 26C745430000                   MOV     word [ES:DI+curdir.flags],0 ; "free" CDS
00007AE1 2C40                           SUB     AL,"A"-1                ; A = 1
00007AE3 363806[4600]                   CMP     [SS:NUMIO],AL           ;smr;SS Override
00007AE8 7236                           JC      short icdsx             ; Drive does not map a physical drive
00007AEA 48                             dec     ax                      ; (AL) = 0 if A, 1 if B, etc.
00007AEB 50                             PUSH    AX                      ; save drive number for later
00007AEC 0441                           add     al,"A"
00007AEE B43A                           MOV     AH,':'
00007AF0 268905                         mov     [ES:DI],ax
                                        ;MOV    [ES:DI+curdir.text],AX      ; set "x:"
                                        ;mov    ax,"\"
                                        ;mov    [es:di+2],ax
                                        ;MOV    word [ES:DI+curdir.text+2],"\"      ; NUL terminate
00007AF3 26C745025C00                   mov     word [ES:DI+curdir.text+2],005Ch ; 19/08/2018
                                        ;or     word [es:di+67],4000h
                                        ;or     byte [es:di+68],40h
00007AF9 26804D4440                     OR      byte [ES:DI+curdir.flags+1],(curdir_inuse>>8)
00007AFE 29C0                           sub     ax,ax
                                        ;MOV    [es:di+73],ax ; 0
00007B00 26894549                       MOV     [ES:DI+curdir.ID],ax
                                        ;mov    [es:di+75],ax ; 0
00007B04 2689454B                       MOV     [ES:DI+curdir.ID+2],ax
00007B08 B002                           mov     al,2
                                        ;mov    [es:di+79],aX ; 2
00007B0A 2689454F                       MOV     [ES:DI+curdir.end],ax
00007B0E 58                             POP     AX                      ; (al) = drive number
00007B0F 1E                             push    ds
00007B10 56                             push    si
00007B11 E8ACFF                         call    FIND_DPB
00007B14 7208                           JC      short icds5             ; OOOOPPPPPSSSS!!!!
                                        ;mov    [es:di+69],si
00007B16 26897545                       MOV     [ES:DI+curdir.devptr],SI
                                        ;mov    [es:di+71],ds
00007B1A 268C5D47                       MOV     [ES:DI+curdir.devptr+2],DS
                                    icds5:
00007B1E 5E                             pop     si
00007B1F 1F                             pop     ds
                                    icdsx:
00007B20 58                             pop     ax
                                    RET45:
00007B21 C3                             retn

                                    ;Break <$UserOper - get/set current user ID (for net)>
                                    ;--------------------------------------------------------------------------
                                    ;   $UserOper - retrieve or initiate a user id string.  MSDOS will only
                                    ;   maintain this string and do no verifications.
                                    ;
                                    ;   Inputs: AL has function type (0-get 1-set 2-printer-set 3-printer-get
                                    ;                                 4-printer-set-flags,5-printer-get-flags)
                                    ;           DS:DX is user string pointer (calls 1,2)
```

```
                                ;           ES:DI is user buffer (call 3)
                                ;           BX is assign index (calls 2,3,4,5)
                                ;           CX is user number (call 1)
                                ;           DX is flag word (call 4)
                                ;   Outputs:       If AL = 0 then the current user string is written to DS:DX
                                ;                  and user CX is set to the user number
                                ;           If AL = 3 then CX bytes have been put at input ES:DI
                                ;           If AL = 5 then DX is flag word
                                ;-----------------------------------------------------------------------------

                                    ; 13/03/2024 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMDOS.COM)
                                    ; 02/12/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
                                    ; 21/05/2019 - Retro DOS v4.0
                                _$UserOper:
                                    ; 05/08/2018 - Retro DOS v3.0
                                    ; MSDOS 6.0 (& MSDOS 3.3)
                                    ;PUSH   AX
                                    ;SUB    AL,1                ; quick dispatch on 0,1
                                    ;POP    AX
                                    ; 01/07/2024
00007B22 3C01                       cmp     al,1
00007B24 720E                       JB      short UserGet       ; return to user the string
00007B26 742B                       JZ      short UserSet       ; set the current user
00007B28 3C05                       CMP     AL,5                ; test for 2,3,4 or 5
00007B2A 763A                       JBE     short UserPrint             ; yep

                                ; 13/03/2024
                                %if 0
                                    ;mov    byte [ss:EXTERR_LOCUS],1
                                    MOV     byte [SS:EXTERR_LOCUS],errLOC_Unk ;smr;SS Override
                                                                    ; Extended Error Locus
                                %else
                                    ; 13/03/2024 (PCDOS 7.1 IBMDOS.COM)
                                    ;;;
00007B2C E8AE97                     call    set_exerr_locus_unk   ; Extended Error Locus
                                    ;;;
                                %endif
                                    ;error  error_invalid_function; not 0,1,2,3
                                    ;mov    al,1
00007B2F B001                       MOV     AL,error_invalid_function
                                useroper_error:
                                    ; 17/12/2022
                                    ; 02/12/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
00007B31 E9448B                     JMP     SYS_RET_ERR
                                    ;jmp    short ASS_ERR

                                UserGet:
                                ; Transfer MYNAME to DS:DX
                                ; Set Return CX to MYNUM
00007B34 1E                         PUSH    DS                  ; switch registers
00007B35 07                         POP     ES
00007B36 89D7                       MOV     DI,DX               ; destination
00007B38 368B0E[0E00]               MOV     CX,[SS:MYNUM]       ; Get number   ;smr;SS Override
00007B3D E83789                     call    Get_User_Stack
                                    ;mov    [si+4],cx
00007B40 894C04                     MOV     [SI+user_env.user_CX],CX ; Set number return
00007B43 16                         push    ss                  ; point to DOSDATA
00007B44 1F                         pop     ds
00007B45 BE[0503]                   MOV     SI,MYNAME           ; point source to user string
                                UserMove:
00007B48 B90F00                     MOV     CX,15
00007B4B F3A4                       REP     MOVSB               ; blam.
00007B4D 31C0                       XOR     AX,AX               ; 16th byte is 0
00007B4F AA                         STOSB
                                UserBye:
00007B50 E91B8B                     jmp     SYS_RET_OK          ; no errors here

                                UserSet:
                                ; Transfer DS:DX to MYNAME
                                ; CX to MYNUM
00007B53 36890E[0E00]               MOV     [SS:MYNUM],CX                   ;smr;SS Override
00007B58 89D6                       MOV     SI,DX               ; user space has source
00007B5A 16                         push    ss
00007B5B 07                         pop     es
00007B5C BF[0503]                   MOV     DI,MYNAME           ; point dest to user string
00007B5F 36FE06[0403]               INC     byte [SS:DIFFNAM]   ; signal change ;smr;SS Override
00007B64 EBE2                       JMP     short UserMove

                                UserPrint:

                                  ;IF NOT Installed
                                  ; transfer PRINTER_GETSET_STRING
                                  ;ELSE
00007B66 50                         PUSH    AX
                                    ;mov    ax,111Fh
                                    ;MOV    AX,(MultNET SHL 8) OR 31
00007B67 B81F11                     mov     ax,(MultNET<<8)|31
00007B6A CD2F                       int     2Fh    ; Multiplex - NETWORK REDIRECTOR - PRINTER SETUP
                                                   ; STACK: WORD function
                                                   ; Return: CF set on error, AX = error code
                                                   ; STACK unchanged
00007B6C 5A                         POP     DX                  ; Clean stack
                                    ;JNC    short OKPA
00007B6D 73E1                       jnc     short UserBye ; 21/05/2019
                                    ; 17/12/2022
00007B6F EBC0                       jmp     short useroper_error
                                    ; 02/12/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
                                    ;jnb     short OKPA
                                    ;jmp     short useroper_error

                                    ; 17/12/2022
                                ;OKPA:
                                ;   jmp     short UserBye

                                  ;ENDIF

                                ;Break      <GetVisDrv - return visible drive>
                                ;-----------------------------------------------------------------------------
                                ;   GetVisDrv - correctly map non-spliced inuse drives
                                ;
                                ;   Inputs: AL has drive identifier (0=default)
                                ;   Outputs:       Carry Set - invalid drive/macro
                                ;           Carry Clear - AL has physical drive (0=A)
                                ;               ThisCDS points to CDS
                                ;   Registers modified: AL
                                ;-----------------------------------------------------------------------------

                                    ; 21/05/2019 - Retro DOS v4.0
                                    ; DOSCODE:AA9Fh (MSDOS 6.21, MSDOS.SYS)

                                    ; 02/12/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
                                    ; DOSCODE:AA3Fh (MSDOS 5.0, MSDOS.SYS)

                                GetVisDrv:
```

```
                                    ; 05/08/2018 - Retro DOS v3.0
                                    ; IBMDOS.COM (MSDOS 3.3, 1987) - Offset 6839h
00007B71 E81900                     CALL    GETTHISDRV              ; get inuse drive
00007B74 72AB                       jc      short RET45
00007B76 1E                         push    ds
00007B77 56                         push    si
00007B78 36C536[A205]               LDS     SI,[SS:THISCDS]                 ;smr;SS Override
                                    ;test   word [si+67],2000h
                                    ; 17/12/2022
                                    ;test   byte [si+68],20h
00007B7D F6444420                   test    byte [SI+curdir.flags+1],(curdir_splice>>8)
                                    ;TEST   word [SI+curdir.flags],curdir_splice
00007B81 5E                         pop     si
00007B82 1F                         pop     ds
00007B83 749C                       jz      short RET45             ; if not spliced, return OK
                                    ; MSDOS 6.0
                                    ;mov    byte [ss:DrvErr],0Fh
00007B85 36C606[1006]0F             MOV     byte [SS:DrvErr],error_invalid_drive ;IFS. ;AN000;smr;SS Override
00007B8B F9                         STC                             ; signal error
00007B8C C3                         retn

                                ;Break <Getthisdrv - map a drive designator (0=def, 1=A...)>
                                ;--------------------------------------------------------------------------
                                ;   GetThisDrv - look through a set of macros and return the current drive and
                                ;   macro pointer
                                ;
                                ;   Inputs: AL has drive identifier (1=A, 0=default)
                                ;   Outputs:
                                ;           Carry Set - invalid drive/macro
                                ;           Carry Clear - AL has physical drive (0=A)
                                ;              ThisCDS points to macro
                                ;   Registers modified: AL
                                ;--------------------------------------------------------------------------

                                    ; 21/05/2019 - Retro DOS v4.0
                                    ; DOSCODE:AABCh (MSDOS 6.21, MSDOS.SYS)

                                    ; 02/12/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
                                    ; DOSCODE:AA5Ch (MSDOS 5.0, MSDOS.SYS)

                                    ; 13/03/2024 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMDOS.COM)
                                    ; PCDOS 7.1 IBMDOS.COM - DOSCODE:0BD48h

                                GETTHISDRV:
                                    ; 05/08/2018
                                    ; 12/07/2018 - Retro DOS v3.0
                                    ; IBMDOS.COM (MSDOS 3.3, 1987) - Offset 6850h
                                    ; MSDOS 3.3 (& MSDOS 6.0)
00007B8D 08C0                       OR      AL,AL                   ; are we using default drive?
00007B8F 7505                       JNZ     SHORT GTD10             ; no, go get the CDS pointers
00007B91 36A0[3603]                 MOV     AL,[SS:CURDRV]          ; get the current drive
                                    ;INC    ax                      ; Counteract next instruction
                                    ; 04/09/2018
                                    ;inc    al
                                    ; 07/12/2022
00007B95 40                         inc     ax
                                GTD10:
                                    ;DEC    AX
                                    ; 02/12/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
00007B96 48                         dec     ax                      ; 0 = A
                                    ;dec    al
00007B97 1E                         PUSH    DS                      ; save world
00007B98 56                         PUSH    SI

                                ; 13/03/2024
                                %if 0
                                    ;mov    byte [ss:EXTERR_LOCUS],2
                                    MOV     BYTE [SS:EXTERR_LOCUS],errLOC_Disk          ;smr;SS Override
                                    TEST    BYTE [SS:FSHARING],-1 ; Logical or Physical?;smr;SS Override
                                    JZ      SHORT GTD20             ; Logical
                                %else
                                    ; 13/03/2024 (PCDOS 7.1 IBMDOS.COM)
                                    ;;;
00007B99 E84A97                     call    set_exerr_locus_disk
00007B9C 36803E[7205]00             cmp     byte [ss:FSHARING],0  ; Logical or Physical?
00007BA2 7420                       jz      short GTD20             ; Logical
                                    ;;;
                                %endif
00007BA4 50                         PUSH    AX
00007BA5 06                         PUSH    ES
00007BA6 57                         PUSH    DI
00007BA7 36C706[A205][F304]         MOV     WORD [SS:THISCDS],DUMMYCDS ;smr;SS Override
                                    ;mov    [SS:THISCDS+2],CS ; MSDOS 3.3
00007BAE 368C16[A405]               MOV     [SS:THISCDS+2],SS ; MSDOS 6.0 ;ThisCDS = &DummyCDS;smr;
00007BB3 0441                       ADD     AL,'A'
00007BB5 E81DFF                     CALL    InitCDS                 ; InitCDS(c);
                                    ;test   word [es:di+67],4000h
                                    ; 17/12/2022
                                    ;test   byte [es:di+68],40h
00007BB8 26F6454440                 test    byte [ES:DI+curdir.flags+1],(curdir_inuse>>8)
                                    ;TEST   WORD [ES:DI+curdir.flags],curdir_inuse      ; Clears carry
00007BBD 5F                         POP     DI
00007BBE 07                         POP     ES
00007BBF 58                         POP     AX
00007BC0 740D                       JZ      SHORT GTD30             ; Not a physical drive.
00007BC2 EB15                       JMP     SHORT GTDX              ; carry clear
                                GTD20:
00007BC4 E81500                     CALL    GetCDSFromDrv
00007BC7 7206                       JC      SHORT GTD30    ; Unassigned CDS -> return error already set
                                    ;test   word [si+43h],4000h
                                    ; 17/12/2022
                                    ;test   byte [si+44h],40h
00007BC9 F6444440                   test    byte [SI+curdir.flags+1],(curdir_inuse>>8)
                                    ;TEST   WORD [SI+curdir.flags],curdir_inuse ; Clears Carry
00007BCD 750A                       JNZ     SHORT GTDX              ; carry clear
                                GTD30:
                                    ; 21/05/2019
                                    ; MSDOS 6.0
00007BCF B00F                       MOV     AL,error_invalid_drive; invalid FAT drive
00007BD1 36A2[1006]                 MOV     BYTE [ss:DrvErr],AL   ; save this for IOCTL

                                ; 13/03/2024
                                %if 0
                                    ; MSDOS 3.3 (.& MSDOS 6.0)
                                    MOV     BYTE [ss:EXTERR_LOCUS],errLOC_Unk
                                %else
                                    ; 13/03/2024 (PCDOS 7.1 IBMDOS.COM)
                                    ;;;
00007BD5 E80597                     call    set_exerr_locus_unk
                                    ;;;
                                %endif
00007BD8 F9                         STC
                                GTDX:
00007BD9 5E                         POP     SI                      ; restore world
```

```
45619 00007BDA 1F                       POP     DS
45620 00007BDB C3                       RETN
45621
45622                                 ;Break <GetCDSFromDrv - convert a drive number to a CDS pointer>
45623                                 ;--------------------------------------------------------------------------
45624                                 ;   GetCDSFromDrv - given a physical drive number, convert it to a CDS
45625                                 ;   pointer, returning an error if the drive number is greater than the
45626                                 ;   number of CDS's
45627                                 ;
45628                                 ;   Inputs: AL is physical unit # A=0...
45629                                 ;   Outputs:       Carry Set if Bad Drive
45630                                 ;           Carry Clear
45631                                 ;               DS:SI -> CDS
45632                                 ;               [THISCDS] = DS:SI
45633                                 ;   Registers modified: DS,SI
45634                                 ;--------------------------------------------------------------------------
45635
45636                                     ; 21/05/2019 - Retro DOS v4.0
45637                                 GetCDSFromDrv:
45638 00007BDC 363A06[4700]             CMP     AL,[SS:CDSCOUNT]      ; is this a valid designator;smr;SS Override
45639                                   ;JB     SHORT GetCDS   ; cf=1 ; yes, go get the macro
45640                                   ;STC                           ; signal error
45641                                   ;RETN                          ; bye
45642                                   ; 23/09/2023
45643 00007BE1 F5                       cmc     ; cf=1 <-> cf=0
45644 00007BE2 7217                     jc      short GetCDS_retn
45645                                 GetCDS:
45646                                   ; 23/09/2023
45647                                   ;PUSH   BX
45648 00007BE4 50                       PUSH    AX
45649 00007BE5 36C536[3C00]             LDS     SI,[SS:CDSADDR]               ; get pointer to table;smr;SS Override
45650                                   ;mov    bl,81 ; MSDOS 3.3
45651                                   ;mov    bl,88 ; MSDOS 6.0
45652                                   ; 23/09/2023
45653                                   ;MOV    BL,curdir.size        ; size in convenient spot
45654                                   ;MUL    BL                    ; get net offset
45655 00007BEA B458                     mov     ah,curdir.size
45656 00007BEC F6E4                     mul     ah
45657 00007BEE 01C6                     ADD     SI,AX ; *             ; convert to true pointer
45658 00007BF0 368936[A205]             MOV     [SS:THISCDS],SI             ; store convenient offset;smr;SS Override
45659 00007BF5 368C1E[A405]             MOV     [SS:THISCDS+2],DS     ; store convenient segment;smr;SS Override
45660 00007BFA 58                       POP     AX
45661                                   ; 23/09/2023
45662                                   ;POP    BX
45663                                   ; (cf must be 0 here) ; *
45664                                   ;CLC                          ; no error
45665                                 GetCDS_retn:
45666 00007BFB C3                       RETN                          ; bye!
45667
45668                                 ;==========================================================================
45669                                 ; MACRO2.ASM, MSDOS 6.0, 1991
45670                                 ;==========================================================================
45671                                 ; Retro    DOS v3.0 - 12/07/2018
45672                                 ; 22/05/2019 - Retro DOS v4.0
45673                                 ; 13/03/2024 - Retro DOS v5.0
45674
45675                                 ;BREAK <TransFCB - convert an FCB into a path, doing substitution>
45676                                 ;--------------------------------------------------------------------------
45677                                 ;   TransFCB - Copy an FCB from DS:DX into a reserved area doing all of the
45678                                 ;       gritty substitution.
45679                                 ;
45680                                 ;   Inputs:     DS:DX - pointer to FCB
45681                                 ;               ES:DI - point to destination
45682                                 ;   Outputs:    Carry Set - invalid path in final map
45683                                 ;               Carry Clear - FCB has been mapped into ES:DI
45684                                 ;                   Sattrib is set from possibly extended FCB
45685                                 ;                   ExtFCB set if extended FCB found
45686                                 ;   Registers modified: most
45687                                 ;--------------------------------------------------------------------------
45688
45689                                     ; 04/12/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
45690                                 TransFCB:
45691                                     ; 22/05/2019 - Retro DOS v4.0
45692                                     ; 12/07/2018 - Retro DOS v3.0
45693                                     ;LocalVar FCBTmp,16
45694                                     ;ENTER
45695 00007BFC 55                       push    bp
45696 00007BFD 89E5                     mov     bp,sp
45697                                   ;sub    sp,15   ; MSDOS 3.3
45698 00007BFF 83EC10                   sub     sp,16   ; MSDOS 6.0
45699 00007C02 16                       push    ss
45700 00007C03 07                       pop     es
45701 00007C04 06                       push    es
45702 00007C05 57                       push    di
45703                                   ;lea    di,[bp-15] ; MSDOS 3.3
45704                                   ;LEA    DI,FCBTmp
45705 00007C06 8D7EF0                   lea     di,[bp-16]            ; point to FCB temp area
45706 00007C09 36C606[6C05]00           mov     byte [SS:EXTFCB],0    ; no extended FCB found ;smr;SS Override
45707 00007C0F 36C606[6D05]00           mov     byte [SS:SATTRIB],0   ; default search attributes;smr;SS Override
45708 00007C15 E832A6                   call    GetExtended             ; get FCB, extended or not
45709                                   ; 06/12/2022
45710 00007C18 740D                     jz      short GetDrive        ; not an extended FCB, get drive
45711 00007C1A 8A44FF                   mov     AL,[SI-1]               ; get attributes
45712 00007C1D 36A2[6D05]               mov     [SS:SATTRIB],AL             ; store search attributes;smr;SS Override
45713 00007C21 36C606[6C05]FF           mov     byte [SS:EXTFCB],-1   ; signal extended FCB  ;smr;SS Override
45714                                 GetDrive:
45715 00007C27 AC                       lodsb                         ; get drive byte
45716 00007C28 E862FF                   call    GETTHISDRV
45717 00007C2B 722A                     jc      short BadPack
45718 00007C2D E86F03                   call    TextFromDrive           ; convert 0-based drive to text
45719
45720                                 ; Scan the source to see if there are any illegal chars
45721
45722                                     ;mov    bx,CharType           ; load lookup table
45723 00007C30 B90B00                   mov     cx,11
45724 00007C33 56                       push    si                    ; back over name, ext
45725                                 FCBScan:
45726 00007C34 AC                       lodsb                         ; get a byte
45727
45728                                     ; 09/08/2018
45729                                     ;;xlat  byte [es:bx]
45730                                     ;es     xlat
45731
45732                                     ; 22/05/2019 - Retro DOS v4.0
45733 00007C35 E897E1                   call    GetCharType           ; get flags
45734
45735                                     ;test   al,8
45736 00007C38 A808                     test    al,FFCB
45737 00007C3A 741B                     jz      short BadPack
45738                                 NextCh:
45739 00007C3C E2F6                     loop    FCBScan
45740 00007C3E 5E                       pop     si
45741 00007C3F 89FB                     mov     bx,di
45742 00007C41 E863AA                   call    PackName              ; crunch the path
```

```
45743 00007C44 5F                       pop     di                      ; get original destination
45744 00007C45 07                       pop     es
45745 00007C46 16                       push    ss                      ; get DS addressability
45746 00007C47 1F                       pop     ds
45747                                   ;lea    si,[bp-15] ; MSDOS 3.3
45748                                   ;LEA    SI,FCBTmp               ; point at new pathname
45749 00007C48 8D76F0                   lea     si,[bp-16]
45750 00007C4B 803F00                   cmp     byte [bx],0
45751 00007C4E 7407                     jz      short BadPack
45752 00007C50 55                       push    bp
45753 00007C51 E80E00                   call    TransPathSet            ; convert the path
45754 00007C54 5D                       pop     bp
45755 00007C55 7303                     jnc     short FCBRet            ; bye with transPath error code
45756                                 BadPack:
45757 00007C57 F9                       STC
45758                                   ;mov    al,3
45759 00007C58 B003                     MOV     AL,error_path_not_found
45760                                 FCBRet:
45761                                   ;LEAVE
45762 00007C5A 89EC                     mov     sp,bp
45763 00007C5C 5D                       pop     bp
45764                                 TransPath_retn:
45765 00007C5D C3                       retn
45766
45767                                 ; 12/07/2018 - Retro DOS v3.0
45768
45769                                 ;BREAK <TransPath - copy a path, do string sub and put in current dir>
45770                                 ;--------------------------------------------------------------------------
45771                                 ;
45772                                 ;   TransPath - copy a path from DS:SI to ES:DI, performing component string
45773                                 ;       substitution, insertion of current directory and fixing . and ..
45774                                 ;       entries. Perform splicing. Allow input string to match splice
45775                                 ;       exactly.
45776                                 ;
45777                                 ;   TransPathSet - Same as above except No splicing is performed if input path
45778                                 ;       matches splice.
45779                                 ;
45780                                 ;   TransPathNoSet - No splicing/local using is performed at all.
45781                                 ;
45782                                 ;   The following anomalous behaviour is required:
45783                                 ;
45784                                 ;       Drive letters on devices are ignored. (set up DummyCDS)
45785                                 ;       Paths on devices are ignored. (truncate to 0-length)
45786                                 ;       Raw net I/O sets ThisCDS => NULL.
45787                                 ;       fSharing => dummyCDS and no subst/splice. Only canonicalize.
45788                                 ;
45789                                 ;   Other behaviour:
45790                                 ;
45791                                 ;       ThisCDS set up.
45792                                 ;       FatRead done on local CDS.
45793                                 ;       ValidateCDS done on local CDS.
45794                                 ;
45795                                 ;   Brief flowchart:
45796                                 ;
45797                                 ;       if fSharing then
45798                                 ;           set up DummyCDS (ThisCDS)
45799                                 ;           canonicalize (sets cMeta)
45800                                 ;           splice
45801                                 ;           fatRead
45802                                 ;           return
45803                                 ;       if \\ or d:\\ lead then
45804                                 ;           set up null CDS (ThisCDS)
45805                                 ;           canonicalize (sets cMeta)
45806                                 ;           return
45807                                 ;       if device then
45808                                 ;           set up dummyCDS (ThisCDS)
45809                                 ;           canonicalize (sets cMeta)
45810                                 ;           return
45811                                 ;       if file then
45812                                 ;           getCDS (sets (ThisCDS) from name)
45813                                 ;           validateCDS (may reset current dir)
45814                                 ;           Copy current dir
45815                                 ;           canonicalize (set cMeta)
45816                                 ;           splice
45817                                 ;           generate correct CDS (ThisCDS)
45818                                 ;           if local then
45819                                 ;               fatread
45820                                 ;           return
45821                                 ;
45822                                 ;   Inputs:     DS:SI - point to ASCIZ string path
45823                                 ;               DI - point to buffer in DOSDATA
45824                                 ;   Outputs:    Carry Set - invalid path specification: too many .., bad
45825                                 ;                   syntax, etc. or user FAILed to I 24.
45826                                 ;               WFP_Start - points to beginning of buffer
45827                                 ;               Curr_Dir_End - points to end of current dir in path
45828                                 ;               DS - DOSDATA
45829                                 ;   Registers modified: most
45830                                 ;
45831                                 ;--------------------------------------------------------------------------
45832
45833                                 ; 22/05/2019
45834                                 ; 13/05/2019 - Retro DOS v4.0
45835                                 ; DOSCODE:AB99h (MSDOS 6.21, MSDOS.SYS)
45836
45837                                 ; 04/12/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
45838                                 ; DOSCODE:AB39h (MSDOS 5.0, MSDOS.SYS)
45839
45840                                     ; 13/03/2024 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMDOS.COM)
45841                                     ; PCDOS 7.1 IBMDOS.COM - DOSCODE:0BE1Dh
45842
45843                                 TransPath:
45844 00007C5E 30C0                     XOR     AL,AL
45845 00007C60 EB02                     JMP     SHORT SetSplice
45846                                 TransPathSet:
45847 00007C62 B0FF                     MOV     AL,-1
45848                                 SetSplice:
45849 00007C64 36A2[4C03]               MOV     [SS:NoSetDir],AL        ; NoSetDir = !fExact; ;smr;SS Override
45850 00007C68 B0FF                     MOV     AL,-1
45851                                 TransPathNoSet:
45852 00007C6A 36A2[7105]               MOV     [SS:FSPLICE],AL                 ; fSplice = TRUE;       ;smr;SS Override
45853 00007C6E 36C606[7A05]FF           MOV     byte [ss:CMETA],-1                                      ;smr;SS Override
45854 00007C74 36893E[B205]             MOV     [SS:WFP_START],DI                                       ;smr;SS Override
45855 00007C79 36C706[B605]FFFF         MOV     word [SS:CURR_DIR_END],-1 ; crack from start;smr;SS Override
45856 00007C80 16                       push    ss
45857 00007C81 07                       pop     es
45858                                   ;lea    bp,[di+134]
45859 00007C82 8DAD8600                 LEA     BP,[DI+TEMPLEN]         ; end of buffer
45860                                 ;
45861                                 ; if this is through the server dos call, fsharing is set. We set up a
45862                                 ; dummy cds and let the operation go.
45863                                 ;
45864                                     ;TEST   byte [SS:FSHARING],-1 ; if no sharing                   ;smr;SS Override
45865                                     ;JZ     short CheckUNC          ; skip to UNC check
45866                                     ; 13/03/2024 (PCDOS 7.1 IBMDOS.COM)
```

```
                                    ;;;
00007C86 36803E[7205]00             cmp     byte [ss:FSHARING],0
00007C8C 7435                       jz      short CheckUNC
                                    ;;;
                                ;
                                ; ES:DI point to buffer
                                ;
00007C8E E8F802                     CALL    DriveFromText           ; get drive and advance DS:SI
00007C91 E8F9FE                     call    GETTHISDRV              ; Set ThisCDS and convert to 0-based
00007C94 722A                       jc      short NoPath
00007C96 E80603                     CALL    TextFromDrive         ; drop in new
00007C99 8D5D01                     LEA     BX,[DI+1]               ; backup limit
00007C9C E83401                     CALL    Canonicalize            ; copy and canonicalize
00007C9F 72BC                       jc      short TransPath_retn  ; errors
                                ;
                                ; Perform splices for net guys.
                                ;
00007CA1 16                         push    ss
00007CA2 1F                         pop     ds
00007CA3 8B36[B205]                 MOV     SI,[WFP_START]                ; point to name
00007CA7 F606[7105]FF               TEST    byte [FSPLICE],-1
00007CAC 7403                       JZ      short NoServerSplice
00007CAE E82D02                     CALL    Splice
                                NoServerSplice:
00007CB1 16                         push    ss
00007CB2 1F                         pop     ds                      ; for FATREAD
00007CB3 C43E[A205]                 LES     DI,[THISCDS]          ; for fatread
00007CB7 E8289C                     call    ECritDisk
00007CBA E89EE8                     call    FATREAD_CDS
00007CBD E84F9C                     call    LCritDisk
                                NoPath:
                                    ;mov    al,3
00007CC0 B003                       MOV     AL,error_path_not_found ; Set up for possible bad path error
00007CC2 C3                         retn                            ; any errors are in Carry flag

                                ; Let the network decide if the name is for a spooled device. It will map
                                ; the name if so.

                                CheckUNC:
00007CC3 36C706[A205]FFFF           MOV     WORD [SS:THISCDS],-1  ; NULL thisCDS        ;smr;SS Override
                                    ;CallInstall NetSpoolCheck,MultNET,35
00007CCA B82311                     mov     ax,1123h
00007CCD CD2F                       int     2Fh     ; Multiplex - NETWORK REDIRECTOR - QUALIFY REMOTE FILENAME
                                                    ; DS:SI -> ASCIZ filename to canonicalize
                                                    ; ES:DI -> 128-byte buffer for qualified name
                                                    ; Return: CF set if not resolved
00007CCF 7329                       JNC     short UNCDone

                                ; At this point the name is either a UNC-style name (prefixed with two leading
                                ; \\s) or is a local file/device. Remember that if a net-spooled device was
                                ; input, then the name has been changed to the remote spooler by the above net
                                ; call. Also, there may be a drive in front of the \\.

                                NO_CHECK:
00007CD1 E8B502                     CALL    DriveFromText         ; eat drive letter
00007CD4 50                         PUSH    AX                      ; save it
00007CD5 8B04                       MOV     AX,[SI]                     ; get first two bytes of path
00007CD7 E810E1                     call    PATHCHRCMP              ; convert to normal form
00007CDA 86E0                       XCHG    AH,AL                   ; swap for second byte
00007CDC E80BE1                     call    PATHCHRCMP              ; convert to normal form
00007CDF 751F                       JNZ     short CheckDevice     ; not a path char
00007CE1 38C4                       CMP     AH,AL                   ; are they same?
00007CE3 751B                       JNZ     short CheckDevice     ; nope

                                ; We have a UNC request. We must copy the string up to the beginning of the
                                ; local machine root path

00007CE5 58                         POP     AX
00007CE6 A5                         MOVSW                           ; get the lead \\.
                                UNCCpy:
00007CE7 AC                         LODSB                           ; get a byte
00007CE8 E8ACE0                     call    UCase                   ;AN000;; convert the char
00007CEB 08C0                       OR      AL,AL
00007CED 740E                       JZ      short UNCTerm         ; end of string. All done.
00007CEF E8F8E0                     call    PATHCHRCMP              ; is it a path char?
00007CF2 89FB                       MOV     BX,DI                   ; backup position
00007CF4 AA                         STOSB
00007CF5 75F0                       JNZ     short UNCCpy          ; no, go copy
00007CF7 E8D900                     CALL    Canonicalize            ; wham (and set cMeta)
                                UNCDone:
00007CFA 16                         push    ss
00007CFB 1F                         pop     ds
00007CFC C3                         retn                            ; return error code
                                UNCTerm:
00007CFD AA                         STOSB                           ;AN000;
00007CFE EBFA                       JMP     short UNCDone         ;AN000;

                                CheckDevice:

                                ; Check DS:SI for device. First eat any path stuff

00007D00 58                         POP     AX                      ; retrieve drive info
00007D01 803C00                     CMP     BYTE [SI],0           ; check for null file
00007D04 7504                       JNZ     short CheckPath
                                    ;mov    al,2
00007D06 B002                       MOV     AL,error_file_not_found ; bad file error
00007D08 F9                         STC                             ; signal error on null input
00007D09 C3                         RETN                            ; bye!
                                CheckPath:
00007D0A 50                         push    ax
00007D0B 55                         push    bp                      ; save drive number

                                ; 04/12/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
                                %if 0
                                    ; MSDOS 6.0
                                ;;;BUGBUG BUG 10-26-1992 scottq
                                ;;;This is a hack for the CDROM extensions (2.1) who scan looking
                                ;;;for the following POP BP == 5Dh (restore <bp,ax>).
                                ;;;The problem is that a direct call to CheckThisDevice can (and did)
                                ;;;end up having a 5D in the opcode's displacement field. The
                                ;;;scanning code would choke on this thinking it was a POP BP instruction.
                                ;;;
                                ;;;What we do here is do a call to a function that is less than 5Dh
                                ;;;bytes away (and assert its not exactly 5D away) that jmps (transfers)
                                ;;;to the correct function. This cannot accidently insert a 5Dh.
                                ;;;
                                ;;;More info:
                                ;;;  This particular scan is begun at the UNCdone label for 32 bytes
                                ;;;looking for pop BP, so you cannot put a 5D between here and there.
                                ;;;
                                    call    no5Dshere
                                start5Dhack:
                                ;following is replaced with 5Dhack code--Invoke CheckThisDevice
                                backfrom5Dhack:
```

```
                                %endif

                                ; 13/03/2024
                                ; PCDOS 7.1 IBMDOS.COM - DOSCODE:BECBh
                                ; (MSDOS 6.22 MSDOS.SYS - DOSCODE:AC47h)
                                ; ((Windows ME IO.SYS - BIOSCODE:A6C2h))
                                %if 0
                                    call    no5Dshere
                                %else
                                ; 13/03/2024 - Retro DOS v5.0

                                    ; 04/12/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
                                    ; Note: 'call no5Dshere' is not required for MSDOS 5.0 MSDOS.SYS
00007D0C E81ED2                     call    CheckThisDevice       ; E8h,6Fh,0D6h
                                %endif
00007D0F 5D                         pop     bp
00007D10 58                         pop     ax                    ; get drive letter back
00007D11 731C                       JNC     short DoFile          ; yes we have a file.

                                ; We have a device. AX has drive letter. At this point we may fake a CDS ala
                                ; sharing DOS call. We know by getting here that we are NOT in a sharing DOS
                                ; call.

00007D13 36C606[7205]FF             MOV     byte [SS:FSHARING],-1 ; simulate sharing dos call;smr;SS Override
00007D19 E871FE                     call    GETTHISDRV              ; set ThisCDS and init DUMMYCDS
00007D1C 36C606[7205]00             MOV     byte [SS:FSHARING],0  ;                       ;smr;SS Override

                                ; Now that we have noted that we have a device, we put it into a form that
                                ; getpath can understand. Normally getpath requires d:\ to begin the input
                                ; string. We relax this to state that if the d:\ is present then the path
                                ; may be a file. If D:/ (note the forward slash) is present then we have
                                ; a device.

00007D22 E87A02                     CALL    TextFromDrive
00007D25 B02F                       MOV     AL,'/'                ; path sep.
00007D27 AA                         STOSB
00007D28 E8869A                     call    StrCpy                ; move remainder of string

00007D2B F8                         CLC                             ; everything OK.
00007D2C 16                         push    ss
00007D2D 1F                         pop     ds                    ; remainder of OK stuff
                                DoFile_retn:
00007D2E C3                         retn

                                ; 13/03/2024 - Retro DOS v5.0
                                ; PCDOS 7.1 IBMDOS.COM - DOSCODE:BEEEh
                                ; (MSDOS 6.22 MSDOS.SYS - DOSCODE:AC6Ah)
                                ; ((Windows ME IO.SYS - BIOSCODE:A6F2h))
                                ;%if 1

                                ; 04/12/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
                                %if 0

                                no5Dshere:
                                    ; 10/08/2018
                                    jmp     CheckThisDevice           ; snoop for device
                                %endif

                                ;.erre (no5Dshere - start5Dhack - 5D)

                                ; We have a file. Get the raw CDS.

                                DoFile:
                                    ; MSDOS 3.3 (& MSDOS 6.0)

00007D2F E83FFE                     call    GetVisDrv             ; get proper CDS
                                    ;mov    al,3
00007D32 B003                       MOV     AL,error_path_not_found ; Set up for possible bad file error
00007D34 72F8                       jc      short DoFile_retn  ; CARRY set -> bogus drive/spliced

                                ; ThisCDS has correct CDS. DS:SI advanced to point to beginning of path/file.
                                ; Make sure that CDS has valid directory; ValidateCDS requires a temp buffer
                                ; Use the one that we are going to use (ES:DI).

                                    ;SAVE    <DS,SI,ES,DI>        ; save all string pointers.
00007D36 1E                         push    ds
00007D37 56                         push    si
00007D38 06                         push    es
00007D39 57                         push    di
00007D3A E823D1                     call    ValidateCDS           ; poke CDS and make everything OK
                                    ;RESTORE <DI,ES,SI,DS>        ; get back pointers
00007D3D 5F                         pop     di
00007D3E 07                         pop     es
00007D3F 5E                         pop     si
00007D40 1F                         pop     ds
                                    ;mov    al,3
00007D41 B003                       MOV     AL,error_path_not_found ; Set up for possible bad path error
                                    ;retc                         ; someone failed an operation
00007D43 72E9                       jc      short DoFile_retn

                                ; ThisCDS points to correct CDS. It contains the correct text of the
                                ; current directory. Copy it in.

00007D45 1E                         push    ds
00007D46 56                         push    si
00007D47 36C536[A205]               LDS     SI,[SS:THISCDS]             ; point to CDS ;smr;SS Override
00007D4C 89FB                       MOV     BX,DI                 ; point to destination
                                    ;add    bx,[si+79] ; MSDOS 6.0
00007D4E 035C4F                     ADD     BX,[SI+curdir.end]    ; point to backup limit
                                    ;lea    bp,[di+134]
00007D51 8DAD8600                   LEA     BP,[DI+TEMPLEN]         ; regenerate end of buffer
                                                                  ;AN000;
00007D55 E8689A                     call    FStrCpy                 ; copy string. ES:DI point to end
00007D58 4F                         DEC     DI                      ; point to NUL byte

                                ; Make sure that there is a path char at end.

00007D59 B05C                       MOV     AL,'\'
00007D5B 263845FF                   CMP     [ES:DI-1],AL
00007D5F 7401                       JZ      short GetOrig
00007D61 AA                         STOSB

                                ; Now get original string.

                                GetOrig:
00007D62 4F                         DEC     DI                      ; point to path char
00007D63 5E                         pop     si
00007D64 1F                         pop     ds

                                ; BX points to the end of the root part of the CDS (at where a path char
                                ; should be). Now, we decide whether we use this root or extend it with the
                                ; current directory. See if the input string begins with a leading
00007D65 E8CE00                     CALL    PathSep                 ; is DS:SI a path sep?
```

```
00007D68 7511                   JNZ     short PathAssure      ; no, DI is correct. Assure a path char
00007D6A 08C0                   OR      AL,AL                   ; end of string?
00007D6C 7410                   JZ      short DoCanon         ; yes, skip.
                        ;
                        ; The string does begin with a \. Reset the beginning of the canonicalization
                        ; to this root. Make sure that there is a path char there and advance the
                        ; source string over all leading \'s.
                        ;
00007D6E 89DF                     MOV     DI,BX                   ; back up to root point.
                        SkipPath:
00007D70 AC                     LODSB
00007D71 E876E0                 call    PATHCHRCMP
00007D74 74FA                   JZ      short SkipPath
00007D76 4E                     DEC     SI
00007D77 08C0                   OR      AL,AL
00007D79 7403                   JZ      short DoCanon

                        ; DS:SI start at some file name. ES:DI points at some path char. Drop one in
                        ; for yucks.

                        PathAssure:
00007D7B B05C                   MOV     AL,'\' ; 5Ch
00007D7D AA                     STOSB

                        ; ES:DI point to the correct spot for canonicalization to begin.
                        ; BP is the max extent to advance DI
                        ; BX is the backup limit for ..

                        DoCanon:
00007D7E E85200                 CALL    Canonicalize            ; wham.
                                ;retc                         ; badly formatted path.
00007D81 72AB                   jc      short DoFile_retn

                        ; The string has been moved to ES:DI. Reset world to DOS context, pointers
                        ; to wfp_start and do string substitution. BP is still the max position in
                        ; buffer.

00007D83 16                     push    ss
00007D84 1F                     pop     ds
00007D85 8B3E[B205]             MOV     DI,[WFP_START]        ; DS:SI point to string
00007D89 C536[A205]             LDS     SI,[THISCDS]          ; point to CDS
00007D8D E81702                 CALL    PathPref                ; is there a prefix?
00007D90 7514                   JNZ     short DoSplice        ; no, do splice

                        ; We have a match. Check to see if we ended in a path char.

00007D92 8A44FF                 MOV     AL,[SI-1]             ; last char to match
00007D95 E852E0                 call    PATHCHRCMP              ; did we end on a path char? (root)
00007D98 740C                   JZ      short DoSplice        ; yes, no current dir here.
                        Pathline:                                       ; 2/13/KK
00007D9A 26803D00               CMP     BYTE [ES:DI],0        ; end at NUL?
00007D9E 7406                   JZ      short DoSplice
00007DA0 47                     INC     DI                      ; point to after current path char
00007DA1 36893E[B605]           MOV     [SS:CURR_DIR_END],DI  ; point to correct spot ;smr;SS Override

                        ; Splice the result.

                        DoSplice:
00007DA6 16                     push    ss
00007DA7 1F                     pop     ds                    ; back to DOSDATA
00007DA8 8B36[B205]             MOV     SI,[WFP_START]        ; point to beginning of string
00007DAC 31C9                   XOR     CX,CX
00007DAE F606[7105]FF           TEST    byte [FSPLICE],-1
00007DB3 7403                   JZ      short SkipSplice
00007DB5 E82601                 CALL    Splice                  ; replaces in place.
                        SkipSplice:

                        ; The final thing is to assure ourselves that a FATREAD is done on the local
                        ; device.

00007DB8 16                     push    ss
00007DB9 1F                     pop     ds
00007DBA C43E[A205]             LES     DI,[THISCDS]          ; point to correct drive
                                ;test   word [es:di+67],8000h
                                ; 17/12/2022
                                ;test   byte [es:di+68],80h
00007DBE 26F6454480             test    byte [ES:DI+curdir.flags+1],curdir_isnet>>8 ; 04/12/2022
                                ;TEST   word [ES:DI+curdir.flags],curdir_isnet ; 8000h
00007DC3 750D                   JNZ     short Done            ; net, no fatread necessary (retnz)
00007DC5 E30B                   JCXZ    Done
00007DC7 E8189B                 call    ECritDisk
00007DCA E88EE7                 call    FATREAD_CDS
00007DCD E83F9B                 call    LCritDisk
                                ;mov    al, 3
00007DD0 B003                   MOV     AL,error_path_not_found ; Set up for possible bad path error
                        Done:
00007DD2 C3                     retn                          ; any errors in carry flag.

                        ; 13/07/2018

                        ;BREAK <Canonicalize - copy a path and remove . and .. entries>
                        ;--------------------------------------------------------------------------
                        ;   Canonicalize - copy path removing . and .. entries.
                        ;
                        ;   Inputs:     DS:SI - point to ASCIZ string path
                        ;               ES:DI - point to buffer
                        ;               BX - backup limit (offset from ES) points to slash
                        ;               BP - end of buffer
                        ;   Outputs:    Carry Set - invalid path specification: too many .., bad
                        ;                   syntax, etc.
                        ;               Carry Clear -
                        ;                   DS:DI - advanced to end of string
                        ;                   ES:DI - advanced to end of canonicalized form after nul
                        ;   Registers modified: AX CX DX (in addition to those above)
                        ;--------------------------------------------------------------------------

                            ; 04/12/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)

                            ; 13/03/2024 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMDOS.COM)

                        Canonicalize:

                        ; We copy all leading path separators.

00007DD3 AC                     LODSB                           ;   while (PathChr (*s))
00007DD4 E813E0                 call    PATHCHRCMP
00007DD7 7507                   JNZ     short CanonDec
00007DD9 39EF                   CMP     DI,BP                   ;       if (d > dlim)
00007DDB 7319                   JAE     short CanonBad        ;         goto error;
00007DDD AA                     STOSB
00007DDE EBF3                   JMP     short Canonicalize    ;         *d++ = *s++;
                        CanonDec:
00007DE0 4E                     DEC     SI
```

```
                                ; Main canonicalization loop. We come here with DS:SI pointing to a textual
                                ; component (no leading path separators) and ES:DI being the destination
                                ; buffer.

                                CanonLoop:

                                ; If we are at the end of the source string, then we need to check to see that
                                ; a potential drive specifier is correctly terminated with a path sep char.
                                ; Otherwise, do nothing

00007DE1 31C0                       XOR     AX,AX
00007DE3 3804                       CMP     [SI],AL                 ;       if (*s == 0) {
00007DE5 751A                       JNZ     short DoComponent
00007DE7 26807DFF3A                 CMP     BYTE [ES:DI-1],':'      ;         if (d[-1] == ':')
00007DEC 7505                       JNZ     short DoTerminate
00007DEE B05C                       MOV     AL,'\'                  ;               *d++ = '\';
00007DF0 AA                         STOSB
00007DF1 88E0                       MOV     AL,AH
                                DoTerminate:
00007DF3 AA                         STOSB                           ;           *d++ = 0;
00007DF4 F8                         CLC                             ;           return (0);
00007DF5 C3                         retn

                                CanonBad:
00007DF6 E8C701                     CALL    ScanPathChar            ; check for path chars in rest of string
                                    ;mov    al,3
00007DF9 B003                       MOV     AL,error_path_not_found ; Set up for bad path error
00007DFB 7402                       JZ      short PathEnc           ; path character encountered in string
                                    ;mov    al,2
00007DFD B002                       MOV     AL,error_file_not_found ; Set bad file error
                                PathEnc:
00007DFF F9                         STC
                                CanonBad_retn:
00007E00 C3                         retn

                                ; We have a textual component that we must copy. We uppercase it and truncate
                                ; it to 8.3

                                DoComponent:                            ;           }
00007E01 E84C00                     CALL    CopyComponent           ;       if (!CopyComponent (s, d))
00007E04 72FA                       jc      short CanonBad_retn     ;           return (-1);

                                ; We special case the . and .. cases. These will be backed up.


                                    ;CMP    WORD PTR ES:[DI],'.' + (0 SHL 8)
00007E06 26833D2E                   CMP     WORD [ES:DI],002Eh
00007E0A 7408                       JZ      short Skip1
                                    ;CMP    WORD PTR ES:[DI],'..'
00007E0C 26813D2E2E                 CMP     WORD [ES:DI],2E2Eh
00007E11 7508                       JNZ     short CanonNormal
00007E13 4F                         DEC     DI                      ;           d--;
                                Skip1:
00007E14 E82800                     CALL    SkipBack                ;           SkipBack ();
                                    ;;mov   al,3
                                    ; 07/07/2024 (*)
                                    ;MOV    AL,error_path_not_found ; Set up for possible bad path error
00007E17 72E7                       jc      short CanonBad_retn ; AL=3 (*)
00007E19 EB02                       JMP     short CanonPath         ;           }

                                ; We have a normal path. Advance destination pointer over it.

                                CanonNormal:                            ;       else
00007E1B 01CF                       ADD     DI,CX                   ;           d += ct;

                                ; We have successfully copied a component. We are now pointing at a path
                                ; sep char or are pointing at a nul or are pointing at something else.
                                ; If we point at something else, then we have an error.

                                CanonPath:
00007E1D E81600                     CALL    PathSep
00007E20 75D4                       JNZ     short CanonBad        ; something else...

                                ; Copy the first path char we see.

00007E22 AC                         LODSB                           ; get the char
00007E23 E8C4DF                     call    PATHCHRCMP              ; is it path char?
00007E26 75B8                       JNZ     short CanonDec        ; no, go test for nul
00007E28 39EF                       CMP     DI,BP                   ; beyond buffer end?
00007E2A 73CA                       JAE     short CanonBad        ; yep, error.
00007E2C AA                         STOSB                           ; copy the one byte

                                ; Skip all remaining path chars

                                CanonPathLoop:
00007E2D AC                         LODSB                           ; get next byte
00007E2E E8B9DF                     call    PATHCHRCMP              ; path char again?
00007E31 74FA                       JZ      short CanonPathLoop   ; yep, grab another
00007E33 4E                         DEC     SI                      ; back up
00007E34 EBAB                       JMP     short  CanonLoop      ; go copy component

                                ;BREAK <PathSep - determine if char is a path separator>
                                ;--------------------------------------------------------------------------
                                ;   PathSep - look at DS:SI and see if char is / \ or NUL
                                ;   Inputs:     DS:SI - point to a char
                                ;   Outputs:    AL has char from DS:SI (/ => \)
                                ;               Zero set if AL is / \ or NUL
                                ;               Zero reset otherwise
                                ;   Registers modified: AL
                                ;--------------------------------------------------------------------------

                                PathSep:
00007E36 8A04                       MOV     AL,[SI]                 ; get the character
                                PathSepGotCh:                           ; already have character
00007E38 08C0                       OR      AL,AL                   ; test for zero
00007E3A 74C4                       jz      short CanonBad_retn   ; return if equal to zero (NUL)
                                    ;call   PATHCHRCMP              ; check for path character
                                    ;retn                           ; and return HIS determination
                                    ; 18/12/2022
00007E3C E9ABDF                     jmp     PATHCHRCMP


                                ;BREAK <SkipBack - move backwards to a path separator>
                                ;--------------------------------------------------------------------------
                                ;   SkipBack - look at ES:DI and backup until it points to a / ;   Inputs:     ES:DI - point to a char
                                ;               BX has current directory back up limit (point to a / \)
                                ;   Outputs:    ES:DI backed up to point to a path char
                                ;               AL has char from output ES:DI (path sep if carry clear)
                                ;               Carry set if illegal backup
                                ;               Carry Clear if ok
                                ;   Registers modified: DI,AL
                                ;--------------------------------------------------------------------------

                                SkipBack:
00007E3F 39DF                       CMP     DI,BX                   ;   while (TRUE) {
```

```
00007E41 720A                           JB      short SkipBad           ;       if (d < dlim)
00007E43 4F                             DEC     DI                      ;           goto err;
00007E44 268A05                         MOV     AL,[ES:DI]              ;       if (pathchr (*--d))
00007E47 E8A0DF                         call    PATHCHRCMP              ;           break;
00007E4A 75F3                           JNZ     short SkipBack          ;       }
                                        ;CLC                            ;   return (0);
                                        ; 01/07/2024
                                        ; cf=0
00007E4C C3                             retn                            ;
                                SkipBad:                                    ;err:
                                        ;mov    al,3
00007E4D B003                           MOV     AL,error_path_not_found ; bad path error
                                        ;STC                            ;   return (-1);
                                        ; 01/07/2024
                                        ; cf=1
00007E4F C3                             retn                            ;

                                ;Break <CopyComponent - copy out a file path component>
                                ;--------------------------------------------------------------------------
                                ;   CopyComponent - copy a file component from a path string (DS:SI) into ES:DI
                                ;
                                ;   Inputs:     DS:SI - source path
                                ;               ES:DI - destination
                                ;               ES:BP - end of buffer
                                ;   Outputs:    Carry Set - too long
                                ;               Carry Clear - DS:SI moved past component
                                ;                   CX has length of destination
                                ;   Registers modified: AX,CX,DX
                                ;--------------------------------------------------------------------------

                                    ; 04/12/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)

                                    ; 13/03/2024 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMDOS.COM)

                                CopyComponent:

                                %define CopyBP      [BP]            ; word
                                %define CopyD       [BP+2]          ; dword
                                %define CopyDoff [BP+2]             ; word
                                %define CopyS       [BP+6]          ; dword
                                %define CopySoff [BP+6]             ; word
                                %define CopyTemp [BP+10]  ; byte

00007E50 83EC0E                         SUB     SP,14                   ; room for temp buffer
00007E53 1E                             push    ds
00007E54 56                             push    si
00007E55 06                             push    es
00007E56 57                             push    di
00007E57 55                             push    bp
00007E58 89E5                           MOV     BP,SP
00007E5A B42E                           MOV     AH,'.'
00007E5C AC                             LODSB
00007E5D AA                             STOSB
00007E5E 38E0                           CMP     AL,AH                   ;   if ((*d++=*s++) == '.') {
00007E60 7518                           JNZ     short NormalComp
00007E62 E8D1FF                         CALL    PathSep                 ;       if (!pathsep(*s))
00007E65 740B                           JZ      short NulTerm
                                TryTwoDot:
00007E67 AC                             LODSB                           ;           if ((*d++=*s++) != '.'
00007E68 AA                             STOSB
00007E69 38E0                           CMP     AL,AH
00007E6B 7557                           JNZ     short CopyBad
00007E6D E8C6FF                         CALL    PathSep
00007E70 7552                           JNZ     short CopyBad           ;               || !pathsep (*s))
                                NulTerm:                                    ;                   return -1;
00007E72 30C0                           XOR     AL,AL                   ;       *d++ = 0;
00007E74 AA                             STOSB
00007E75 897606                         MOV     CopySoff,SI
00007E78 EB47                           JMP     SHORT _GoodRet          ;       }
                                NormalComp:                                 ;   else {
00007E7A 8B7606                         MOV     SI,CopySoff ; [bp+6]
00007E7D E8AEDE                         call    NameTrans               ;       s = NameTrans (s, Name1);
00007E80 3B7606                         CMP     SI,CopySoff             ;       if (s == CopySOff)
00007E83 743F                           JZ      short CopyBad           ;           return (-1);
00007E85 36F606[7205]FF                 TEST    byte [SS:FSHARING],-1   ;       if (!fSharing) {;smr;SS Override
00007E8B 7510                           JNZ     short DoPack
00007E8D 80E201                         AND     DL,1                    ;           cMeta += fMeta;
00007E90 3600160[7A05]                  ADD     [ss:CMETA],DL           ;         if (cMeta > 0);smr;SS Override
00007E95 7F2D                           JG      short CopyBad           ;             return (-1);
00007E97 7504                           JNZ     short DoPack            ;         else
00007E99 08D2                           OR      DL,DL                   ;           if (cMeta == 0 && fMeta == 0)
00007E9B 742F                           JZ      short CopyBadPath       ;             return (-1);
                                DoPack:                                     ;           }
00007E9D 897606                         MOV     CopySoff,SI ; [bp+6]
00007EA0 16                             push    ss
00007EA1 1F                             pop     ds
00007EA2 BE[4B05]                       MOV     SI,NAME1
00007EA5 8D7E0A                         LEA     DI,CopyTemp ; [bp+10]
00007EA8 57                             push    di
00007EA9 E8FBA7                         call    PackName                ;       PackName (Name1, temp);
00007EAC 5F                             pop     di
00007EAD E81999                         call    StrLen                  ;       if (strlen(temp)+d > bp)
00007EB0 49                             DEC     CX
00007EB1 034E02                         ADD     CX,CopyDoff ; [bp+2]
                                        ; 04/12/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
                                        ;cmp    cx,[bp+0]
                                        ; 15/12/2022
                                        ;cmp    cx,[bp]
00007EB4 3B4E00                         CMP     CX,CopyBP   ; [bp+0]
00007EB7 730B                           JAE     short CopyBad           ;           return (-1);
00007EB9 89FE                           MOV     SI,DI                   ;       strcpy (d, temp);
00007EBB C47E02                         LES     DI,CopyD    ; [bp+2]
00007EBE E8FF98                         call    FStrCpy
                                _GoodRet:                               ;       }
00007EC1 F8                             CLC
00007EC2 EB0B                           JMP     SHORT CopyEnd           ;   return 0;
                                CopyBad:
00007EC4 F9                             STC
00007EC5 E8F800                         CALL    ScanPathChar            ; check for path chars in rest of string
                                        ;mov    al,2
00007EC8 B002                           MOV     AL,error_file_not_found ; Set up for bad file error
00007ECA 7503                           JNZ     short CopyEnd
                                CopyBadPath:
00007ECC F9                             STC
                                        ;mov    al,3
00007ECD B003                           MOV     AL,error_path_not_found ; Set bad path error
                                CopyEnd:
00007ECF 5D                             pop     bp
00007ED0 5F                             pop     di
00007ED1 07                             pop     es
00007ED2 5E                             pop     si
00007ED3 1F                             pop     ds
00007ED4 9F                             LAHF
00007ED5 83C40E                         ADD     SP,14                   ; reclaim temp buffer
```

```
46489 00007ED8 E8EE98                   call    StrLen
46490 00007EDB 49                       DEC     CX
46491 00007EDC 9E                       SAHF
46492 00007EDD C3                       retn
46493
46494                                ; 14/05/2019 - Retro DOS v4.0
46495                                ; DOSCODE:AE22h (MSDOS 6.21, MSDOS.SYS)
46496
46497                                ; 04/12/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
46498                                ; DOSCODE:ADBFh (MSDOS 5.0, MSDOS.SYS)
46499
46500                                ;Break <Splice - pseudo mount by string substitution>
46501                                ;--------------------------------------------------------------------------
46502                                ;   Splice - take a string and substitute a prefix if one exists. Change
46503                                ;       ThisCDS to point to physical drive CDS.
46504                                ;   Inputs:     DS:SI point to string
46505                                ;               NoSetDir = TRUE => exact matches with splice fail
46506                                ;   Outputs:    DS:SI points to thisCDS
46507                                ;               ES:DI points to DPB
46508                                ;               String at DS:SI may be reduced in length by removing prefix
46509                                ;               and substituting drive letter.
46510                                ;               CX = 0 If no splice done
46511                                ;               CX <> 0 otherwise
46512                                ;               ThisCDS points to proper CDS if spliced, otherwise it is
46513                                ;                   left alone
46514                                ;               ThisDPB points to proper DPB
46515                                ;   Registers modified: DS:SI, ES:DI, BX,AX,CX
46516                                ;--------------------------------------------------------------------------
46517
46518                                    ; 13/03/2024 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMDOS.COM)
46519                                    ; PCDOS 7.1 IBMDOS.COM - DOSCODE:0C0A6h
46520
46521                                Splice:
46522 00007EDE 36F606[5A00]FF           TEST    byte [SS:SPLICES],-1  ;smr;SS Override
46523 00007EE4 7469                     JZ      short AllDone
46524 00007EE6 36FF36[A205]             push    word [SS:THISCDS]
46525 00007EEB 36FF36[A405]             push    word [SS:THISCDS+2]   ; TmpCDS = ThisCDS;smr;SS Override
46526 00007EF0 1E                       push    ds
46527 00007EF1 56                       push    si
46528 00007EF2 5F                       pop     di
46529 00007EF3 07                       pop     es
46530 00007EF4 31C0                     XOR     AX,AX                 ;   for (i=1; s = GetCDSFromDrv (i); i++)
46531                                SpliceScan:
46532 00007EF6 E8E3FC                   call    GetCDSFromDrv
46533 00007EF9 724A                     JC      short SpliceDone
46534 00007EFB FEC0                     INC     AL
46535                                   ; 17/12/2022
46536                                   ;test   byte [si+68],20h
46537 00007EFD F6444420                 test    byte [si+curdir.flags+1],curdir_splice>>8 ; 04/12/2022
46538                                   ;;test  word [si+67],2000h
46539                                   ;TEST   word [SI+curdir.flags],curdir_splice
46540 00007F01 74F3                     JZ      short SpliceScan      ;       if ( Spliced (i) ) {
46541 00007F03 57                       push    di
46542 00007F04 E8A000                   CALL    PathPref              ;           if (!PathPref (s, d))
46543 00007F07 7403                     JZ      short SpliceFound     ;
46544                                SpliceSkip:
46545 00007F09 5F                       pop     di
46546 00007F0A EBEA                     JMP     short SpliceScan      ;               continue;
46547                                SpliceFound:
46548 00007F0C 26803D00                 CMP     BYTE [ES:DI],0        ;           if (*s || NoSetDir) {
46549 00007F10 7508                     JNZ     short SpliceDo
46550 00007F12 36F606[4C03]FF           TEST    byte [ss:NoSetDir],-1               ;smr;SS Override
46551 00007F18 75EF                     JNZ     short SpliceSkip
46552                                SpliceDo:
46553 00007F1A 89FE                     MOV     SI,DI                 ;               p = src + strlen (p);
46554 00007F1C 06                       push    es
46555 00007F1D 1F                       pop     ds
46556 00007F1E 5F                       pop     di
46557 00007F1F E87F00                   CALL    TextFromDrive1        ;               src = TextFromDrive1(src,i);
46558 00007F22 36A1[B605]               MOV     AX,[SS:CURR_DIR_END]                ;smr;SS Override
46559 00007F26 09C0                     OR      AX,AX
46560 00007F28 7808                     JS      short NoPoke
46561 00007F2A 01F8                     ADD     AX,DI                 ;               curdirend += src-p;
46562 00007F2C 29F0                     SUB     AX,SI
46563 00007F2E 36A3[B605]               MOV     [SS:CURR_DIR_END],AX                ;smr;SS Override
46564                                NoPoke:
46565 00007F32 803C00                   CMP     BYTE [SI],0           ;               if (*p)
46566 00007F35 7503                     JNZ     short SpliceCopy      ;                   *src++ = '\\';
46567 00007F37 B05C                     MOV     AL,"\"
46568 00007F39 AA                       STOSB
46569                                SpliceCopy:                               ;               strcpy (src, p);
46570 00007F3A E88398                   call    FStrCpy
46571 00007F3D 83C404                   ADD     SP,4                  ; throw away saved stuff
46572 00007F40 80C901                   OR      CL,1                  ; signal splice done.
46573 00007F43 EB0C                     JMP     SHORT DoSet           ;               return;
46574                                SpliceDone:                               ;               }
46575 00007F45 368F06[A405]             pop     word [SS:THISCDS+2]   ;   ThisCDS = TmpCDS;
46576 00007F4A 368F06[A205]             pop     word [SS:THISCDS]                   ;smr;SS Override
46577                                AllDone:
46578 00007F4F 31C9                     XOR     CX,CX
46579                                DoSet:
46580 00007F51 36C536[A205]             LDS     SI,[SS:THISCDS]             ;   ThisDPB = ThisCDS->devptr;;smr;SS Override
46581                                   ;les    di,[si+69]
46582 00007F56 C47C45                   LES     DI,[SI+curdir.devptr]
46583 00007F59 36893E[8A05]             MOV     [SS:THISDPB],DI                         ;smr;SS Override
46584 00007F5E 368C06[8C05]             MOV     [SS:THISDPB+2],ES                   ;smr;SS Override
46585                                Splice_retn:
46586 00007F63 C3                       retn
46587
46588                                ; 15/05/2019 - Retro DOS v4.0
46589                                ; DOSCODE:AEA9h (MSDOS 6.21, MSDOS.SYS)
46590
46591                                ; 04/12/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
46592                                ; DOSCODE:AE46h (MSDOS 5.0, MSDOS.SYS)
46593
46594                                ; 13/03/2024 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMDOS.COM)
46595                                ; PCDOS 7.1 IBMDOS.COM - DOSCODE:0C12Dh
46596
46597                                ;Break <$NameTrans - partially process a name>
46598                                ;--------------------------------------------------------------------------
46599                                ;   $NameTrans - allow users to see what names get mapped to. This call
46600                                ;   performs only string substitution and canonicalization, not splicing.  Due
46601                                ;   to Transpath playing games with devices, we need to insure that the output
46602                                ;   has drive letter and : in it.
46603                                ;
46604                                ;   Inputs:     DS:SI - source string for translation
46605                                ;               ES:DI - pointer to buffer
46606                                ;   Outputs:
46607                                ;       Carry Clear
46608                                ;               Buffer at ES:DI is filled in with data
46609                                ;               ES:DI point byte after nul byte at end of dest string in buffer
46610                                ;       Carry Set
46611                                ;               AX = error_path_not_found
46612                                ;   Registers modified: all
```

```
                                ;--------------------------------------------------------------------------
                                
                                _$NameTrans:
00007F64 1E                         push    ds
00007F65 56                         push    si
00007F66 06                         push    es
00007F67 57                         push    di
00007F68 51                         push    cx ; MSDOS 6.0
                                
                                    ; MSDOS 6.0
                                ; M027 - Start
                                ;
                                ; Sattrib must be set up with default values here. Otherwise, the value from
                                ; a previous DOS call is used for attrib and DevName thinks it is not a
                                ; device if the old call set the volume attribute bit. Note that devname in
                                ; dir2.asm gets ultimately called by Transpath. See also M026. Also save
                                ; and restore CX.
                                
                                    ;mov    ch,16h
00007F69 B516                       mov     ch,attr_hidden+attr_system+attr_directory
00007F6B E8EE02                     call    SetAttrib
                                
                                ; M027 - End
                                
                                    ; MSDOS 3.3 (& MSDOS 6.0)
00007F6E BF[BE03]                   MOV     DI,OPENBUF
00007F71 E8EAFC                     CALL    TransPath               ; to translation (everything)
00007F74 59                         pop     cx ; MSDOS 6.0
00007F75 5F                         pop     di
00007F76 07                         pop     es
00007F77 5E                         pop     si
00007F78 1F                         pop     ds
00007F79 7303                       JNC     short TransOK
00007F7B E9FA86                     jmp     SYS_RET_ERR
                                TransOK:
00007F7E BE[BE03]                   MOV     SI,OPENBUF
00007F81 16                         push    ss
00007F82 1F                         pop     ds
                                
                                ;GotText:
00007F83 E83A98                     call    FStrCpy
00007F86 E9E586                     jmp     SYS_RET_OK
                                
                                ;Break   <DriveFromText - return drive number from a text string>
                                ;--------------------------------------------------------------------------
                                ;   DriveFromText - examine DS:SI and remove a drive letter, advancing the
                                ;   pointer.
                                ;
                                ;   Inputs:     DS:SI point to a text string
                                ;   Outputs:    AL has drive number
                                ;               DS:SI advanced
                                ;   Registers modified: AX,SI.
                                ;--------------------------------------------------------------------------
                                
                                DriveFromText:
00007F89 30C0                       XOR     AL,AL                   ;       drive = 0;
                                    ;CMP    BYTE [SI],0             ;       if (*s &&
                                    ; 23/09/2023
00007F8B 3804                       cmp     [si],al ; 0
00007F8D 74D4                       jz      short Splice_retn
00007F8F 807C013A                   CMP     BYTE [SI+1],':'             ;           s[1] == ':') {
00007F93 75CE                       jnz     short Splice_retn
00007F95 AD                         LODSW                               ;           drive = (*s | 020) - 'a'+1;
00007F96 0C20                       OR      AL,20h  ; (convert to lowercase)
                                    ;sub    al,60h
00007F98 2C60                       SUB     AL,'a'-1                    ;           s += 2;
00007F9A 75C7                       jnz     short Splice_retn
00007F9C B0FF                       MOV     AL,-1                       ; nuke AL...
                                    ; 23/09/2023
                                    ;dec    al ; -1
00007F9E C3                         retn                                ;           }
                                
                                ;Break   <TextFromDrive - convert a drive number to a text string>
                                ;--------------------------------------------------------------------------
                                ;   TextFromDrive - turn AL into a drive letter: and put it at es:di with
                                ;   trailing :. TextFromDrive1 takes a 1-based number.
                                ;
                                ;   Inputs:     AL has 0-based drive number
                                ;   Outputs:    ES:DI advanced
                                ;   Registers modified: AX
                                ;--------------------------------------------------------------------------
                                
                                TextFromDrive:
00007F9F FEC0                       INC     AL
                                TextFromDrive1:
                                    ;add    al,40h
00007FA1 0440                       ADD     AL,'A'-1                    ;   *d++ = drive-1+'A';
00007FA3 B43A                       MOV     AH,":"  ; 3Ah               ;   strcat (d, ":");
00007FA5 AB                         STOSW
                                PathPref_retn:
00007FA6 C3                         retn
                                
                                ;Break   <PathPref - see if one path is a prefix of another>
                                ;--------------------------------------------------------------------------
                                ;   PathPref - compare DS:SI with ES:DI to see if one is the prefix of the
                                ;   other.  Remember that only at a pathchar break are we allowed to have a
                                ;   prefix: A:\ and A:\FOO
                                ;
                                ;   Inputs:     DS:SI potential prefix
                                ;               ES:DI string
                                ;   Outputs:    Zero set => prefix found
                                ;                   DI/SI advanced past matching part
                                ;               Zero reset => no prefix, DS/SI garbage
                                ;   Registers modified: CX
                                ;--------------------------------------------------------------------------
                                
                                PathPref:
00007FA7 E82D98                     call    DStrLen                 ; get length
00007FAA 49                         DEC     CX                      ; do not include nul byte
00007FAB F3A6                       REPZ    CMPSB                   ; compare
00007FAD 75F7                       jnz     short PathPref_retn     ; if NZ then return NZ
00007FAF 50                         push    ax                      ; save char register
00007FB0 8A44FF                     MOV     AL,[SI-1]               ; get last byte to match
00007FB3 E834DE                     call    PATHCHRCMP              ; is it a path char (Root!)
00007FB6 7406                       JZ      short Prefix            ; yes, match root (I hope)
                                NotSep:                                 ; 2/13/KK
00007FB8 268A05                     MOV     AL,[ES:DI]              ; get next char to match
00007FBB E87AFE                     CALL    PathSepGotCh            ; was it a pathchar?
                                Prefix:
00007FBE 58                         pop     ax                      ; get back original
00007FBF C3                         retn
                                
                                ;Break   <ScanPathChar - see if there is a path character in a string>
                                ;--------------------------------------------------------------------------
                                ;     ScanPathChar - search through the string (pointed to by DS:SI) for
```

```
                                ;     a path separator.
                                ;
                                ;     Input:    DS:SI target string (null terminated)
                                ;     Output:   Zero set => path separator encountered in string
                                ;               Zero clear => null encountered
                                ;     Registers modified: SI
                                ;-----------------------------------------------------------------------------

                                ScanPathChar:
00007FC0 AC                         LODSB                           ; fetch a character
00007FC1 E874FE                     call    PathSepGotCh
00007FC4 75FA                       JNZ     short ScanPathChar    ; not \, / or NUL => go back for more
                                    ;call   PATHCHRCMP              ; path separator?
                                    ;retn
                                    ; 18/12/2022
00007FC6 E921DE                     jmp     PATHCHRCMP

                                ;=============================================================================
                                ; FILE.ASM, MSDOS 6.0, 1991
                                ;=============================================================================
                                ; 14/07/2018 - Retro DOS v3.0

                                ; 13/05/2019 - Retro DOS v4.0
                                ; DOSCODE:AF10h (MSDOS 6.21, MSDOS.SYS)

                                ; 04/12/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
                                ; DOSCODE:AEADh (MSDOS 5.0, MSDOS.SYS)

                                ; 14/03/2024 - Retro DOS v5.0

                                ; MSDOS 2.11
                                ;BREAK <$Open - open a file handle>
                                ;-----------------------------------------------------------------------------
                                ;   Assembler usage:
                                ;           LDS     DX, Name
                                ;           MOV     AH, Open
                                ;           MOV     AL, access
                                ;           INT     int_command
                                ;
                                ;       ACCESS          Function
                                ;       ------          --------
                                ;       open_for_read   file is opened for reading
                                ;       open_for_write  file is opened for writing
                                ;       open_for_both   file is opened for both reading and writing.
                                ;
                                ;   Error returns:
                                ;           AX = error_invalid_access
                                ;              = error_file_not_found
                                ;              = error_access_denied
                                ;              = error_too_many_open_files
                                ;-----------------------------------------------------------------------------

                                ; MSDOS 6.0
                                ;   BREAK <$Open - open a file from a path string>
                                ;-----------------------------------------------------------------------------
                                ;
                                ;** $OPen - Open a File
                                ;
                                ;   given a path name in DS:DX and an open mode in AL, $Open opens the
                                ;   file and and returns a handle
                                ;
                                ;   ENTRY   (DS:DX) = pointer to asciz name
                                ;           (AL) = open mode
                                ;   EXIT    'C' clear if OK
                                ;             (ax) = file handle
                                ;           'C' set if error
                                ;             (ax) = error code
                                ;   USES    all
                                ;
                                ;-----------------------------------------------------------------------------

                                ; 13/05/2019 - Retro DOS v4.0
                                ; 04/12/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)

                                ; 14/03/2024 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMDOS.COM)
                                ; PCDOS 7.1 IBMDOS.COM - DOSCODE:0C194h
                                ; (MSDOS 6.22 MSDOS.SYS - DOSCODE:0AF10h)
                                ; (Windows ME IO.SYS - BIOSCODE:0A9A7h)

                                _$OPEN:
00007FC9 30E4                       xor     ah,ah  ; MSDOS 6.0
                                _$Open2:
                                    ;mov    ch,16h
00007FCB B516                       mov     ch,attr_hidden+attr_system+attr_directory
00007FCD E88C02                     call    SetAttrib
00007FD0 B9[FB31]                   mov     cx,DOS_OPEN

                                    ;xor    ah,ah  ; MSDOS 3.3

00007FD3 50                         push    ax

                                ;*  General file open/create code. The $CREATE call and the various
                                ;   $OPEN calls all come here.
                                ;
                                ;   We'll share a lot of the standard stuff of allocating SFTs, cracking
                                ;   path names, etc., and then dispatch to our individual handlers.
                                ;   WARNING - this info and list is just a guess, not definitive - jgl
                                ;
                                ;   (TOS) = create mode
                                ;   (CX) = address of routine to call to do actual function
                                ;   (DS:DX) = ASCIZ name
                                ;   SAttrib = Attribute mask

                                ;   Get a free SFT and mark it "being allocated"

                                AccessFile:
00007FD4 E80B99                     call    ECritSFT
00007FD7 E84CF7                     call    SFNFree                 ; get a free sfn
00007FDA E83299                     call    LCritSFT
                                    ;jc     short OpenFailJ                 ; oops, no free sft's
                                    ; 14/03/2024
00007FDD 7248                       jc      short OpenFail

00007FDF 36891E[AA05]               MOV     [SS:SFN],BX             ; save the SFN for later;smr;SS Override
00007FE4 36893E[9E05]               MOV     [SS:THISSFT],DI                 ; save the SF offset  ;smr;SS Override
00007FE9 368C06[A005]               MOV     [SS:THISSFT+2],ES       ; save the SF segment ;smr;SS Override

                                ;   Find a free area in the user's JFN table.

00007FEE E822F7                     call    JFNFree                 ; get a free jfn
                                    ;jnc    short SaveJFN
                                    ; 14/03/2024
00007FF1 7234                       jc      short OpenFail
                                    ;
```

```
                                ;OpenFailJ:
                                    ;JMP   OpenFail             ; there were free JFNs... try SFN
                                
                                SaveJFN:
00007FF3 36893E[AE05]               mov    [ss:PJFN],DI         ; save the jfn offset ;smr;SS Override
00007FF8 368C06[B005]               MOV    [ss:PJFN+2],ES       ; save the jfn segment;smr;SS Override
00007FFD 36891E[AC05]               MOV    [ss:JFN],BX          ; save the jfn itself ;smr;SS Override
                                
                                ;   We have been given an JFN. We lock it down to prevent other tasks from
                                ;   reusing the same JFN.
                                
00008002 368B1E[AA05]               MOV    BX,[ss:SFN]                                ;smr;SS Override
00008007 26881D                     MOV    [ES:DI],BL           ; assign the JFN
0000800A 89D6                       MOV    SI,DX                ; get name in appropriate place
0000800C BF[BE03]                   MOV    DI,OPENBUF           ; appropriate buffer
0000800F 51                         push   cx                   ; save routine to call
00008010 E84BFC                     call   TransPath            ; convert the path
00008013 5B                         pop    bx                   ; (bx) = routine to call
                                
00008014 36C536[9E05]               LDS    SI,[SS:THISSFT]                                ;smr;SS Override
                                    ;JC    short OpenCleanJ     ; no error, go and open file
                                    ; 14/03/2024
00008019 7260                       jc     short OpenClean
                                
0000801B 36803E[7A05]FF             CMP    byte [ss:CMETA],-1                         ;smr;SS Override
00008021 7408                       JZ     short SetSearch
                                    ;mov   al,2
00008023 B002                       MOV    AL,error_file_not_found ; no meta chars allowed
                                OpenCleanJ:
00008025 EB54                       JMP    short OpenClean
                                
                                    ; 14/03/2024 (PCDOS 7.1 IBMDOS.COM)
                                    ;;;
                                OpenFail:
00008027 FB                         STI
00008028 59                         pop    cx                   ; Clean stack
                                    ;
00008029 EB5D                       jmp    short OpenCritLeave
                                    ;;;
                                
                                SetSearch:
0000802B 58                         pop    ax                   ; Mode (Open), Attributes (Create)
                                
                                ;   We need to get the new inheritance bits.
                                
0000802C 31C9                       xor    cx,cx
                                    ; MSDOS 6.0
                                    ;mov   [si+2],cx ; 0
0000802E 894C02                     MOV    [SI+SF_ENTRY.sf_mode],cx ; initialize mode field to 0
                                    ;mov    [si+51],cx ; 0
00008031 894C33                     MOV    [SI+SF_ENTRY.sf_MFT],cx       ; clean out sharing info
                                    ;
00008034 81FB[FB31]                 CMP    BX,DOS_OPEN
00008038 7509                       JNZ    short _DoOper
                                    ;test   al,80h
0000803A A880                       test   AL,SHARING_NO_INHERIT ; look for no inher
0000803C 7405                       JZ     short _DoOper ; 10/08/2018
0000803E 247F                       AND    AL,7Fh               ; mask off inherit bit
                                    ;mov   cx,1000h
00008040 B90010                     MOV    CX,sf_no_inherit
                                _DoOper:
                                    ;; MSDOS 3.3
                                    ;;mov  word [si+2], 0
                                    ;;mov  word [si+33h], 0
                                    ;MOV   word [SI+SF_ENTRY.sf_mode],0
                                    ;MOV   word [SI+SF_ENTRY.sf_MFT],0
                                
                                    ; MSDOS 6.0
                                ;** Check if this is an extended open. If so you must set the
                                ;   modes in sf_mode. Call Set_EXT_mode to do all this. See
                                ;   Set_EXT_mode in creat.asm
                                
                                    ; MSDOS 6.0
                                    ;SAVE  <di, es>               ;M022 conditional removed here
00008043 57                         push   di
00008044 06                         push   es
00008045 1E                         push   ds
00008046 07                         pop    es
00008047 56                         push   si
00008048 5F                         pop    di                   ; (es:di) = SFT address
00008049 E89BB1                     call   Set_EXT_mode
                                    ;RESTORE <es, di>
0000804C 07                         pop    es
0000804D 5F                         pop    di
                                
                                    ;Context DS
0000804E 16                         push   ss
0000804F 1F                         pop    ds
                                
00008050 51                         push   cx
00008051 FFD3                       CALL   BX                   ; blam!
00008053 59                         pop    cx
00008054 C536[9E05]                 LDS    SI,[THISSFT]
00008058 7240                       JC     short OpenE2         ;AN000;FT. chek extended open hooks first
                                    ;jc    short OpenE ; MSDOS 3.3
                                
                                ;   The SFT was successfully opened. Remove busy mark.
                                
                                OpenOK:
                                    ;MOV   word [SI+SF_ENTRY.sf_ref_count],1
0000805A C7040100                   mov    word [SI],1
                                    ;or    [SI+5],cx
0000805E 094C05                     OR     [SI+SF_ENTRY.sf_flags],CX ; set no inherit bit if necessary
                                
                                ; If the open mode is 70, we scan the system for other SFT's with the same
                                ; contents. If we find one, then we can 'collapse' thissft onto the already
                                ; opened one. Otherwise we use this new one. We compare uid/pid/mode/mft
                                ;
                                ; Since this is only relevant on sharer systems, we stick this code into the
                                ; sharer.
                                
                                    ; 01/07/2024
                                    ;;;
00008061 1E                         push   ds
00008062 07                         pop    es
00008063 89F7                       mov    di,si
00008065 E8B0C9                     call   set_sftfcb_entry     ; set SFT-FCB entry
                                                                ; in the internal (SFT_FCB) table
                                                                ; (used for FCB calls only!)
                                    ;;;
                                
00008068 36A1[AC05]                 MOV    AX,[ss:JFN]                            ;smr;SS Override
0000806C 36FF1E[C000]               Call   far [ss:JShare+(12*4)]; 12 = ShCol   ;smr;SS Override
                                
```

```
00008071 36C706[AA05]FFFF           MOV     word [ss:SFN],-1      ; clear out sfn pointer       ;smr;SS Override
                                    ; 04/12/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
                                OpenOkj:
00008078 E9F385                     jmp     SYS_RET_OK            ; bye with no errors

                                ;   Extended Open hooks check
                                ;
                                ;   AL has error code. Stack has argument to dos_open/dos_create.

                                OpenClean:
0000807B 5B                         pop     bx                    ; clean off stack
                                OpenE:
                                    ;MOV    word [SI+SF_ENTRY.sf_ref_count],0 ; release SFT
0000807C C7040000                   mov     word [SI],0
00008080 36C536[AE05]               LDS     SI,[ss:PJFN]          ;smr;SS Override
00008085 C604FF                     MOV     BYTE [SI],0FFh        ; free the SFN...

                                    ; 14/03/2024
                                    ;JMP    SHORT OpenCritLeave
                                    ;
                                ;OpenFail:
                                    ;STI
                                    ;pop    cx                    ; Clean stack

                                OpenCritLeave:
00008088 36C706[AA05]FFFF           MOV     word [SS:SFN],-1      ; remove mark.

                                    ; MSDOS 6.0
                                ; File Tagging DOS 4.00
0000808F 36833E[2403]25             CMP     word [SS:EXTERR],error_Code_Page_Mismatched
                                                                  ;AN000;;FT. code page mismatch
00008095 7538                       JNZ     short NORERR          ;AN000;;FT. no
00008097 E9E685                     jmp     From_GetSet           ;AN000;;FT. yes

                                ; 14/03/2024
                                    ; MSDOS 6.0
                                ;Extended Open hooks check
                                OpenE2:                           ;AN000;;EO.
0000809A 83F857                     CMP     AX,error_invalid_parameter ;AN000;;EO. IFS extended open ?
0000809D 75DD                       JNZ     short OpenE           ;AN000;;EO. no.
0000809F EBE7                       JMP     short OpenCritLeave   ;AN000;;EO. keep handle

                                ; 15/03/2024
                                %if 0
                                NORERR:                           ;AN000;
                                ; File Tagging DOS 4.00

                                    jmp     SYS_RET_ERR           ; no free, return error
                                %endif

                                ; MSDOS 2.11
                                ;BREAK <$CREAT - create a new file and open him for input>
                                ;-----------------------------------------------------------------------------
                                ;   Assembler usage:
                                ;           LDS     DX, name
                                ;           MOV     AH, Creat
                                ;           MOV     CX, access
                                ;           INT     21h
                                ;       ; AX now has the handle
                                ;
                                ;   Error returns:
                                ;           AX = error_access_denied
                                ;              = error_path_not_found
                                ;              = error_too_many_open_files
                                ;-----------------------------------------------------------------------------

                                ; MSDOS 6.0
                                ;   BREAK <$Creat - create a brand-new file>
                                ;-----------------------------------------------------------------------------
                                ;
                                ;** $Creat - Create a File
                                ;
                                ;   $Creat creates the directory entry specified in DS:DX and gives it the
                                ;   initial attributes contained in CX
                                ;
                                ;   ENTRY   (DS:DX) = ASCIZ path name
                                ;           (CX) = initial attributes
                                ;   EXIT    'C' set if error
                                ;             (ax) = error code
                                ;           'C' clear if OK
                                ;             (ax) = file handle
                                ;   USES    all
                                ;
                                ;-----------------------------------------------------------------------------

                                _$CREAT:
000080A1 51                         push    cx                    ; Save attributes on stack
000080A2 B9[CA30]                   mov     CX,DOS_CREATE         ; routine to call
                                AccessSet:
                                    ;mov    byte [ss:SATTRIB],6
000080A5 36C606[6D05]06             mov     byte [ss:SATTRIB],attr_hidden+attr_system ;smr;SS Override
                                    ; 10/08/2018
000080AB E926FF                     JMP     AccessFile            ; use good ol' open


                                ; MSDOS 6.0 (MSDOS 3.3)
                                ;   BREAK <$CHMOD - change file attributes>
                                ;-----------------------------------------------------------------------------
                                ;
                                ;** $CHMOD - Change File Attributes
                                ;
                                ;   Assembler usage:
                                ;       LDS     DX, name
                                ;       MOV     CX, attributes
                                ;       MOV     AL,func (0=get, 1=set)
                                ;       INT     21h
                                ;   Error returns:
                                ;       AX = error_path_not_found
                                ;       AX = error_access_denied
                                ;
                                ;-----------------------------------------------------------------------------

                                    ; 04/12/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)

                                    ; 14/03/2024 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMDOS.COM)
                                    ; PCDOS 7.1 IBMDOS.COM - DOSCODE:0C27Ch
                                    ; (MSDOS 6.22 MSDOS.SYS - DOSCODE:0AFF4h)
                                    ; (Windows ME IO.SYS - BIOSCODE:0AAB2h)

                                _$CHMOD:
                                    ; 14/03/2024 (PCDOS 7.1 IBMDOS.COM)
                                    ;;;
000080AE 3CFF                       cmp     al,0FFh               ; MS-DOS 7.20 (Win98) - EXTENDED-LENGTH FILENAME OPERATIONS
                                                                  ; AX = 43FFh
```

```
                                                    ; BP = 5053h ('PS')
                                                    ; CL = function
                                                    ; 39h "mkdir" create directory
                                                    ; DS:DX -> ASCIZ pathname
                                                    ; 56h rename file
                                                    ; DS:DX -> ASCIZ filename of existing file (no wildcards)
                                                    ; ES:DI -> ASCIZ new filename (no wildcards)
                                                    ;
                                                    ; ref: Ralf Brown's Interrupt List
000080B0 7520                       jnz     short std_chmod
000080B2 81FD5350                   cmp     bp,5053h
000080B6 7515                       jnz     short chmod_x_2
000080B8 88CC                       mov     ah,cl
                                    ;mov    word [ss:PATHNAMELEN],128
000080BA 36C606[5411]80             mov     byte [ss:PATHNAMELEN],128 ; Retro DOS v5.0
000080C0 80F939                     cmp     cl,39h
000080C3 7503                       jne     short chmod_x_1
000080C5 E95AA7                     jmp     mkdir_x
                                chmod_x_1:
000080C8 80F956                     cmp     cl,56h
000080CB 7472                       je      short rename_x
                                chmod_x_2:
000080CD B001                       mov     al,1
                                    ;jmp    short NORERR
                                    ; 15/03/2024
                                NORERR:
000080CF E9A685                     jmp     SYS_RET_ERR

                                std_chmod:
                                    ;;;

                                    ; 05/08/2018 - Retro DOS v3.0
                                    ; IBMDOS.COM (MSDOS 3.3, 1987) - Offset 6FCCh
000080D2 BF[BE03]                   MOV     DI,OPENBUF              ; appropriate buffer
000080D5 50                         push    ax
000080D6 51                         push    cx                      ; save function and attributes
000080D7 89D6                       MOV     SI,DX                   ; get things in appropriate places
000080D9 E886FB                     call    TransPathSet            ; get correct path
000080DC 59                         pop     cx
000080DD 58                         pop     ax                      ; and get function and attrs back
000080DE 7255                       JC      short ChModErr          ; errors get mapped to path not found
000080E0 16                         push    ss                      ; set up for later possible calls
000080E1 1F                         pop     ds
000080E2 803E[7A05]FF               CMP     byte [CMETA],-1
000080E7 754C                       JNZ     short ChModErr
                                    ;mov    byte [SATTRIB],16h
000080E9 C606[6D05]16               MOV     byte [SATTRIB],attr_hidden+attr_system+attr_directory
000080EE 2C01                       SUB     AL,1                    ; fast way to discriminate
000080F0 7209                       JB      short ChModGet          ; 0 -> go get value
000080F2 7415                       JZ      short ChModSet          ; 1 -> go set value

                                ; 14/04/2024
                                %if 0
                                    ;mov    byte [EXTERR_LOCUS],1
                                    MOV     byte [EXTERR_LOCUS],errLOC_Unk ; Extended Error Locus
                                %else
                                    ; 14/03/2024 (PCDOS 7.1 IBMDOS.COM)
                                    ;;;
000080F4 E8E691                     call    set_exerr_locus_unk     ; Extended Error Locus
                                    ;;;
                                %endif
                                    ;mov    al,1
000080F7 B001                       mov     al,error_invalid_function ; bad value
                                    ; 04/12/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
                                chmod_errj:
                                    ;;jmp   SYS_RET_ERR
                                    ;jmp    short ChModE
000080F9 EBD4                       jmp     short NORERR    ; 06/12/2022
                                ChModGet:
000080FB E8C5AE                     call    GET_FILE_INFO           ; suck out the ol' info
000080FE 7237                       JC      short ChModE            ; error codes are already set for ret
00008100 E87483                     call    Get_User_Stack          ; point to user saved vaiables
                                    ;mov    [SI+4],ax
00008103 894404                     MOV     [SI+user_env.user_CX],AX ; return the attributes
                                    ; 04/12/2022 (MSDOS 5.0 MSDOS.SYS compatibility
                                OpenOkj2:
                                    ; 17/12/2022
                                    ;;jmp   SYS_RET_OK              ; say sayonara
                                    ;jmp    short OpenOkj
                                    ; 25/06/2019
00008106 E96885                     jmp     SYS_RET_OK_clc

                                ChModSet:
00008109 89C8                       MOV     AX,CX                   ; get attrs in position
0000810B E813AF                     call    SET_FILE_ATTRIBUTE      ; go set
0000810E 7227                       JC      short ChModE            ; errors are set
                                    ; 04/12/2022 (MSDOS 5.0 MSDOS.SYS compatibility
                                    ;jmp    SYS_RET_OK
                                OpenOkj3:
                                    ;jmp    short OpenOkj2
                                    ; 17/12/2022
00008110 E95B85                     jmp     SYS_RET_OK

                                ; 17/12/2022
                                %if 0
                                ChModErr:
                                NotFound:   ; 17/12/2022
                                    ;mov    al,3
                                    mov     al,error_path_not_found
                                ChModE:
                                UnlinkE:    ; 17/12/2022
                                    ; 04/12/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
                                    ;;jmp   SYS_RET_ERR
                                    ;jmp    short chmod_errj
                                    ; 17/12/2022
                                    jmp     short NORERR
                                %endif

                                ; 22/05/2019 - Retro DOS v4.0
                                ; DOSCODE:B039h (MSDOS 6.21, MSDOS.SYS)

                                ; 04/12/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
                                ; DOSCODE:AFD6h (MSDOS 5.0, MSDOS.SYS)

                                ;   BREAK <$UNLINK - delete a file entry>
                                ;-----------------------------------------------------------------------------
                                ;
                                ;** $UNLINK - Delete a File
                                ;
                                ;
                                ;   Assembler usage:
                                ;       LDS     DX, name
                                ;       IF VIA SERVER DOS CALL
                                ;        MOV     CX,SEARCH_ATTRIB
```

```
47233                                   ;       MOV     AH, Unlink
47234                                   ;       INT     21h
47235                                   ;
47236                                   ;   ENTRY   (ds:dx) = path name
47237                                   ;           (cx) = search_attribute, if via server_dos
47238                                   ;   EXIT    'C' clear if no error
47239                                   ;           'C' set if error
47240                                   ;             (ax) = error code
47241                                   ;                  = error_file_not_found
47242                                   ;                  = error_access_denied
47243                                   ;
47244                                   ;--------------------------------------------------------------------------
47245
47246                                       ; 15/03/2024 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMDOS.COM)
47247                                       ; PCDOS 7.1 IBMDOS.COM - DOSCODE:0C2E4h
47248
47249                                   _$UNLINK:
47250 00008113 51                          push    cx                      ; Save possible CX input parm
47251 00008114 89D6                        MOV     SI,DX                   ; Point at input string
47252 00008116 BF[BE03]                    MOV     DI,OPENBUF              ; temp spot for path
47253 00008119 E846FB                      call    TransPathSet            ; go get normalized path
47254 0000811C 59                          pop     cx
47255 0000811D 7216                        JC      short ChModErr          ; badly formed path
47256 0000811F 36803E[7A05]FF              CMP     byte [ss:CMETA],-1      ; meta chars?  ;smr;SS Override
47257 00008125 750E                        JNZ     short NotFound
47258 00008127 16                          push    ss
47259 00008128 1F                          pop     ds
47260                                      ;mov    ch,6
47261 00008129 B506                        mov     ch,attr_hidden+attr_system ; unlink appropriate files
47262 0000812B E82E01                      call    SetAttrib
47263 0000812E E833AA                      call    DOS_DELETE              ; remove that file
47264                                      ;JC     short UnlinkE           ; error is there
47265                                      ; 17/12/2022
47266 00008131 729C                        jc      short NORERR
47267
47268                                       ; 04/12/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
47269                                   UnlinkOk:
47270                                      ;jmp    SYS_RET_OK              ; okey doksy
47271 00008133 EBDB                        jmp     short OpenOkj3
47272
47273                                       ; 17/12/2022
47274                                   ChModErr:   ; 17/12/2022
47275                                   NotFound:
47276                                      ;mov    al,3
47277 00008135 B003                        MOV     AL,error_path_not_found
47278                                   ChModE:             ; 17/12/2022
47279                                   UnlinkE:
47280                                       ; 04/12/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
47281                                      ;;jmp   SYS_RET_ERR             ; bye
47282                                      ;jmp    short ChModE
47283                                      ; 17/12/2022
47284 00008137 EB96                        jmp     short NORERR
47285
47286                                   ;BREAK <$RENAME - move directory entries around>
47287                                   ;--------------------------------------------------------------------------
47288                                   ;
47289                                   ;   Assembler usage:
47290                                   ;       LDS     DX, source
47291                                   ;       LES     DI, dest
47292                                   ;       IF VIA SERVER DOS CALL
47293                                   ;         MOV   CX,SEARCH_ATTRIB
47294                                   ;       MOV     AH, Rename
47295                                   ;       INT     21h
47296                                   ;
47297                                   ;   Error returns:
47298                                   ;       AX = error_file_not_found
47299                                   ;          = error_not_same_device
47300                                   ;          = error_access_denied
47301                                   ;
47302                                   ;--------------------------------------------------------------------------
47303
47304                                       ; 04/12/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
47305                                       ; 15/03/2024 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMDOS.COM)
47306
47307                                   _$RENAME:
47308                                       ; 15/03/2024 (PCDOS 7.1 IBMDOS.COM)
47309                                       ;;;
47310                                       ;mov    word [ss:PATHNAMELEN],67
47311                                       ; 15/03/2024 - Retro DOS v5.0
47312 00008139 36C606[5411]43              mov     byte [ss:PATHNAMELEN],67 ; DIRSTRLEN = 67
47313                                   rename_x:
47314                                       ;;;
47315
47316                                       ; MSDOS 3.3 (& MSDOS 6.0)
47317 0000813F 51                          push    cx
47318 00008140 1E                          push    ds
47319 00008141 52                          push    dx                      ; save source and possible CX arg
47320 00008142 06                          PUSH    ES
47321 00008143 1F                          POP     DS                      ; move dest to source
47322 00008144 89FE                        MOV     SI,DI                   ; save for offsets
47323 00008146 BF[3E04]                    MOV     DI,RENBUF
47324 00008149 E816FB                      call    TransPathSet            ; munge the paths
47325 0000814C 36FF36[B205]                PUSH    word [ss:WFP_START]     ; get pointer  ;smr;SS Override
47326 00008151 368F06[B405]                POP     word [ss:REN_WFP]       ; stash it     ;smr;SS Override
47327 00008156 5E                          pop     si
47328 00008157 1F                          pop     ds
47329 00008158 59                          pop     cx                      ; get back source and possible CX arg
47330                                   epjc2:
47331 00008159 72DA                        JC      short ChModErr          ; get old error
47332 0000815B 36803E[7A05]FF              CMP     byte [ss:CMETA],-1                  ;smr;SS Override
47333 00008161 75D2                        JNZ     short NotFound
47334 00008163 51                          push    cx                      ; Save possible CX arg
47335 00008164 BF[BE03]                    MOV     DI,OPENBUF              ; appropriate buffer
47336 00008167 E8F8FA                      call    TransPathSet            ; wham
47337 0000816A 59                          pop     cx
47338                                      ;JC     short epjc2
47339                                      ; 15/03/2024
47340 0000816B 72C8                        jc      short ChModErr
47341
47342 0000816D 16                          push    ss
47343 0000816E 1F                          pop     ds
47344 0000816F 803E[7A05]FF                CMP     byte [CMETA],-1
47345 00008174 72BF                        JB      short NotFound
47346
47347                                       ; MSDOS 6.0
47348                                      ;PUSH   WORD [THISCDS]           ;AN000;;MS.save thiscds
47349                                      ;PUSH   WORD [THISCDS+2]         ;AN000;;MS.
47350                                      ; 15/03/2024 (PCDOS 7.1 IBMDOS.COM)
47351                                      ;;;
47352 00008176 C43E[A205]                  les     di,[THISCDS]
47353 0000817A 57                          push    di
47354 0000817B 06                          push    es
47355                                      ;;;
47356
```

```
47357 0000817C BF[BE03]                 MOV     DI,OPENBUF              ;AN000;;MS.
47358 0000817F 16                       PUSH    SS                      ;AN000;;MS.
47359 00008180 07                       POP     ES                      ;AN000;;MS.es:di-> source
47360 00008181 30C0                     XOR     AL,AL                   ;AN000;;MS.scan all CDS
47361                                 rnloop:                           ;AN000;
47362 00008183 E856FA                   call    GetCDSFromDrv           ;AN000;;MS.
47363 00008186 720F                     JC      short dorn              ;AN000;;MS. end of CDS
47364 00008188 E80496                   call    StrCmp                  ;AN000;;MS. current dir ?
47365 0000818B 7404                     JZ      short rnerr             ;AN000;;MS. yes
47366 0000818D FEC0                     INC     AL                      ;AN000;;MS. next
47367 0000818F EBF2                     JMP     short rnloop            ;AN000;;MS.
47368                                 rnerr:                                    ;AN000;
47369                                   ;ADD    SP,4                    ;AN000;;MS. pop thiscds
47370                                   ; 15/03/2024 (PCDOS 7.1 IBMDOS.COM)
47371 00008191 58                       pop     ax
47372 00008192 58                       pop     ax
47373
47374                                   ;error  error_current_directory    ;AN000;;MS.
47375 00008193 B010                     mov     al,error_current_directory
47376                                   ;jmp    SYS_RET_ERR
47377                                   ; 07/12/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
47378 00008195 EBA0                     jmp     short UnlinkE
47379                                 dorn:
47380
47381                                 ; 15/03/2024
47382                                 %if 0                                     ;AN000;
47383                                     POP     WORD [SS:THISCDS+2]     ;AN000;;MS.;PBUGBUG;SS REQD??
47384                                     POP     WORD [SS:THISCDS]       ;AN000;;MS.;PBUGBUG;SS REQD??
47385                                 %endif
47386 00008197 16                       push    ss
47387 00008198 1F                       pop     ds
47388
47389                                 ; 15/03/2024
47390                                 %if 1
47391 00008199 8F06[A405]               pop     word [THISCDS+2]
47392 0000819D 8F06[A205]               pop     word [THISCDS]
47393                                 %endif
47394                                   ; MSDOS 3.3 (& MSDOS 6.0)
47395                                   ;mov    ch,16h
47396 000081A1 B516                     mov     ch,attr_directory+attr_hidden+attr_system
47397                                                                     ; rename appropriate files
47398 000081A3 E8B600                   call    SetAttrib
47399 000081A6 E8CEAB                   call    DOS_RENAME              ; do the deed
47400 000081A9 728C                     JC      short UnlinkE           ; errors
47401
47402                                   ; 04/12/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
47403                                   ;jmp    SYS_RET_OK
47404 000081AB EB86                     jmp     short UnlinkOk
47405
47406                                 ; 04/12/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
47407
47408                                 ; 14/07/2018 - Retro DOS v3.0
47409                                 ; MSDOS 3.3 (& MSDOS 6.0)
47410
47411                                 ;Break <$CreateNewFile - Create a new directory entry>
47412                                 ;--------------------------------------------------------------------------
47413                                 ;   CreateNew - Create a new directory entry. Return a file handle if there
47414                                 ;   was no previous directory entry, and fail if a directory entry with
47415                                 ;   the same name existed previously.
47416                                 ;
47417                                 ;   Inputs: DS:DX point to an ASCIZ file name
47418                                 ;           CX contains default file attributes
47419                                 ;   Outputs:       Carry Clear:
47420                                 ;                      AX has file handle opened for read/write
47421                                 ;           Carry Set:
47422                                 ;                      AX has error code
47423                                 ;   Registers modified: All
47424                                 ;--------------------------------------------------------------------------
47425
47426                                 _$CreateNewFile:
47427 000081AD 51                       push    cx                      ; Save attributes on stack
47428 000081AE B9[9131]                 MOV     CX,DOS_Create_New       ; routine to call
47429 000081B1 E9F1FE                   JMP     AccessSet               ; use good ol' open
47430
47431                                 ;** BinToAscii - convert a number to a string.
47432                                 ;--------------------------------------------------------------------------
47433                                 ;   BinToAscii converts a 16 bit number into a 4 ascii characters.
47434                                 ;   This routine is used to generate temp file names so we don't spend
47435                                 ;   the time and code needed for a true hex number, we just use
47436                                 ;   A thorugh O.
47437                                 ;
47438                                 ;   ENTRY   (ax) = value
47439                                 ;           (es:di) = destination
47440                                 ;   EXIT    (es:di) updated by 4
47441                                 ;   USES    cx, di, flags
47442                                 ;--------------------------------------------------------------------------
47443
47444                                 ; MSDOS 3.3
47445                                 ;BinToAscii:
47446                                 ;   mov     cx,4
47447                                 ;bta5:
47448                                 ;   push    cx
47449                                 ;   mov     cl,4
47450                                 ;   rol     ax,cl
47451                                 ;   push    ax
47452                                 ;   and     al,0Fh
47453                                 ;   add     al,'0'
47454                                 ;   cmp     al,'9'
47455                                 ;   jbe     short bta6
47456                                 ;   add     al,7
47457                                 ;bta6:
47458                                 ;   stosb
47459                                 ;   pop     ax
47460                                 ;   pop     cx
47461                                 ;   loop    bta5
47462                                 ;   retn
47463
47464                                 ; 15/03/2024
47465                                 ; MSDOS 5.0-6.22 & Windows ME
47466                                 ; (MSDOS 6.22 MSDOS.SYS - DOSCODE:0B0D9h)
47467                                 ; (Windows ME IO.SYS - BIOSCODE:0ABA4h)
47468                                 %if 0
47469
47470                                 ; MSDOS 6.0
47471                                 BinToAscii:
47472                                     mov     cx,404h             ; (ch) = digit counter, (cl) = shift cnt
47473                                 bta5:
47474                                     ROL     AX,CL               ; move leftmost nibble into rightmost
47475                                     push    ax                  ; preserve remainder of digits
47476                                     AND     AL,0Fh              ; grab low nibble
47477                                     ADD     AL,'A'              ; turn into ascii
47478                                     STOSB                       ; drop in the character
47479                                     pop     ax                  ; (ax) = shifted number
47480                                     dec     ch
```

```
                                    jnz     short bta5              ; process 4 digits
                                    retn
                                %else
                                ; 15/03/2024
                                ; PCDOS 7.1 IBMDOS.COM - DOSCODE:0C385h

                                BinToAscii:
000081B4 50                         push    ax              ; convert a number to a string ; ax = value
000081B5 86E0                       xchg    ah,al
                                    ;db     0D4h,10h
000081B7 D410                       aam     10h             ; AH = AL / 16 and AL = remainder
000081B9 054141                     add     ax,4141h        ; 'AA'
000081BC AB                         stosw
000081BD 58                         pop     ax
                                    ;db     0D4h,10h
000081BE D410                       aam     10h
000081C0 054141                     add     ax,4141h        ; add ax,'AA'
000081C3 AB                         stosw
000081C4 C3                         retn
                                %endif

                                ;Break      <$CreateTempFile - create a unique name>
                                ;--------------------------------------------------------------------------
                                ;   $CreateTemp - given a directory, create a unique name in that directory.
                                ;   Method used is to get the current time, convert to a name and attempt
                                ;   a create new. Repeat until create new succeeds.
                                ;
                                ;   Inputs: DS:DX point to a null terminated directory name.
                                ;           CX  contains default attributes
                                ;   Outputs:       Unique name is appended to DS:DX directory.
                                ;           AX has handle
                                ;   Registers modified: all
                                ;--------------------------------------------------------------------------

                                    ; 10/07/2024
                                    ; 04/12/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
                                    ; 15/03/2024 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMDOS.COM)

                                _$CreateTempFile:
                                    ;Enter
000081C5 55                         push    bp
000081C6 89E5                       mov     bp,sp

                                    ;LocalVar  EndPtr,DWORD
                                    ;LocalVar  FilPtr,DWORD
                                    ;LocalVar  Attr,WORD

000081C8 83EC0A                     sub     sp,10

                                    ;test   cx,0FFD8h
000081CB F7C1D8FF                   test    CX,~attr_changeable
000081CF 7405                       JZ      short OKatts            ; Ok if no non-changeable bits set

                                ; We need this "hook" here to detect these cases (like user sets one both of
                                ; vol_id and dir bits) because of the structure of the or $CreateNewFile loop
                                ; below. The code loops on error_access_denied, but if one of the non
                                ; changeable attributes is specified, the loop COULD be infinite or WILL be
                                ; infinite because CreateNewFile will fail with access_denied always. Thus we
                                ; need to detect these cases before getting to the loop.

                                    ;mov    ax, 5
000081D1 B80500                     MOV     AX,error_access_denied
000081D4 EB7A                       JMP     SHORT SETTMPERR

                                OKatts:
                                    ;MOV    attr,CX                 ; save attribute
000081D6 894EF6                     mov      [bp-10],cx
                                    ;MOV    FilPtrL,DX              ; pointer to file
000081D9 8956F8                     mov     [bp-8],dx
                                    ;MOV    FilPtrH,DS
000081DC 8C5EFA                     mov     [bp-6],ds
                                    ;MOV    EndPtrH,DS              ; seg pointer to end of dir
000081DF 8C5EFE                     mov     [bp-2],ds
000081E2 1E                         PUSH    DS
000081E3 07                         POP     ES                      ; destination for nul search
000081E4 89D7                       MOV     DI,DX
000081E6 89F9                       MOV     CX,DI
000081E8 F7D9                       NEG     CX                      ; number of bytes remaining in segment
                                    ; MSDOS 6.0
000081EA 09C9                       OR      CX,CX                   ;AN000;MS. cx=0 ? ds:dx on segment boundary
000081EC 7501                       JNZ     short okok              ;AN000;MS. no
                                    ;MOV    CX,-1                   ;AN000;MS.
                                    ; 04/12/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
                                    ; 17/12/2022
000081EE 49                         dec     cx  ; mov cx,-1
                                    ;mov    cx,-1 ; 0FFFh
                                okok:                                       ;AN000;
000081EF 31C0                       XOR     AX,AX                   ;AN000;
000081F1 F2AE                       REPNZ   SCASB                   ;AN000;
                                                                    ;AN000;
000081F3 4F                         DEC     DI                      ; point back to the null
000081F4 268A45FF                   MOV     AL,[ES:DI-1]            ; Get char before the NUL
000081F8 E8EFDB                     call    PATHCHRCMP              ; Is it a path separator?
000081FB 7403                       JZ      short SETENDPTR                 ; Yes
                                STOREPTH:
000081FD B05C                       MOV     AL,'\'
000081FF AA                         STOSB                           ; Add a path separator (and INC DI)
                                SETENDPTR:
                                    ; 10/07/2024 - Retro DOS v5.0
                                    ;MOV    EndPtrL,DI              ; pointer to the tail
                                    ; 09/07/2024 (Retro DOS v4 BugFix - Erdogan Tan - Istanbul)
                                    ; (Note: I find this Retro DOS v4 Kernel bug while searching the reason
                                    ;  of the AutoCAD R12 running/startup problem. Now it is solved here.)
                                    ;mov    [bp-4],dl ; (Retro DOS v4 !Bug!)
00008200 897EFC                     mov     [bp-4],di ; !Fix!
                                CreateLoop:
00008203 16                         push    ss                      ; let ReadTime see variables
00008204 1F                         pop     ds
00008205 55                         push    bp
00008206 E88189                     call    READTIME                ; go get time
00008209 5D                         pop     bp
                                ;
                                ; Time is in CX:DX. Go drop it into the string.
                                ;
                                    ;les    di,EndPtr               ; point to the string
0000820A C47EFC                     les     di,[BP-4]
0000820D 89C8                       mov     ax,cx
0000820F E8A2FF                     call    BinToAscii              ; store upper word
00008212 89D0                       mov     ax,dx
00008214 E89DFF                     call    BinToAscii              ; store lower word
00008217 30C0                       xor     al,al
00008219 AA                         STOSB                           ; nul terminate
                                    ;LDS    DX,FilPtr               ; get name
0000821A C556F8                     lds     dx,[bp-8]
```

```
                            ;MOV    CX,Attr                 ; get attr
0000821D 8B4EF6             mov     cx,[bp-10]
00008220 55                 push    bp
00008221 E889FF             CALL    _$CreateNewFile                 ; try to create a new file
00008224 5D                 pop     bp
00008225 732A               JNC     short CreateDone        ; failed, go try again

                        ; The operation failed and the error has been mapped in AX. Grab the extended
                        ; error and figure out what to do.

                            ;; MSDOS 3.3                   ; M049 - start
                        ;;  mov     ax,[ss:EXTERR]                          ;smr;SS Override
                        ;;  cmp     al,error_file_exists
                        ;;  jz      short CreateLoop        ; file existed => try with new name
                        ;;  cmp     al,error_access_denied
                        ;;  jz      short CreateLoop        ; access denied (attr mismatch)

                            ; MSDOS 6.0
                            ;cmp    al,50h
00008227 3C50               CMP     AL,error_file_exists    ; Q: did file already exist
00008229 74D8               JZ      short CreateLoop        ; Y: try again
                            ;cmp    al,5
0000822B 3C05               CMP     AL,error_access_denied; Q: was it access denied
0000822D 7521               JNZ     short SETTMPERR                 ; N: Error out
                                                            ; Y: Check to see if we got this due
                                                            ;    to the network drive. Note that
                                                            ;    the redir will set the exterr
                                                            ;    to error_cannot_make if this is
                                                            ;    so.
                        ; 15/03/2024
                        %if 0
                            CMP     byte [SS:EXTERR],error_net_access_denied ; M069
                                                            ; See if it's REALLY an att mismatch
                            je      short SETTMPERR                 ; no, network error, stop
                        ;M070
                        ; If the user failed on an I24, we do not want to try again
                        ;
                            cmp     byte [SS:EXTERR],error_FAIL_I24 ;User failed on I24? ;M070
                            ;je     short SETTMPERR                 ;yes, do not try again ;M070

                            ;jmp    short CreateLoop        ;attr mismatch, try again ;M070
                            ; 17/12/2022
                            jne     short CreateLoop ; 10/06/2019
                            ; 04/12/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
                            ;jz     short SETTMPERR
                            ;jmp    short CreateLoop
                        %else
                            ; 15/03/2024 (PCDOS 7.1 IBMDOS.COM)
                            ;;;
0000822F 36A0[2403]         mov     al,[ss:EXTERR]
00008233 3C41               cmp     al,41h                  ; error_net_access_denied
00008235 7419               je      short SETTMPERR
00008237 3C53               cmp     al,53h                  ; error_FAIL_I24
00008239 7415               je      short SETTMPERR
0000823B 3C50               cmp     al,50h                  ; error_file_exists
0000823D 74C4               je      short CreateLoop
0000823F 36C43E[A205]       les     di,[ss:THISCDS]
00008244 83FFFF             cmp     di,0FFFFh ; -1
00008247 74BA               je      short CreateLoop        ; No CDS
                            ;;test  word [es:di+43h],8000h
                            ;test   word [es:di+curdir.flags],curdir_isnet
00008249 26F6454480         test    byte [es:di+curdir.flags+1],(curdir_isnet>>8)
0000824E 75B3               jnz     short CreateLoop
                            ;;;
                        %endif

                        ;;  MOV     AL,error_access_denied; Return this "extended" error
                                                            ; M049 - end
                        SETTMPERR:
00008250 F9                 STC
                        CreateDone:
                            ;Leave
00008251 89EC               mov     sp,bp
00008253 5D                 pop     bp
00008254 7203               JC      short CreateFail
00008256 E91584             jmp     SYS_RET_OK              ; success!
                        CreateFail:
00008259 E91C84             jmp     SYS_RET_ERR

                        ;   SetAttrib will set the search attribute (SAttrib) either to the normal
                        ;   (CH) or to the value in CL if the current system call is through
                        ;   serverdoscall.
                        ;
                        ;   Inputs: fSharing == FALSE => set sattrib to CH
                        ;           fSharing == TRUE => set sattrib to CL
                        ;   Outputs:       none
                        ;   Registers changed:    CX

                        SetAttrib:
                            ;test   byte [SS:FSHARING],-1           ;smr;SS Override
                            ;jnz    short Set
                            ; 15/03/2024
0000825C 36803E[7205]00     cmp     byte [ss:FSHARING],0
00008262 7502               jnz     short Set

00008264 88E9               mov     cl,ch
                        Set:
00008266 36880E[6D05]       mov     byte [ss:SATTRIB],cl            ;smr;SS Override
0000826B C3                 retn

                        ;--------------------------------------------------------------------------
                            ; 16/03/2024 - Retro DOS v5.0
                        ext_inval2:
                            ;mov    al,1
0000826C B001               mov     al,error_invalid_function
                        eo_err:
                            ;jmp    SYS_RET_ERR
0000826E EBE9               jmp     short CreateFail

                        ; 14/07/2018 - Retro DOS v3.0
                        ; MSDOS 6.0

                        ; 29/04/2019 - Retro DOS v4.0

                        ;Break      <Extended_Open- Extended open the file>
                        ;--------------------------------------------------------------------------
                        ; Input: AL= 0 reserved  AH=6CH
                        ;    BX= mode
                        ;    CL= create attribute  CH=search attribute (from server)
                        ;    DX= flag
                        ;    DS:SI = file name
                        ;    ES:DI = parm list
                        ;                    DD  SET EA list (-1) null
                        ;                    DW  n  parameters
```

```
                                ;                   DB  type (TTTTTTLL)
                                ;                   DW  IOMODE
                                ; Function: Extended Open
                                ; Output: carry clear
                                ;               AX= handle
                                ;               CX=1 file opened
                                ;                  2 file created/opened
                                ;                  3 file replaced/opened
                                ;     carry set: AX has error code
                                ;-----------------------------------------------------------------------------

                                    ; 04/12/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
                                    ; 16/03/2024 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMDOS.COM)

                                _$Extended_Open:                    ;AN000;
                                    ;ASSUME CS:DOSCODE,SS:DOSDATA   ;AN000;
00008270 368916[F405]               MOV     [SS:EXTOPEN_FLAG],DX    ;AN000;EO. save ext. open flag;smr;SS Override
00008275 36C706[F705]0000           MOV     word [SS:EXTOPEN_IO_MODE],0 ;AN000;EO. initialize IO mode;smr;SS Override
                                    ; 17/12/2022
0000827C F6C6FE                     test    dh,0FEh ; 04/12/2022
                                    ;;test  dx,0FE00h
                                    ;TEST   DX,RESERVED_BITS_MASK   ;AN000;EO. reserved bits 0 ?
0000827F 75EB                       JNZ     short ext_inval2        ;AN000;EO. no
00008281 88D4                       MOV     AH,DL                   ;AN000;EO. make sure flag is right
00008283 80FA00                     CMP     DL,0                    ;AN000;EO. all fail ?
00008286 74E4                       JZ      short ext_inval2        ;AN000;EO. yes, error
                                    ;and    dl,0Fh
00008288 80E20F                     AND     DL,EXISTS_MASK          ;AN000;EO. get exists action byte
0000828B 80FA02                     CMP     DL,2                    ;AN000;EO, > 2
0000828E 77DC                       JA      short ext_inval2        ;AN000;EO. yes, error
                                    ;and    ah,0F0h
00008290 80E4F0                     AND     AH,NOT_EXISTS_MASK      ;AN000;EO. get no exists action byte
00008293 80FC10                     CMP     AH,10H                  ;AN000;EO. > 10
00008296 77D4                       JA      short ext_inval2        ;AN000;EO. yes, error

00008298 368C06[FB05]               MOV     [SS:SAVE_ES],ES                 ;AN000;EO. save API parms;smr;SS Override
0000829D 36893E[F905]               MOV     [SS:SAVE_DI],DI                 ;AN000;EO.;smr;SS Override
000082A2 36FF36[F405]               PUSH    word [SS:EXTOPEN_FLAG]  ;AN000;EO.;smr;SS Override
000082A7 368F06[FD05]               POP     word [SS:SAVE_DX]       ;AN000;EO.;smr;SS Override
000082AC 36890E[FF05]               MOV     [SS:SAVE_CX],CX                 ;AN000;EO.;smr;SS Override
000082B1 36891E[0106]               MOV     [SS:SAVE_BX],BX                 ;AN000;EO.;smr;SS Override
000082B6 368C1E[0506]               MOV     [SS:SAVE_DS],DS                 ;AN000;EO.;smr;SS Override
000082BB 368936[0306]               MOV     [SS:SAVE_SI],SI                 ;AN000;EO.;smr;SS Override
000082C0 89F2                       MOV     DX,SI                   ;AN000;EO. ds:dx points to file name
000082C2 89D8                       MOV     AX,BX                   ;AN000;EO. ax= mode
                                ; 16/03/2024
                                %if 0
                                    JMP     SHORT goopen2           ;AN000;;EO. do normal
                                ext_inval2:                         ;AN000;;EO.
                                    ;mov    al,1
                                    mov     al,error_invalid_function ;AN000;EO.. invalid function
                                    ; 04/12/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
                                eo_err:
                                    ;jmp    SYS_RET_ERR
                                    jmp     short CreateFail
                                %endif

                                ; 16/03/2024
                                %if 0
                                ext_inval_parm:                             ;AN000;EO..
                                    POP     CX                      ;AN000;EO..  pop up satck
                                    POP     SI                      ;AN000;EO..
                                    ;error  error_invalid_data      ;AN000;EO..  invalid parms
                                    ;mov    al,13
                                    mov     al,error_invalid_data
                                    ; 04/12/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
                                    ;;jmp   SYS_RET_ERR
                                    ;jmp    short eo_err
                                    ; 17/12/2022
                                    jmp     short CreateFail
                                %endif

                                    ; 17/12/2022
                                ;error_return:                             ;AN000;EO.
                                ;   retn                        ;AN000;EO.. return with error

                                goopen2:                            ;AN000;
                                    ; 17/12/2022
                                    ;test   bh,20h
000082C4 F6C720                     test    bh,INT_24_ERROR>>8 ; 04/12/2022
                                    ;;test  bx,2000h
                                    ;TEST   BX,INT_24_ERROR                 ;AN000;EO.. disable INT 24 error ?
000082C7 7406                       JZ      short goopen            ;AN000;EO.. no
                                    ;or     byte [SS:EXTOPEN_ON],2
000082C9 36800E[F605]02             OR      byte [SS:EXTOPEN_ON],EXT_OPEN_I24_OFF ;AN000;EO.. set bit to disable;smr;SS Override
                                goopen:                             ;AN000;
                                    ;or     byte [SS:EXTOPEN_ON],1
000082CF 36800E[F605]01             OR      byte [SS:EXTOPEN_ON],EXT_OPEN_ON  ;AN000;EO.. set Extended Open active;smr;SS Override
                                    ;AND    word [SS:EXTOPEN_FLAG],0FFh  ;AN000;EO.create new ?;smr;SS Override
                                    ; 18/12/2022
000082D5 36C606[F505]00             mov     byte [SS:EXTOPEN_FLAG+1],0 ; AND word [SS:EXTOPEN_FLAG],0FFh
                                    ;cmp    word [SS:EXTOPEN_FLAG],10h
000082DB 36833E[F405]10             CMP     word [SS:EXTOPEN_FLAG],EXT_EXISTS_FAIL+EXT_NEXISTS_CREATE ;AN000;FT.;smr;SS Override
000082E1 7516                       JNZ     short chknext           ;AN000;;EO. no
000082E3 E8C7FE                     call    _$CreateNewFile                 ;AN000;;EO. yes
000082E6 723F                       JC      short error_return      ;AN000;;EO. error

000082E8 36803E[F605]00             CMP     byte [SS:EXTOPEN_ON],0  ;AN000;;EO. IFS does it;smr;SS Override
000082EE 7438                       JZ      short ok_return2        ;AN000;;EO. yes
                                    ;mov    word [SS:EXTOPEN_FLAG],2
000082F0 36C706[F405]0200           MOV     word [SS:EXTOPEN_FLAG],ACTION_CREATED_OPENED ;AN000;EO. created/opened;smr;SS Override
000082F7 EB7F                       JMP     short setXAttr ; 16/03/2024 ;AN000;;EO. set XAs

                                    ; 17/12/2022
                                ;ok_return2:
                                ;   jmp     SYS_RET_OK              ;AN000;;EO.

                                chknext:
                                    ; 17/12/2022
000082F9 36F606[F405]01             test    byte [SS:EXTOPEN_FLAG],EXT_EXISTS_OPEN ; 1
                                    ;;test  word [SS:EXTOPEN_FLAG],1
                                    ;TEST   word [SS:EXTOPEN_FLAG],EXT_EXISTS_OPEN ;AN000;;EO. exists open;smr;SS Override
000082FF 752A                       JNZ     short exist_open        ;AN000;;EO. yes
00008301 E89DFD                     call    _$CREAT                 ;AN000;;EO. must be replace open
00008304 7221                       JC      short error_return      ;AN000;;EO. return with error
00008306 36803E[F605]00             CMP     byte [SS:EXTOPEN_ON],0  ;AN000;;EO. IFS does it;smr;SS Override
0000830C 741A                       JZ      short ok_return2        ;AN000;;EO. yes
0000830E 36C706[F405]0200           MOV     word [SS:EXTOPEN_FLAG],ACTION_CREATED_OPENED ;AN000;EO. presume create/open;smr;SS Override
00008315 36F606[F605]04             TEST    byte [SS:EXTOPEN_ON],EXT_FILE_NOT_EXISTS ;AN000;;EO. file not exists ?;smr;SS Override
0000831B 755B                       JNZ     short setXAttr          ;AN000;;EO. no
0000831D 36C706[F405]0300           MOV     word [SS:EXTOPEN_FLAG],ACTION_REPLACED_OPENED ;AN000;;EO. replaced/opened;smr;SS Override
00008324 EB52                       JMP     SHORT setXAttr          ;AN000;;EO. set XAs
                                error_return2:
00008326 F9                         STC                             ; Set Carry again to flag error ;AN001;
```

```
                                error_return:       ; 17/12/2022
00008327 C3                         retn                            ;AN000;;EO. return with error
                                    ; 17/12/2022
                                ok_return:
                                ok_return2:
00008328 E94383                     jmp     SYS_RET_OK

                                exist_open:                         ;AN000;
                                    ;test   byte [SS:FSHARING],-1   ;AN000;;EO. server doscall?;smr;SS Override
                                    ;jz     short noserver          ;AN000;;EO. no
                                    ; 16/03/2024
                                    ;;;
0000832B 36803E[7205]00             cmp     byte [ss:FSHARING],0  ; server doscall?
00008331 7402                       jz      short noserver        ; no
                                    ;;;
00008333 88E9                       MOV     CL,CH                   ;AN000;;EO. cl=search attribute
                                noserver:
00008335 E893FC                     call    _$Open2                 ;AN000;;EO. do open
00008338 732F                       JNC     short ext_ok            ;AN000;;EO.
0000833A 36803E[F605]00             CMP     byte [SS:EXTOPEN_ON],0  ;AN000;;EO. error and IFS call;smr;SS Override
00008340 74E4                       JZ      short error_return2     ;AN000;;EO. return with error
                                local_extopen:
                                    ;cmp    ax,2
00008342 83F802                     CMP     AX,error_file_not_found   ;AN000;;EO. file not found error
00008345 75DF                       JNZ     short error_return2     ;AN000;;EO. no,
                                    ;;test  word [SS:EXTOPEN_FLAG],10h
                                    ; 17/12/2022
00008347 36F606[F405]10             test    byte [SS:EXTOPEN_FLAG],EXT_NEXISTS_CREATE ; 10h
                                    ;TEST   word [SS:EXTOPEN_FLAG],EXT_NEXISTS_CREATE ;AN000;;EO. want to fail;smr;SS Override
                                    ;JNZ    short do_creat          ;AN000;;EO. yes
                                    ;JMP    short extexit           ;AN000;;EO. yes
                                    ; 17/12/2022
0000834D 7446                       jz      short extexit ; 10/06/2019
                                    ; 04/12/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
                                    ;jnz    short do_creat
                                    ;jmp    short extexit
                                do_creat:
0000834F 368B0E[FF05]               MOV     CX,[SS:SAVE_CX]                 ;AN000;;EO. get ds:dx for file name;smr;SS Override
00008354 36C536[0306]               LDS     SI,[SS:SAVE_SI]                 ;AN000;;EO. cx = attribute;smr;SS Override
00008359 89F2                       MOV     DX,SI                   ;AN000;;EO.
0000835B E843FD                     call    _$CREAT                 ;AN000;;EO. do create
0000835E 7235                       JC      short extexit           ;AN000;;EO. error
                                    ;mov    word [SS:EXTOPEN_FLAG],2
00008360 36C706[F405]0200           MOV     word [SS:EXTOPEN_FLAG],ACTION_CREATED_OPENED
                                                                    ;AN000;;EO. is created/opened;smr;SS Override
00008367 EB0F                       JMP     SHORT setXAttr          ;AN000;;EO. set XAs

                                ext_ok:
00008369 36803E[F605]00             CMP     byte [SS:EXTOPEN_ON],0  ;AN000;;EO. IFS call ?;smr;SS Override
0000836F 74B7                       JZ      short ok_return                 ;AN000;;EO. yes
                                    ;mov    word [SS:EXTOPEN_FLAG],1
00008371 36C706[F405]0100           MOV     word [SS:EXTOPEN_FLAG],ACTION_OPENED ;AN000;;EO. opened;smr;SS Override
                                setXAttr:
                                    ; 29/04/2019
00008378 50                         push    ax
00008379 E8FB80                     call    Get_User_Stack          ;AN000;;EO.
0000837C 36A1[F405]                 MOV     AX,[SS:EXTOPEN_FLAG]    ;AN000;;EO.;smr;SS Override
                                    ;mov    [si+4],ax
00008380 894404                     MOV     [SI+user_env.user_CX],AX  ;AN000;;EO. set action code for cx
00008383 58                         pop     ax                      ;AN000;;EO.
00008384 8904                       mov     [si],ax
                                    ;MOV    [SI+user_env.user_AX],AX  ;AN000;;EO. set handle for ax
                                    ; 17/12/2022
00008386 EBA0                       jmp     short ok_return
                                ;ok_return:                         ;AN000;
                                    ;jmp    SYS_RET_OK              ;AN000;;EO.

                                ; 16/03/2024
                                %if 0
                                extexit2:                           ;AN000; ERROR RECOVERY
                                    POP     BX                      ;AN000;EO. close the handle
                                    PUSH    AX                      ;AN000;EO. save error code from set XA
                                    ;cmp    word [SS:EXTOPEN_FLAG],2
                                    CMP     word [SS:EXTOPEN_FLAG],ACTION_CREATED_OPENED
                                                                    ;AN000;EO. from create;smr;SS Override
                                    JNZ     short justopen          ;AN000;EO.
                                    LDS     SI,[SS:SAVE_SI]                 ;AN000;EO. cx = attribute;smr;SS Override
                                    LDS     DX,[SI]                 ;AN000;EO.
                                    call    _$UNLINK                ;AN000;EO. delete the file
                                    JMP     SHORT reserror          ;AN000;EO.

                                justopen:                           ;AN000;
                                    call    _$CLOSE                 ;AN000;EO. pretend never happend
                                reserror:                           ;AN000;
                                    POP     AX                      ;AN000;EO. restore error code from set XA

                                    JMP     SHORT extexit           ;AN000;EO.

                                ext_file_unfound:                   ;AN000;
                                    ;mov    ax,2
                                    MOV     AX,error_file_not_found   ;AN000;EO.
                                    JMP     SHORT extexit           ;AN000;EO.
                                ext_inval:                          ;AN000;
                                    ;mov    ax,1
                                    MOV     AX,error_invalid_function ;AN000;EO.

                                lockoperr:  ; 17/12/2022
                                extexit:
                                    jmp     SYS_RET_ERR             ;AN000;EO.

                                %endif

                                ;============================================================================
                                ; LOCK.ASM, MSDOS 6.0, 1991
                                ;============================================================================
                                ; 14/07/2018 - Retro DOS v3.0
                                ; 22/05/2019 - Retro DOS v4.0

                                ;BREAK <$LockOper - Lock Calls>
                                ;----------------------------------------------------------------------------
                                ;
                                ;   Assembler usage:
                                ;       MOV     BX, Handle      (DOS 3.3)
                                ;       MOV     CX, OffsetHigh
                                ;       MOV     DX, OffsetLow
                                ;       MOV     SI, LengthHigh
                                ;       MOV     DI, LengthLow
                                ;       MOV     AH, LockOper
                                ;       MOV     AL, Request
                                ;       INT     21h
                                ;
                                ;   Error returns:
                                ;       AX = error_invalid_handle
```

```
47977                                  ;          = error_invalid_function
47978                                  ;          = error_lock_violation
47979                                  ;
47980                                  ;   Assembler usage:
47981                                  ;       MOV     AX, 5C??         (DOS 4.00)
47982                                  ;
47983                                  ;                               0? lock all
47984                                  ;                               8? lock write
47985                                  ;                               ?2 lock multiple
47986                                  ;                               ?3 unlock multiple
47987                                  ;                               ?4 lock/read
47988                                  ;                               ?5 write/unlock
47989                                  ;                               ?6 add (lseek EOF/lock/write/unlock)
47990                                  ;       MOV     BX, Handle
47991                                  ;       MOV     CX, count or size
47992                                  ;       LDS     DX, buffer
47993                                  ;       INT     21h
47994                                  ;
47995                                  ;   Error returns:
47996                                  ;       AX = error_invalid_handle
47997                                  ;          = error_invalid_function
47998                                  ;          = error_lock_violation
47999                                  ;
48000                                  ;--------------------------------------------------------------------------
48001
48002                                      ; 04/12/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
48003
48004                                      ; 17/03/2024
48005                                      ; 16/03/2024 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMDOS.COM)
48006
48007                                  _$LockOper:
48008 00008388 3C01                      CMP     AL,1
48009 0000838A 770C                      JA      short lock_bad_func
48010
48011 0000838C 57                        PUSH    DI                              ; Save LengthLow
48012 0000838D E843F3                    call    SFFromHandle                    ; ES:DI -> SFT
48013 00008390 731B                      JNC     short lock_do                   ; have valid handle
48014 00008392 5F                        POP     DI                              ; Clean stack
48015                                    ;mov    al,6
48016 00008393 B006                      mov     al,error_invalid_handle
48017
48018                                    ; 16/03/2024
48019                                  extexit:
48020                                    ; 04/12/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
48021                                  lockoperr:
48022 00008395 E9E082                    jmp     SYS_RET_ERR
48023                                    ; 17/12/2022
48024                                    ;jmp    short lockoperr ; jmp SYS_RET_ERR
48025
48026                                  lock_bad_func:
48027
48028                                  ; 16/03/2024
48029                                  %if 0
48030                                    ;mov    byte [ss:EXTERR_LOCUS],1
48031                                    MOV     byte [SS:EXTERR_LOCUS],errLOC_Unk ; Extended Error Locus;smr;SS Override
48032                                  %else
48033                                    ; 16/03/2024 (PCDOS 7.1 IBMDOS.COM)
48034                                    ;;;
48035 00008398 E8428F                    call    set_exerr_locus_unk     ; Extended Error Locus
48036                                    ;;;
48037                                  %endif
48038                                    ;mov    al,1
48039 0000839B B001                      mov     al,error_invalid_function
48040                                    ; 04/12/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
48041                                  lockoperrj:
48042                                    ;jmp    SYS_RET_ERR
48043 0000839D EBF6                      jmp     short lockoperr
48044
48045                                    ; 22/05/2019 - Retro DOS v4.0
48046
48047                                    ; MSDOS 6.0
48048                                  ; Align_buffer call has been deleted, since it corrupts the DTA (6/5/88) P5013
48049                                  ; Dead code deleted, MD, 23 Mar 90
48050
48051                                  ;lock_do:
48052                                  ;   ; MSDOS 3.3
48053                                  ;   or      al,al
48054                                  ;   pop     ax
48055                                  ;   jz      short DOS_Lock
48056                                  ;DOS_Unlock:
48057                                  ;   ;test   word [es:di+5],8000h
48058                                  ;   test    word [ES:DI+SF_ENTRY.sf_flags],sf_isnet
48059                                  ;   JZ      short LOCAL_UNLOCK
48060                                  ;   push    ax
48061                                  ;   mov     ax,110Bh
48062                                  ;   int     2Fh     ; Multiplex - NETWORK REDIRECTOR - UNLOCK REGION OF FILE
48063                                  ;                   ; BX = file handle, CX:DX = starting offset, SI = high word of size
48064                                  ;                   ; STACK: WORD low word of size, ES:DI -> SFT for file
48065                                  ;                   ; SFT DPB field -> DPB of drive containing file
48066                                  ;                   ; Return: CF set error
48067                                  ;   pop     bx
48068                                  ;   jmp     short ValChk
48069                                  ;
48070                                  ;LOCAL_UNLOCK:
48071                                  ;   Call    far [ss:JShare+(7*4)] ; 7 = clr_block ;smr;SS Override
48072                                  ;ValChk:
48073                                  ;   JNC     short Lock_OK
48074                                  ;lockerror:
48075                                  ;   jmp     SYS_RET_ERR
48076                                  ;Lock_OK:
48077                                  ;   ;MOV    AX,[SS:Temp_VAR] ;AN000;;MS. AX= number of bytes ;smr;SS Override
48078                                  ;   jmp     SYS_RET_OK
48079                                  ;DOS_Lock:
48080                                  ;   ;test   word [es:di+5],8000h
48081                                  ;   test    word [ES:DI+SF_ENTRY.sf_flags],sf_isnet
48082                                  ;   JZ      short LOCAL_LOCK
48083                                  ;   ;CallInstall NET_XLock,MultNET,10
48084                                  ;   mov     ax, 110Ah
48085                                  ;   int     2Fh     ; Multiplex - NETWORK REDIRECTOR - LOCK REGION OF FILE
48086                                  ;                   ; BX = file handle, CX:DX = starting offset, SI = high word of size
48087                                  ;                   ; STACK: WORD low word of size, ES:DI -> SFT
48088                                  ;                   ; SFT DPB field -> DPB of drive containing file, SS = DOS CS
48089                                  ;                   ; Return: CF set error
48090                                  ;   JMP     short ValChk
48091                                  ;
48092                                  ;LOCAL_LOCK:
48093                                  ;   Call    far [ss:JShare+(6*4)] ; 6 = Set_Block ;smr;SS Override
48094                                  ;   JMP     short ValChk
48095
48096                                  ; 17/12/2022
48097                                  LOCAL_UNLOCK:
48098                                      ; MSDOS 3.3
48099                                      ;Call   far [ss:JShare+(7*4)] ; 7 = clr_block ;smr;SS Override
48100                                      ; MSDOS 6.0
```

```
48101 0000839F FF1E[AC00]               Call    far [JShare+(7*4)]    ; 7 = clr_block ;smr;SS Override
48102                                   ValChk:
48103 000083A3 7302                         JNC     short Lock_OK
48104                                   lockerror:
48105                                       ; 04/12/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
48106                                       ;;jmp   SYS_RET_ERR
48107                                       ;jmp    short lockoperrj
48108                                       ; 17/12/2022
48109 000083A5 EBEE                         jmp     short lockoperr       ; jmp SYS_RET_ERR
48110                                   Lock_OK:
48111                                       ;MOV    AX,[SS:TEMP_VAR] ;AN000;;MS. AX= number of bytes ;smr;SS Override
48112                                       ; 10/06/2019
48113 000083A7 A1[0C06]                     mov     ax,[TEMP_VAR]
48114 000083AA E9C182                       jmp     SYS_RET_OK
48115
48116                                       ; 22/05/2019
48117                                   lock_do:
48118                                       ; MSDOS 6.0
48119 000083AD 89C3                         MOV     BX,AX                         ; save AX
48120 000083AF BD[A903]                     MOV     BP,Lock_Buffer                ; get DOS LOCK buffer
48121                                       ; 04/12/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
48122                                       ;;mov   [bp+0],dx
48123                                       ;MOV    [BP+LockBuf.Lock_position],DX ; set low offset
48124                                       ; 15/12/2022
48125 000083B2 895600                       mov     [bp],dx
48126                                       ;mov    [bp+2],cx
48127 000083B5 894E02                       MOV     [BP+LockBuf.Lock_position+2],CX; set high offset
48128
48129                                       ; 16/03/2024
48130                                       ;POP    CX                            ; get low length
48131                                       ;;mov   [bp+4],cx
48132                                       ;MOV    [BP+LockBuf.Lock_length],CX   ; set low length
48133 000083B8 8F4604                       pop     word [bp+LockBuf.Lock_length]
48134
48135                                       ;mov    [bp+6],si
48136 000083BB 897606                       MOV     [BP+LockBuf.Lock_length+2],SI ; set high length
48137 000083BE B90100                       MOV     CX,1                          ; one range
48138
48139                                   ;   PUSH    CS                            ;
48140                                   ;   POP     DS                            ; DS:DX points to
48141
48142 000083C1 16                           push    ss
48143 000083C2 1F                           pop     ds
48144
48145 000083C3 89EA                         MOV     DX,BP                         ; Lock_Buffer
48146                                       ;test   al,1
48147 000083C5 A801                         TEST    AL,UNLOCK_ALL                 ; function 1
48148                                       ;JNZ    short DOS_Unlock              ; yes
48149                                       ;JMP    short DOS_Lock                ; function 0
48150                                       ; 17/12/2022
48151                                       ; 10/06/2019
48152 000083C7 740E                         jz      short DOS_Lock
48153                                       ; 04/12/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
48154                                       ;JNZ    short DOS_Unlock
48155                                       ;JMP    short DOS_Lock
48156
48157                                   DOS_Unlock:
48158                                       ;;test  word [es:di+5],8000h
48159                                       ;test   word [ES:DI+SF_ENTRY.sf_flags],sf_isnet
48160 000083C9 26F6450680                  test    byte [ES:DI+SF_ENTRY.sf_flags+1],(sf_isnet>>8)
48161 000083CE 74CF                        JZ      short LOCAL_UNLOCK
48162
48163                                   ; 17/03/2024
48164                                   ;lock_unlock: ; 22/05/2019
48165
48166                                       ;CallInstall Net_Xlock,MultNET,10
48167                                   ;
48168                                   ;   ; MSDOS 3.3
48169                                   ;   ;mov     ax,110Bh
48170                                   ;   ;int     2Fh   ; Multiplex - NETWORK REDIRECTOR - UNLOCK REGION OF FILE
48171                                   ;                  ; BX = file handle, CX:DX = starting offset, SI = high word of size
48172                                   ;                  ; STACK: WORD low word of size, ES:DI -> SFT for file
48173                                   ;                  ; SFT DPB field -> DPB of drive containing file
48174                                   ;                  ; Return: CF set error
48175
48176                                   ; 17/03/2024 - Retro DOS v5.0
48177                                   lock_unlock:
48178
48179                                       ; MSDOS 6.0
48180 000083D0 B80A11                     mov     ax,110Ah
48181 000083D3 CD2F                       int     2Fh    ; Multiplex - NETWORK REDIRECTOR - LOCK REGION OF FILE
48182                                                    ; BX = file handle, CX:DX = starting offset, SI = high word of size
48183                                                    ; STACK: WORD low word of size, ES:DI -> SFT
48184                                                    ; SFT DPB field -> DPB of drive containing file, SS = DOS CS
48185                                                    ; Return: CF set error
48186
48187 000083D5 EBCC                       JMP     SHORT ValChk
48188
48189                                   ; 17/12/2022
48190                                   %if 0
48191                                   LOCAL_UNLOCK:
48192                                       ; MSDOS 3.3
48193                                       ;Call   far [ss:JShare+(7*4)] ; 7 = clr_block ;smr;SS Override
48194                                       ; MSDOS 6.0
48195                                       Call    far [JShare+(7*4)]    ; 7 = clr_block ;smr;SS Override
48196                                   ValChk:
48197                                       JNC     short Lock_OK
48198                                   lockerror:
48199                                       ; 04/12/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
48200                                       ;jmp    SYS_RET_ERR
48201                                       jmp     short lockoperrj
48202                                   Lock_OK:
48203                                       ;MOV    AX,[SS:TEMP_VAR] ;AN000;;MS. AX= number of bytes ;smr;SS Override
48204                                       ; 10/06/2019
48205                                       mov     ax,[TEMP_VAR]
48206                                       jmp     SYS_RET_OK
48207                                   %endif
48208
48209                                   DOS_Lock:
48210                                       ;;test  word [es:di+5],8000h
48211                                       ;test   word [ES:DI+SF_ENTRY.sf_flags],sf_isnet
48212 000083D7 26F6450680                  test    byte [ES:DI+SF_ENTRY.sf_flags+1],(sf_isnet>>8)
48213                                       ;JZ     short LOCAL_LOCK
48214                                       ; 17/03/2024
48215 000083DC 75F2                        jnz     short lock_unlock
48216
48217                                   ; 17/03/2024
48218                                   %if 0
48219                                       ;CallInstall NET_XLock,MultNET,10
48220
48221                                       mov     ax,110Ah
48222                                       int     2Fh    ; Multiplex - NETWORK REDIRECTOR - LOCK REGION OF FILE
48223                                                      ; BX = file handle, CX:DX = starting offset, SI = high word of size
48224                                                      ; STACK: WORD low word of size, ES:DI -> SFT
```

```
                                                ; SFT DPB field -> DPB of drive containing file, SS = DOS CS
                                                ; Return: CF set error

                                    JMP     short ValChk
                                %endif

                                LOCAL_LOCK:
                                    ; MSDOS 3.3
                                    ;Call   far [ss:JShare+(6*4)] ; 6 = Set_Block ;smr;SS Override
                                    ; MSDOS 6.0
000083DE FF1E[A800]                 Call    far [JShare+(6*4)]    ; 6 = Set_Block ;smr;SS Override

000083E2 EBBF                       JMP     short ValChk

                                ; 14/07/2018 - Retro DOS v3.0
                                ; LOCK_CHECK
                                ;MSDOS 6.0 (& MSDOS 3.3)

                                ;--------------------------------------------------------------------------
                                ; Inputs:
                                ;   Outputs of SETUP
                                ;   [USER_ID] Set
                                ;   [PROC_ID] Set
                                ; Function:
                                ;   Check for lock violations on local I/O
                                ;   Retries are attempted with sleeps in between
                                ; Outputs:
                                ;    Carry clear
                                ;   Operation is OK
                                ;    Carry set
                                ;   A lock violation detected
                                ; Outputs of SETUP preserved
                                ;--------------------------------------------------------------------------

                                    ; 04/12/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
                                    ; 22/05/2019 - Retro DOS v4.0
                                LOCK_CHECK:
000083E4 8B1E[1A00]                 MOV     BX,[RetryCount]       ; Number retries
                                LockRetry:
000083E8 53                         push    bx              ; save regs
000083E9 50                         push    ax ; MSDOS 6.0

                                    ;MSDOS 3.3
                                    ;Call   far [ss:JShare+(8*4)] ; 8 = chk_block
                                    ;MSDOS 6.0
000083EA FF1E[B000]                 Call    far [JShare+(8*4)]    ; 8 = chk_block

000083EE 58                         pop     ax ; MSDOS 6.0
000083EF 5B                         pop     bx              ; restrore regs
000083F0 7307                       jnc     short lc_ret_label ; There are no locks (retnc)
                                LockN:
000083F2 E8EF93                     call    Idle            ; wait a while
000083F5 4B                         DEC     BX              ; remember a retry
000083F6 75F0                       JNZ     short LockRetry       ; more retries left...
000083F8 F9                         STC
                                lc_ret_label:
000083F9 C3                         retn

                                ; 14/07/2018 - Retro DOS v3.0
                                ; LOCK_VIOLATION
                                ;MSDOS 6.0 (& MSDOS 3.3)

                                ;--------------------------------------------------------------------------
                                ; Inputs:
                                ;   [THISDPB] set
                                ;   [READOP] indicates whether error on read or write
                                ; Function:
                                ;   Handle Lock violation on compatibility (FCB) mode SFTs
                                ; Outputs:
                                ;   Carry set if user says FAIL, causes error_lock_violation
                                ;   Carry clear if user wants a retry
                                ;
                                ; DS, ES, DI, CX preserved, others destroyed
                                ;--------------------------------------------------------------------------

                                    ; 19/03/2024 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMDOS.COM)

                                LOCK_VIOLATION:
000083FA 1E                         PUSH    DS
000083FB 06                         PUSH    ES
000083FC 57                         PUSH    DI
000083FD 51                         PUSH    CX
                                    ;mov    ax,21h
000083FE B82100                     MOV     AX,error_lock_violation
                                    ;mov    byte [ALLOWED],18h
00008401 C606[4B03]18               MOV     byte [ALLOWED],Allowed_FAIL+Allowed_RETRY
00008406 C42E[8A05]                 LES     BP,[THISDPB]
0000840A BF0100                     MOV     DI,1          ; Fake some registers
0000840D 89F9                       MOV     CX,DI

                                    ; 19/03/2024 (PCDOS 7.1 IBMDOS.COM)
                                    ;;;
0000840F 31D2                       xor     dx,dx   ; 0
                                    ;cmp    [es:bp+0Fh],dx
00008411 2639560F                   cmp     [es:bp+DPB.FAT_SIZE],dx ; 0 ?
00008415 750E                       jnz     short lockv_1  ; not FAT32
                                    ; FAT32
                                    ;mov    dx,[es:bp+2Bh]
00008417 268B562B                   mov     dx,[es:bp+DPB.FCLUS_FSECTOR+2]
0000841B 8916[0706]                 mov     [HIGH_SECTOR],dx
                                    ;mov    dx,[es:bp+29h]
0000841F 268B5629                   mov     dx,[es:bp+DPB.FCLUS_FSECTOR]
00008423 EB08                       jmp     short lockv_2
                                lockv_1:
00008425 8916[0706]                 mov      [HIGH_SECTOR],dx ; 0
                                    ;;;

                                    ;mov    dx,[es:bp+11]
00008429 268B560B                   MOV     DX,[ES:BP+DPB.FIRST_SECTOR]
                                lockv_2:    ; 19/03/2024
0000842D E824DC                     call    HARDERR
00008430 59                         POP     CX
                                sharev_3:   ; 01/07/2024
00008431 5F                         POP     DI
00008432 07                         POP     ES
00008433 1F                         POP     DS
00008434 3C01                       CMP     AL,1
00008436 74C1                       jz      short lc_ret_label ; 1 = retry, carry clear
00008438 F9                         STC
00008439 C3                         retn

                                ; 14/07/2018 - Retro DOS v3.0

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

```
48349
48350                                   ;   do a retz to return error
48351
48352                                       ; 22/05/2019 - Retro DOS v4.0
48353                                   CheckShare:
48354                                       ; MSDOS 3.3
48355                                       ;cmp    byte [cs:fShare],0
48356                                       ;retn
48357
48358                                       ; MSDOS 6.0
48359 0000843A 1E                          push    ds                      ;smr;
48360                                       ;getdseg <ds>                   ; ds -> dosdata
48361 0000843B 2E8E1E[0700]                mov     ds,[cs:DosDSeg]
48362 00008440 803E[0303]00                cmp     byte [fShare],0
48363 00008445 1F                          pop     ds                      ;smr;
48364 00008446 C3                          retn
48365
48366                                   ;==========================================================================
48367                                   ; SHARE.ASM, MSDOS 6.0, 1991
48368                                   ;==========================================================================
48369                                   ; 14/07/2018 - Retro DOS v3.0
48370                                   ; 22/05/2019 - Retro DOS v4.0
48371                                   ; 19/03/2024 - Retro DOS v5.0
48372
48373                                   ; SHARE_CHECK
48374                                   ;--------------------------------------------------------------------------
48375                                   ; Inputs:
48376                                   ;       [THISSFT] Points to filled in local file/device SFT for new
48377                                   ;               instance of file sf_mode ALWAYS has mode (even on FCB SFTs)
48378                                   ;       [WFP_START] has full path of name
48379                                   ;       [USER_ID] Set
48380                                   ;       [PROC_ID] Set
48381                                   ; Function:
48382                                   ;       Check for sharing violations on local file/device access
48383                                   ; Outputs:
48384                                   ;    Carry clear
48385                                   ;       Sharing approved
48386                                   ;    Carry set
48387                                   ;       A sharing violation detected
48388                                   ;          AX is error code
48389                                   ; USES    ALL but DS
48390                                   ;--------------------------------------------------------------------------
48391
48392                                       ; 22/05/2019 - Retro DOS v4.0
48393                                   SHARE_CHECK:
48394                                       ; 26/07/2019
48395 00008447 FF1E[9400]                  call    far [JShare+(1*4)]    ; 1 = MFT_Enter
48396                                   shchk_retn:
48397 0000844B C3                          retn
48398
48399                                   ; SHARE_VIOLATION
48400                                   ;--------------------------------------------------------------------------
48401                                   ; Inputs:
48402                                   ;       [THISDPB] Set
48403                                   ;       AX has error code
48404                                   ; Function:
48405                                   ;       Handle Sharing errors
48406                                   ; Outputs:
48407                                   ;       Carry set if user says FAIL, causes error_sharing_violation
48408                                   ;       Carry clear if user wants a retry
48409                                   ;
48410                                   ; DS, ES, DI preserved, others destroyed
48411                                   ;--------------------------------------------------------------------------
48412
48413                                       ; 19/03/2024 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMDOS.COM)
48414
48415                                   SHARE_VIOLATION:
48416 0000844C 1E                          PUSH    DS
48417 0000844D 06                          PUSH    ES
48418 0000844E 57                          PUSH    DI
48419                                       ; 19/03/2024
48420 0000844F 31D2                        xor     dx,dx   ; Retro DOS v5.0
48421                                       ;
48422                                       ;MOV    byte [READOP],0       ; All share errors are reading
48423 00008451 8816[7505]                  mov     [READOP],dl ; 0
48424                                       ;
48425                                       ;mov    byte [ALLOWED],18h
48426 00008455 C606[4B03]18                MOV     byte [ALLOWED],Allowed_FAIL+Allowed_RETRY
48427 0000845A C42E[8A05]                  LES     BP,[THISDPB]
48428 0000845E BF0100                      MOV     DI,1                  ; Fake some registers
48429 00008461 89F9                        MOV     CX,DI
48430
48431                                       ; 19/03/2024
48432                                       ;;mov   dx,[es:bp+17]
48433                                       ;MOV    DX,[ES:BP+DPB.DIR_SECTOR]
48434
48435                                       ; 19/03/2024 - Retro DOS v5.0
48436                                       ; PCDOS 7.1 IBMDOS.COM
48437                                       ;;;
48438                                       ;cmp    word [es:bp+0Fh],0
48439 00008463 2639560F                    cmp     word [es:bp+DPB.FAT_SIZE],dx ; 0
48440 00008467 740A                        jz      short sharev_1 ; FAT32
48441                                       ; FAT16 or FAT12
48442                                       ;mov    word [HIGH_SECTOR],0
48443 00008469 8916[0706]                  mov     [HIGH_SECTOR],dx ; 0
48444                                       ;mov    dx,[es:bp+11h]
48445 0000846D 268B5611                    mov     dx,[es:bp+DPB.DIR_SECTOR]
48446 00008471 EB0C                        jmp     short sharev_2
48447                                   sharev_1:
48448                                       ;mov    dx,[es:bp+2Bh]
48449 00008473 268B562B                    mov     dx,[es:bp+DPB.FCLUS_FSECTOR+2]
48450 00008477 8916[0706]                  mov     [HIGH_SECTOR],dx
48451                                       ;mov    dx,[es:bp+29h]
48452 0000847B 268B5629                    mov     dx,[es:bp+DPB.FCLUS_FSECTOR]
48453                                   sharev_2:
48454                                       ;;;
48455 0000847F E8D2DB                      call    HARDERR
48456                                   ; 01/07/2024
48457                                   %if 0
48458                                       POP     DI
48459                                       POP     ES
48460                                       POP     DS
48461                                       CMP     AL,1
48462                                       jz      short shchk_retn      ; 1 = retry, carry clear
48463                                       STC
48464                                       retn
48465                                   %else
48466 00008482 EBAD                        jmp     short sharev_3
48467                                   %endif
48468
48469                                   ;--------------------------------------------------------------------------
48470                                   ;   ShareEnd - terminate sharing info on a particular SFT/UID/PID. This does
48471                                   ;       NOT perform a close, it merely asserts that the sharing information
48472                                   ;       for the SFT/UID/PID may be safely released.
```

```
                                ;
                                ;   Inputs:     ES:DI points to an SFT
                                ;   Outputs:    None
                                ;   Registers modified: all except DS,ES,DI
                                ;-----------------------------------------------------------------------

                                ShareEnd:
                                    ; 26/07/2019
00008484 FF1E[9800]             call    far [JShare+(2*4)]    ; 2 = MFTClose
00008488 C3                     retn

                                ;Break <ShareEnter - attempt to enter a node into the sharing set>
                                ;-----------------------------------------------------------------------
                                ;   ShareEnter - perform a retried entry of a nodde into the sharing set. If
                                ;   the max number of retries is exceeded, we notify the user via int 24.
                                ;
                                ;   Inputs:     ThisSFT points to the SFT
                                ;               WFP_Start points to the WFP
                                ;   Outputs:    Carry clear => successful entry
                                ;               Carry set => failed system call
                                ;   Registers modified: all
                                ;-----------------------------------------------------------------------

                                ShareEnter:
00008489 51                         push    cx
                                retry:
0000848A 8B0E[1A00]                 mov     cx,[RetryCount]
                                attempt:
0000848E C43E[9E05]                 les     di,[THISSFT]          ; grab sft
00008492 31C0                       XOR     AX,AX
                                    ;mov    [es:di+51],ax
00008494 26894533                   MOV     [ES:DI+SF_ENTRY.sf_MFT],AX ; indicate free SFT
00008498 51                         push    cx
00008499 E8ABFF                     call    SHARE_CHECK             ; attempt to enter into the sharing set
0000849C 59                         pop     cx
0000849D 730A                       jnc     short done            ; success, let the user see this
0000849F E84293                     call    Idle                    ; wait a while
000084A2 E2EA                       loop    attempt                 ; go back for another attempt
000084A4 E8A5FF                     call    SHARE_VIOLATION         ; signal the problem to the user
000084A7 73E1                       jnc     short retry           ; user said to retry, go do it
                                done:
000084A9 59                         pop     cx
000084AA C3                         retn

                                ;============================================================================
                                ; EXEPATCH.ASM (MSDOS 6.0, 1991)
                                ;============================================================================
                                ; 29/04/2019 - Retro DOS 4.0
                                ; 20/03/2024 - Retro DOS 5.0

                                ;** EXEPATCH.ASM
                                ;-----------------------------------------------------------------------
                                ;   Contains the foll:
                                ;
                                ;           - code to find and overlay buggy unpack code
                                ;           - new code to be overlayed on buggy unpack code
                                ;           - old code sequence to identify buggy unpack code
                                ;
                                ;   Revision history:
                                ;
                                ;           Created: 5/14/90
                                ;-----------------------------------------------------------------------

                                ;-----------------------------------------------------------------------
                                ;
                                ; M020 : Fix for rational bug - for details see routine header
                                ; M028 : 4b04 implementation
                                ; M030 : Fixing bug in EXEPACKPATCH (EXEC_CS is an un-relocated value)
                                ; M032 : set turnoff bit only if DOS in HMA.
                                ; M033 : if IP < 2 then not exepacked.
                                ; M046 : support for a 4th version of exepacked files.
                                ; M068 : support for copy protected apps.
                                ; M071 : use A20OFF_COUNT of 10.
                                ;
                                ;-----------------------------------------------------------------------

                                PATCH1_COM_OFFSET  EQU     06CH
                                PATCH1_OFFSET              EQU     028H
                                PATCH1_CHKSUM              EQU     0EF4EH
                                CHKSUM1_LEN        EQU     11CH/2 ; 142

                                PATCH2_COM_OFFSET  EQU     076H
                                PATCH2_OFFSET              EQU     032H

                                    ; The strings that start at offset 076h have two possible
                                    ; check sums that are defined as PATCH2_CHKSUM PATCH2A_CHKSUM

                                PATCH2_CHKSUM              EQU     78B2H
                                CHKSUM2_LEN        EQU     119H/2
                                PATCH2A_CHKSUM             EQU     1C47H          ; M046
                                CHKSUM2A_LEN               EQU     103H/2         ; M046

                                PATCH3_COM_OFFSET  EQU     074H
                                PATCH3_OFFSET              EQU     032H
                                PATCH3_CHKSUM              EQU     4EDEH
                                CHKSUM3_LEN        EQU     117H/2

                                ;** Data structure passed for ExecReady call
                                ;
                                ;struc ERStruc
                                ; .ER_Reserved:    resw    1       ; reserved, should be zero
                                ; .ER_Flags:       resw    1
                                ; .ER_ProgName:    resd    1       ; ptr to ASCIIZ str of prog name
                                ; .ER_PSP:  resw    1       ; PSP of the program
                                ; .ER_StartAddr: resd      1       ; Start CS:IP of the program
                                ; .ER_ProgSize:    resd    1       ; Program size including PSP
                                ; .size:
                                ;endstruc

                                ;DOSCODE SEGMENT

                                    ; 22/05/2019 - Retro DOS v4.0
                                    ; DOSCODE:B3DDh (MSDOS 6.21, MSDOS.SYS)

                                    ; 04/12/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
                                    ; DOSCODE:B37Ah (MSDOS 5.0, MSDOS.SYS)

                                    ; 20/03/2024 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMDOS.COM)
                                    ; DOSCODE:C697h (PCDOS 7.1, IBMDOS.COM)

                                ; M028 - BEGIN

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

```
48597                                   ;   Procedure Name        : ExecReady
48598                                   ;
48599                                   ;   Input                 : DS:DX -> ERStruc (see exe.inc)
48600                                   ;
48601                                   ;-----------------------------------------------------------------------
48602
48603                                   ExecReady:
48604 000084AB 89D6                         mov     si,dx                   ; move the pointer into a friendly one
48605                                       ;;test  word [si+2],1
48606                                       ; 17/12/2022
48607 000084AD F6440201                     test    byte [si+ERStruc.ER_Flags],ER_EXE ; 1
48608                                       ;test   word [si+ERStruc.ER_Flags],ER_EXE ; COM or EXE ?
48609 000084B1 7418                         jz      short er_setver                 ; only setver for .COM files
48610
48611                                       ;mov    ax,[si+8]
48612 000084B3 8B4408                       mov     ax,[si+ERStruc.ER_PSP]
48613 000084B6 83C010                       add     ax,10h
48614 000084B9 8EC0                         mov     es,ax
48615
48616                                       ;mov    cx,[si+10]
48617 000084BB 8B4C0A                       mov     cx,[si+ERStruc.ER_StartAddr]    ; M030
48618                                       ;mov    ax,[si+12]      ; 11/04/2024
48619 000084BE 8B440C                       mov     ax,[si+ERStruc.ER_StartAddr+2] ; M030
48620
48621                                       ;call   [ss:FixExePatch]
48622 000084C1 36FF16[670D]                 call    word [ss:FixExePatch] ; 28/12/2022
48623
48624                                       ; 20/03/2024
48625                                       ; 04/12/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
48626 000084C6 36FF16[3E13]                 call    word [ss:Rational386PatchPtr]
48627
48628                                   er_setver:
48629                                       ;;test  word [si+2],2           ; Q: is this an overlay
48630                                       ; 17/12/2022
48631 000084CB F6440202                     test    byte [si+ERStruc.ER_Flags],ER_OVERLAY ; 2
48632                                       ;test   word [si+ERStruc.ER_Flags],ER_OVERLAY
48633 000084CF 7518                         jnz     short er_chkdoshi       ; Y: set A20OFF_COUNT if DOS high
48634                                                                       ; N: set up lie version first
48635 000084D1 1E                           push    ds
48636 000084D2 56                           push    si
48637                                       ;lds    si,[si+4]
48638 000084D3 C57404                       lds     si,[si+ERStruc.ER_ProgName]
48639 000084D6 E8DCEC                       call    Scan_Execname1
48640 000084D9 E8EDEC                       call    Scan_Special_Entries
48641 000084DC 5E                           pop     si
48642 000084DD 1F                           pop     ds
48643                                       ;mov    es,[si+8]
48644 000084DE 8E4408                       mov     es,[si+ERStruc.ER_PSP]
48645 000084E1 36A1[BB0E]                   mov     ax,[ss:SPECIAL_VERSION]
48646                                       ;mov    [es:40h],ax ; 11/04/2024
48647 000084E5 26A34000                     mov     [es:PDB.Version],ax
48648
48649                                   er_chkdoshi:
48650 000084E9 36803E[660D]00               cmp     byte [ss:DosHasHMA],0 ; M032: Q: is dos in HMA (M021)
48651 000084EF 741F                         je      short er_done           ; M032: N: done
48652
48653                                                                       ; M068 - Start
48654                                       ;mov    ax,[si+8]
48655 000084F1 8B4408                       mov     ax,[si+ERStruc.ER_PSP]; ax = PSP
48656
48657                                       ;or     byte [ss:DOS_FLAG],4
48658 000084F4 36800E[8600]04               or      byte [ss:DOS_FLAG],EXECA20OFF ; Set bit to signal int 21
48659                                                                       ; ah = 25 & ah= 49. See dossym.inc
48660                                                                       ; under TAG M003 & M009 for
48661                                                                       ; explanation
48662                                       ;;test  word [si+2],1
48663                                       ; 17/12/2022
48664 000084FA F6440201                     test    byte [si+ERStruc.ER_Flags],ER_EXE ; 1
48665                                       ;test   word [si+ERStruc.ER_Flags],ER_EXE ; Q: COM file
48666 000084FE 7507                         jnz     short er_setA20                 ; N: inc a20off_count, set
48667                                                                       ;    a20off_psp and ret
48668 00008500 1E                           push    ds
48669 00008501 8ED8                         mov     ds,ax                   ; DS = load segment of com file.
48670 00008503 E8AC05                       call    IsCopyProt              ; check if copy protected
48671 00008506 1F                           pop     ds
48672
48673                                   er_setA20:
48674                                       ; We need to inc the A20OFF_COUNT here. Note that if the count
48675                                       ; is non-zero at this point it indicates that the A20 is to be
48676                                       ; turned off for that many int 21 calls made by the app. In
48677                                       ; addition the A20 has to be turned off when we exit from this
48678                                       ; call. Hence the inc.
48679
48680 00008507 36FE06[8500]                  inc     byte [ss:A20OFF_COUNT]
48681 0000850C 36A3[6300]                   mov     [ss:A20OFF_PSP],ax      ; set the PSP for which A20 is to be
48682                                                                       ; turned OFF.
48683                                   er_done:                            ; M068 - End
48684 00008510 31C0                         xor     ax,ax
48685 00008512 C3                           retn
48686
48687                                   ; M028 - END
48688
48689                                   ; 20/03/2024 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMDOS.COM)
48690                                   ;%if 1
48691                                   ; 04/12/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
48692                                   ;%if 0
48693
48694                                   ;-----------------------------------------------------------------------
48695                                   ;
48696                                   ; procedure : Rational386Patch
48697                                   ;
48698                                   ; Older versions of the Rational DOS Extender have several bugs which trash
48699                                   ; 386 registers (usually just the high word of 32 bit registers) during
48700                                   ; interrupt processing. Lotus 123 3.1+ is a popular application that uses a
48701                                   ; version of the Rational extender with the 32 bit register trashing bugs.
48702                                   ;
48703                                   ; This routine applies patches to the Rational DOS Extender to work around
48704                                   ; most of the register trashing bugs.
48705                                   ;
48706                                   ; Note that there are additional register trashing bugs not fixed by these
48707                                   ; patches. In particular, the high word of ESP and the FS and GS registers
48708                                   ; may be modified on interrupts.
48709                                   ;
48710                                   ; There are two different Rational DOS Extender patchs in this module.
48711                                   ; Rational386Patch is to correct 386 register trashing bugs on 386 or later
48712                                   ; processors. This patch code is executed when MS-DOS is running on a 386
48713                                   ; or later processor, regardless of whether MS-DOS is running in the HMA
48714                                   ; or not.
48715                                   ;
48716                                   ; The other Rational patch routine (RationalPatch, below) fixes a register
48717                                   ; trashing problem on 286 processors, and is only executed if MS-DOS is
48718                                   ; running in the HMA.
48719                                   ;
48720                                   ; This patch detection and replacement is based on an example supplied by
```

```
                                ; Ben Williams at Rational.
                                ;
                                ;--------------------------------------------------------------------------
                                ; 22/05/2019 - Retro DOS v4.0
                                ; DOSCODE:B448h (MSDOS 6.21, MSDOS.SYS)
                                ;--------------------------------------------------------------------------
                                ;
                                ; INPUT : ES = segment where program got loaded
                                ;
                                ;--------------------------------------------------------------------------
                                rpFind1:
00008513 FAE4216033C0E6438B-        db      0FAh, 0E4h, 21h, 60h, 33h, 0C0h, 0E6h, 43h, 8Bh, 16h
0000851C 16
                                rpFind1Len equ     $ - rpFind1

                                ;   cli
                                ;   in      al, 21h
                                ;   pusha
                                ;   xor     ax, ax
                                ;   out     43h, al
                                ;   mov     dx, ...

                                rpFind1a:
0000851D B00EE63733C0E6F2           db      0B0h, 0Eh, 0E6h, 37h, 33h, 0C0h, 0E6h, 0F2h

                                rpFind1aLen equ    $ - rpFind1a

                                ;   mov     al, 0Eh
                                ;   out     37h, al
                                ;   xor     ax, ax
                                ;   out     0F2h, al

                                ; bug # 1 -- loss of high EAX on 386+ if not VCPI or DPMI

                                rpFind2:
00008525 0F20C0                     db      0Fh, 20h, 0C0h

                                rpFind2Len equ     $ - rpFind2

                                ;   mov     eax, cr0       ;may be preceeded by PUSH CX (51h)

                                rpFind3:
00008528 0F22C0EA                   db      0Fh, 22h, 0C0h, 0EAh

                                rpFind3Len equ     $ - rpFind3

                                ;   mov     cr0, eax       ;may be preceeded by POP CX (59h)
                                ;   jmp     far ptr xxx    ;change far ptr to go to replace3
                                ;   mov     ss, bx         ;8E D3 ... and come back at or after this

                                ; note, there is no rpRepl1 string

                                rpRepl2:
0000852C 6650510F20C0               db      66h, 50h, 51h, 0Fh, 20h, 0C0h

                                rpRepl2Len equ     $ - rpRepl2

                                ;   push    eax
                                ;   push    cx
                                ;   mov     eax, cr0

                                rpRepl3:
00008532 8ED3596658                 db      8Eh, 0D3h, 59h, 66h, 58h

                                rpRepl3Len equ     $ - rpRepl3

                                ;   mov     ss, bx
                                ;   pop     cx
                                ;   pop     eax

                                ; bug # 2 -- loss of high EAX and ESI on 386+ only if VCPI

                                rpFind4:
00008537 93588BCC                   db      93h, 58h, 8Bh, 0CCh

                                rpFind4Len equ     $ - rpFind4

                                ;   xchg    bx, ax
                                ;   pop     ax
                                ;   mov     cx, sp

                                rpFind5:
0000853B B80CDECD678BE1FFE3         db      0B8h, 0Ch, 0DEh, 0CDh, 67h, 8Bh, 0E1h, 0FFh, 0E3h

                                rpFind5Len equ     $ - rpFind5

                                ;   mov     ax, DE0Ch
                                ;   int     67h
                                ;   mov     sp, cx
                                ;   jmp     bx

                                rpRepl4:
00008544 93588BCC                   db      93h, 58h, 8Bh, 0CCh
00008548 2E66A3                     db      02Eh, 066h, 0A3h

                                rpRepl4o1Len equ $ - rpRepl4

0000854B 0000                       db      00h, 00h
0000854D 2E668936                   db      02Eh, 066h, 89h, 36h

                                rpRepl4o2Len equ $ - rpRepl4

00008551 0000                       db      00h, 00h

                                rpRepl4Len equ     $ - rpRepl4

                                ;   xchg    bx, ax
                                ;   pop     ax
                                ;   mov     cx, sp
                                ;   mov     dword ptr cs:[xxxx], eax
                                ;   mov     dword ptr cs:[xxxx], esi

                                rpRepl5:
00008553 8BE1                       db      8Bh, 0E1h
00008555 2E66A1                     db      2Eh, 66h, 0A1h

                                rpRepl5o1Len equ $ - rpRepl5

00008558 0000                       db      00h, 00h
0000855A 2E668B36                   db      2Eh, 66h, 8Bh, 36h
```

```
                                rpRepl5o2Len equ $ - rpRepl5
0000855E 0000                       db      00h, 00h
00008560 FFE3                       db      0FFh, 0E3h

                                rpRepl5Len equ     $ - rpRepl5

                                ;   mov     sp, cx
                                ;   mov     eax, dword ptr cs:[xxxx]
                                ;   mov     esi, dword ptr cs:[xxxx]
                                ;   jmp     bx

                                ; bug # 3 -- loss of high EAX, EBX, ECX, EDX on 386+ only if VCPI

                                rpFind6:
00008562 FA5251                     db      0FAh, 52h, 51h

                                rpFind6Len equ     $ - rpFind6

                                ;   cli
                                ;   push    dx
                                ;   push    cx

                                rpFind7a:
00008565 B80CDE6626FF1E             db      0B8h, 0Ch, 0DEh, 66h, 26h, 0FFh, 1Eh

                                rpFind7aLen equ    $ - rpFind7a

                                ;   mov     ax, 0DE0Ch
                                ;   call    fword ptr es:[xxxx]

                                rpFind7b:
0000856C 595A5B                     db      59h, 5Ah, 5Bh

                                rpFind7bLen equ    $ - rpFind7b

                                ;   pop     cx
                                ;   pop     dx
                                ;   pop     bx

                                rpRepl6:
0000856F FA66506653665166 52        db      0FAh, 66h, 50h, 66h, 53h, 66h, 51h, 66h, 52h

                                rpRepl6Len equ     $ - rpRepl6

                                ;   cli
                                ;   push    eax
                                ;   push    ebx
                                ;   push    ecx
                                ;   push    edx

                                rpRepl7:
00008578 665A6659665B6658 5B        db      66h, 5Ah, 66h, 59h, 66h, 5Bh, 66h, 58h, 5Bh

                                rpRepl7Len equ     $ - rpRepl7

                                ;   pop     edx
                                ;   pop     ecx
                                ;   pop     ebx
                                ;   pop     eax
                                ;   pop     bx

                                ; bug # 4 -- loss of high EAX and EBX on 386+ only if VCPI

                                rpFind8:
00008581 6006 1EB800008ED8          db      60h, 06h, 1Eh, 0B8h, 00h, 00h, 8Eh, 0D8h

                                rpFind8Len equ     $ - rpFind8

                                ;   pusha
                                ;   push    es
                                ;   push    ds
                                ;   mov     ax, dgroup     ;jump back to here from replace8
                                ;   mov     ds, ax

                                rpFind9 :
00008589 1F0761                     db      1Fh, 07h, 61h

                                rpFind9Len equ     $ - rpFind9

                                ;   pop     ds
                                ;   pop     es
                                ;   popa

                                rpRepl8:
0000858C 6660061E                   db      66h, 60h, 06h, 1Eh

                                rpRepl8Len equ     $ - rpRepl8

                                ;   pushad
                                ;   push    es
                                ;   push    ds

                                rpRepl9:
00008590 1F076661C3                 db      1Fh, 07h, 66h, 61h, 0C3h

                                rpRepl9Len equ     $ - rpRepl9

                                ;   pop     ds
                                ;   pop     es
                                ;   popad
                                ;   retn                    ;no need to jmp back to main-line

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

                                struc SearchPair
00000000 ????                    .sp_off1: resw 1  ; offset of 1st search string
00000002 ????                    .sp_len1: resw 1  ; length of 1st search string
00000004 ????                    .sp_off2: resw 1  ; 2nd string
00000006 ????                    .sp_len2: resw 1  ; 2nd string
00000008 ????                    .sp_diff: resw 1  ; max difference between offsets
                                 .size:
                                endstruc

                                ;rpBug1Strs SearchPair    <offset rpFind2, rpFind2Len, offset rpFind3, rpFind3Len, 20h>

                                rpBug1Strs:
00008595 [2585]                     dw      rpFind2
00008597 0300                       dw      rpFind2Len ; 3
00008599 [2885]                     dw      rpFind3
0000859B 0400                       dw      rpFind3Len ; 4
0000859D 2000                       dw      20h
```

```
                                ;rpBug2Strs SearchPair    <offset rpFind4, rpFind4Len, offset rpFind5, rpFind5Len, 80h>
                                
                                rpBug2Strs:
0000859F [3785]                     dw      rpFind4
000085A1 0400                       dw      rpFind4Len ; 4
000085A3 [3B85]                     dw      rpFind5
000085A5 0900                       dw      rpFind5Len ; 9
000085A7 8000                       dw      80h
                                
                                ;rpBug3Strs SearchPair    <offset rpFind6, rpFind6Len, offset rpFind7a, rpFind7aLen, 80h>
                                
                                rpBug3Strs:
000085A9 [6285]                     dw      rpFind6
000085AB 0300                       dw      rpFind6Len ; 3
000085AD [6585]                     dw      rpFind7a
000085AF 0700                       dw      rpFind7aLen ; 7
000085B1 8000                       dw      80h
                                
                                ;rpBug4Strs SearchPair    <offset rpFind8, 4, offset rpFind9, rpFind9Len, 80h>
                                
                                rpBug4Strs:
000085B3 [8185]                     dw      rpFind8
000085B5 0400                       dw      rpRepl8Len ; 4 ; 20/03/2024
000085B7 [8985]                     dw      rpFind9
000085B9 0300                       dw      rpFind9Len ; 3
000085BB 8000                       dw      80h
                                
                                ;--------------------------------------------------------------------------
                                
                                struc StackVars
00000000 ????                    .sv_wVersion:     resw 1         ; Rational extender version #
00000002 ????                    .sv_cbCodeSeg: resw 1            ; code seg size to scan
00000004 ????                    .sv_pPatch:       resw 1         ; offset of next avail patch byte
                                 .size:
                                endstruc
                                
                                ;--------------------------------------------------------------------------
                                ; 22/05/2019 - Retro DOS v4.0
                                
                                Rational386Patch:
                                    ; Do a few quick checks to see if this looks like a Rational
                                    ; Extended application. Hopefully this will quickly weed out
                                    ; most non Rational apps.
                                
000085BD 26813E00008B01             cmp     word [es:0],395             ; version number goes here - versions
000085C4 7322                       jae     short rp3QuickOut     ;   3.95+ don't need patching
                                
000085C6 26833E0C0020               cmp     word [es:0Ch],20h     ; always has this value here
000085CC 751A                       jne     short rp3QuickOut
                                
000085CE 50                         push    ax ; *
                                
000085CF B81800                     mov     ax,18h                ; extender has 18h at
000085D2 2639061800                 cmp     [es:24],ax            ;   offsets 24, 28, & 36
000085D7 750E                       jne     short rp3QO_ax
000085D9 2639061C00                 cmp     [es:28],ax
000085DE 7507                       jne     short rp3QO_ax
000085E0 2639062400                 cmp     [es:36],ax
000085E5 7402                       je      short rp3Maybe
                                rp3QO_ax:
000085E7 58                         pop     ax
                                rp3QuickOut:
000085E8 C3                         retn
                                
                                ; It might be the rational extender, do more extensive checking
                                
                                rp3Maybe:
000085E9 FC                         cld
                                
                                ; 20/03/2024
                                %if 0
                                    push    bx                    ; note ax pushed above
                                    push    cx
                                    push    dx
                                    push    si
                                    push    di
                                    push    es
                                    push    ds                    ; we use all of them
                                    push    bp
                                %else
                                    ; 20/03/2024 (PCDOS 7.1 IBMDOS.COM)
                                    ;;;
000085EA 60                         pusha
000085EB 06                         push    es
000085EC 1E                         push    ds
                                    ;;;
                                %endif
000085ED 83EC06                     sub     sp,StackVars.size  ; 6; make space for stack variables
000085F0 89E5                       mov     bp,sp
                                
000085F2 0E                         push    cs
000085F3 1F                         pop     ds
                                
000085F4 26A10000                   mov     ax,[es:0]             ; save version #
                                    ;mov    [bp+StackVars.sv_wVersion],ax
000085F8 894600                     mov     [bp],ax
                                                                  ; check that binary version # matches
000085FB E89901                     call    VerifyVersion         ;   ascii string
000085FE 7521                       jne     short rp3Exit_j
                                
                                ; Looks like this is it, find where to put the patch code.  The
                                ; patch will be located on top of Rational code specific to 80286
                                ; processors, so these patchs MUST NOT be applied if running on
                                ; an 80286 system.
                                
                                    ; Rational says the code to patch will never be beyond offset 46xxh
                                
00008600 B90045                     mov     cx,4500h              ; force search len to 4700h (searches
                                    ;mov    [bp+2],cx
00008603 894E02                     mov     [bp+StackVars.sv_cbCodeSeg],cx       ; start at offset 200h)
                                
00008606 268E062000                 mov     es,[es:20h]           ; es=code segment
                                
0000860B BE[1385]                   mov     si,rpFind1            ; string to find
0000860E BA0A00                     mov     dx,rpFind1Len ; 10    ; length to match
00008611 E86B01                     call    ScanCodeSeq           ; look for code seq
00008614 740E                       jz      short rpGotPatch
                                
                                ; According to Rational, some very old versions of the extender may not
                                ; have the find1 code sequence. If the find1 code wasn't found above,
                                ; try an alternative patch area which is on top of NEC 98xx switching code.
                                
00008616 BE[1D85]                   mov     si,rpFind1a
```

```
49092 00008619 BA0800                   mov     dx,rpFind1aLen  ;8
49093 0000861C E86001                   call    ScanCodeSeq
49094 0000861F 7403                     jz      short rpGotPatch
49095
49096                                rp3Exit_j:
49097 00008621 E9FE00                   jmp     rp3Exit
49098
49099                                ; Found the location to write patch code! DI = offset in code seg.
49100
49101                                rpGotPatch:
49102                                    ;mov    [bp+4],di
49103 00008624 897EFC                   mov     [bp-StackVars.sv_pPatch],di   ; save patch pointer
49104
49105                                ;--------------------------------------------------------------------------
49106                                ; Bug # 1 -- loss of high EAX on 386+ if not VCPI or DPMI
49107
49108                                    ;cmp    word [bp+0],381
49109                                    ;cmp    word [bp+StackVars.sv_wVersion],381 ; only need bug 1 if version
49110 00008627 817E007D01               cmp     word [bp],381
49111 0000862C 7348                     jae     short rpBug2                  ;   < 3.81
49112
49113 0000862E BB[9585]                 mov     bx,rpBug1Strs                 ; locate find2 & find3 code
49114 00008631 E8F600                   call    FindBadCode
49115 00008634 7240                     jc      short rpBug2
49116
49117                                ; si = rpFind2 offset, di = rpFind3 offset
49118
49119 00008636 57                       push    di
49120 00008637 89F7                     mov     di,si                         ; rpFind2 offset
49121 00008639 BA0300                   mov     dx,rpFind2Len ; 3
49122
49123 0000863C 26807DFF51               cmp     byte [es:di-1],51h            ; find2 preceeded by push cx?
49124 00008641 7502                     jne     short rp_no_cx
49125
49126 00008643 4F                       dec     di                            ;   yes, gobble up push cx too
49127 00008644 42                       inc     dx
49128                                rp_no_cx:
49129 00008645 BE[2C85]                 mov     si,rpRepl2                    ; patch out find2 sequence
49130 00008648 B90600                   mov     cx,rpRepl2Len  ; 6
49131 0000864B E80501                   call    GenPatch
49132
49133 0000864E 5F                       pop     di                            ; rpFind3 offset
49134 0000864F 26807DFF59               cmp     byte [es:di-1],59h            ; find3 preceeded by pop cx?
49135 00008654 7505                     jne     short rp_no_cx2
49136
49137 00008656 26C645FF90               mov     byte [es:di-1],90h            ;   yes, no-op it
49138                                rp_no_cx2:
49139                                    ;mov    ax,[bp+4]
49140 0000865B 8B4604                   mov     ax,[bp+StackVars.sv_pPatch]   ; change offset of far jmp
49141                                    ;mov    [es:di+4],ax
49142 0000865E 26894504                 mov     [es:di+rpFind3Len],ax         ;   to go to patch code
49143
49144 00008662 57                       push    di                            ; save find3 offset
49145 00008663 BE[3285]                 mov     si,rpRepl3                    ; copy repl3 to patch area
49146 00008666 B90500                   mov     cx,rpRepl3Len ; 5
49147 00008669 E8FB00                   call    CopyPatch
49148
49149 0000866C 5B                       pop     bx                            ; find3 offset
49150 0000866D 83C308                   add     bx,rpFind3Len+4        ; 8          ; skip over find3 and far jmp
49151 00008670 E80001                   call    GenJump                       ; jmp back from patch area
49152                                    ;mov    [bp+4],di
49153 00008673 897E04                   mov     [bp+StackVars.sv_pPatch], di  ;   to main-line, update patch
49154                                                                         ;   area pointer
49155
49156                                ;--------------------------------------------------------------------------
49157                                ; Bug # 2 -- loss of high regs on 386+ under VCPI only
49158
49159                                rpBug2:
49160 00008676 BB[9F85]                 mov     bx,rpBug2Strs                 ; locate find4 & find5 code
49161 00008679 E8AE00                   call    FindBadCode
49162 0000867C 7242                     jc      short rpBug3
49163
49164                                ; si = rpFind4 offset, di = rpFind5 offset
49165
49166                                    ;push   word [bp+4]
49167 0000867E FF7604                   push    word [bp+StackVars.sv_pPatch] ; save current patch pointer
49168                                                                         ;   (where repl4 goes)
49169 00008681 57                       push    di                            ; save find5 offset
49170
49171 00008682 89F7                     mov     di,si
49172 00008684 BA0400                   mov     dx,rpFind4Len ; 4
49173 00008687 BE[4485]                 mov     si,rpRepl4
49174 0000868A B90F00                   mov     cx,rpRepl4Len ; 15
49175 0000868D E8C300                   call    GenPatch                      ; patch out find4 code
49176
49177 00008690 5F                       pop     di                            ; find5 offset
49178 00008691 83C705                   add     di,5                          ; keep 5 bytes of find5 code
49179                                    ;mov    bx,[bp+4]
49180 00008694 8B5E04                   mov     bx,[bp+StackVars.sv_pPatch]   ; jump to patch area
49181 00008697 53                       push    bx                            ; save repl5 location
49182 00008698 E8D800                   call    GenJump
49183
49184 0000869B BE[5385]                 mov     si,rpRepl5                    ; copy repl5 code to patch
49185 0000869E B90F00                   mov     cx,rpRepl5Len  ; 15           ;   area -- it has a jmp bx
49186 000086A1 E8C300                   call    CopyPatch                     ;   so no need to jmp back to
49187                                                                         ;   main-line code
49188
49189                                ; patches have been made, now update the patch code to store/load dwords just
49190                                ; after the code in the patch area
49191
49192 000086A4 5F                       pop     di                            ; repl5 location
49193 000086A5 5E                       pop     si                            ; repl4 location
49194
49195                                    ;mov    ax,[bp+4]
49196 000086A6 8B4604                   mov     ax,[bp+StackVars.sv_pPatch]   ; (where dwords go)
49197
49198                                    ;mov    [es:si+7],ax
49199 000086A9 26894407                 mov     [es:si+rpRepl4o1Len],ax             ; offset for EAX
49200                                    ;mov    [es:di+5],ax
49201 000086AD 26894505                 mov     [es:di+rpRepl5o1Len],ax
49202 000086B1 83C004                   add     ax,4
49203                                    ;mov    [es:si+0Dh],ax
49204 000086B4 2689440D                 mov     [es:si+rpRepl4o2Len],ax             ; offset for ESI
49205                                    ;mov    [es:di+0Bh],ax
49206 000086B8 2689450B                 mov     [es:di+rpRepl5o2Len],ax
49207
49208                                    ;add    word [bp+4],8
49209 000086BC 83460408                 add     word [bp+StackVars.sv_pPatch],8     ; reserve space for 2 dwords in
49210                                                                         ; patch area
49211
49212                                ;--------------------------------------------------------------------------
49213                                ; Bug # 3 -- loss of high regs on 386+ under VCPI only
49214
49215                                rpBug3:
```

```
000086C0 BB[A985]                   mov     bx,rpBug3Strs               ; locate find6 & find7a code
000086C3 E86400                     call    FindBadCode
000086C6 722C                       jc      short rpBug4

                                    ;add    di,9
000086C8 83C709                     add     di,rpFind7aLen + 2          ; skip over offset in find7a
000086CB 56                         push    si                          ;   code and locate find7b
000086CC BE[6C85]                   mov     si,rpFind7b                 ;   sequence
000086CF BA0300                     mov     dx,rpFind7bLen ; 3
000086D2 E8AD00                     call    ScanCodeSeq_di
000086D5 5E                         pop     si
000086D6 751C                       jnz     short rpBug4

000086D8 57                         push    di                          ; save find7b code offset

000086D9 89F7                       mov     di,si
000086DB BA0300                     mov     dx,rpFind6Len ; 3
000086DE BE[6F85]                   mov     si,rpRepl6
000086E1 B90900                     mov     cx,rpRepl6Len ; 9
000086E4 E86C00                     call    GenPatch                    ; patch out find6 code

000086E7 5F                         pop     di
000086E8 BA0300                     mov     dx,rpFind7bLen ; 3
000086EB BE[7885]                   mov     si,rpRepl7
000086EE B90900                     mov     cx,rpRepl7Len ; 9
000086F1 E85F00                     call    GenPatch                    ; patch out find7b code

                                ;--------------------------------------------------------------------------
                                ; Bug # 4 -- loss of high regs on 386+ under VCPI only

                                rpBug4:
                                    ;cmp    word [bp+0],360
                                    ;cmp    word [bp+StackVars.sv_wVersion],360 ; only applies if
000086F4 817E006801                 cmp     word [bp],360
000086F9 7627                       jbe     short rp3Exit               ; version > 3.60 and < 3.95

000086FB BB[B385]                   mov     bx,rpBug4Strs               ; locate find8 & find9 code
000086FE E82900                     call    FindBadCode
00008701 721F                       jc      short rp3Exit

00008703 57                         push    di                          ; save find9 code offset

00008704 89F7                       mov     di,si
00008706 BA0300                     mov     dx,3
00008709 BE[8C85]                   mov     si,rpRepl8
0000870C B90400                     mov     cx,rpRepl8Len ; 4
0000870F E84100                     call    GenPatch                    ; patch out find8 code

00008712 5F                         pop     di                          ; find9 offset
                                    ;mov    bx,[bp+4]
00008713 8B5E04                     mov     bx,[bp+StackVars.sv_pPatch] ; patch find9 to jmp to
00008716 E85A00                     call    GenJump                     ;   patch area

00008719 BE[9085]                   mov     si,rpRepl9                  ; copy replacement code to
0000871C B90500                     mov     cx,rpRepl9Len ; 5           ;   patch area--it does a RET
0000871F E84500                     call    CopyPatch                   ;   so no jmp back to main-line

                                rp3Exit:
00008722 83C406                     add     sp,StackVars.size

                                ; 20/03/2024
                                %if 0
                                    pop     bp
                                    pop     ds
                                    pop     es
                                    pop     di
                                    pop     si
                                    pop     dx
                                    pop     cx
                                    pop     bx
                                %else
                                    ; 20/03/2024 (PCDOS 7.1 IBMDOS.COM)
                                    ;;;
00008725 1F                         pop     ds
00008726 07                         pop     es
00008727 61                         popa
                                    ;;;
                                %endif
00008728 58                         pop     ax ; *
00008729 C3                         retn

                                ;--------------------------------------------------------------------------
                                ;
                                ; FindBadCode
                                ;
                                ; Searches Rational code segment looking for a pair of find strings (all
                                ; patches have at least two find strings).
                                ;
                                ; Entry:
                                ;   ES    = code segment to search
                                ;   DS:BX = search pair structure for this search
                                ;   [bp].sv_cbCodeSeg = length of code seg to search
                                ;
                                ; Exit:
                                ;   CY flag clear if both strings found, and
                                ;   SI    = offset in ES of 1st string
                                ;   DI    = offset in ES of 2nd string
                                ;   CY set if either string not found, or strings too far apart
                                ;
                                ; Used:
                                ;   CX
                                ;
                                ;--------------------------------------------------------------------------

                                ;struc SearchPair
                                ; .sp_off1: resw 1 ; offset of 1st search string
                                ; .sp_len1: resw 1 ; length of 1st search string
                                ; .sp_off2: resw 1 ; 2nd string
                                ; .sp_len2: resw 1 ; 2nd string
                                ; .sp_diff: resw 1 ; max difference between offsets
                                ; .size:
                                ;endstruc

                                FindBadCode:
                                    ;mov    cx,[bp+2]
0000872A 8B4E02                     mov     cx,[bp+StackVars.sv_cbCodeSeg]      ; search length

0000872D 8B37                       mov     si,[bx] ; mov si,[bx+0]
                                    ;mov    si,[bx+Searchpair.sp_off1] ; ds:si -> search string

                                    ;mov    dx,[bx+2]
0000872F 8B5702                     mov     dx,[bx+SearchPair.sp_len1] ; dx = search len
00008732 E84A00                     call    ScanCodeSeq
00008735 751A                       jnz     short fbc_error             ; done if 1st not found
```

```
49340
49341 00008737 57                      push    di                      ; save 1st string offset
49342
49343                                  ;mov    si,[bx+4]
49344 00008738 8B7704                  mov     si,[bx+SearchPair.sp_off2]
49345                                  ;mov    dx,[bx+6]
49346 0000873B 8B5706                  mov     dx,[bx+SearchPair.sp_len2]
49347 0000873E E84100                  call    ScanCodeSeq_di          ; don't change flags after this!
49348
49349 00008741 5E                      pop     si                      ; restore 1st string offset
49350 00008742 750D                    jnz     short fbc_error
49351
49352 00008744 89F8                    mov     ax,di                   ; sanity check that
49353 00008746 29F0                    sub     ax,si                   ;   si < di && di - si <= allowed diff
49354 00008748 7207                    jc      short fbc_error
49355                                  ;cmp    ax,[bx+8]
49356 0000874A 3B4708                  cmp     ax,[bx+SearchPair.sp_diff]
49357 0000874D 7702                    ja      short fbc_error
49358
49359 0000874F F8                      clc
49360 00008750 C3                      retn
49361
49362                                fbc_error:
49363 00008751 F9                      stc
49364 00008752 C3                      retn
49365
49366                                ;--------------------------------------------------------------------------
49367                                ;
49368                                ; GenPatch
49369                                ;
49370                                ; Generate a patch sequence. 1) insert a jump at the buggy code location
49371                                ; (jumps to the patch code area), 2) copy the selected patch code to the
49372                                ; patch area, 3) insert a jump from the patch area back to the main-line
49373                                ; code.
49374                                ;
49375                                ; Entry:
49376                                ;   ES:DI = start of buggy code to be patched
49377                                ;   DX    = length of buggy code to be patched
49378                                ;   DS:SI = replacement patch code
49379                                ;   CX    = length of replacement patch code
49380                                ;   [bp].sv_pPatch = offset in ES of where to copy patch code
49381                                ;
49382                                ; Exit:
49383                                ;   DI, [bp].sv_pPatch = byte after generated patch code
49384                                ;
49385                                ; Used:
49386                                ;   AX, BX, SI, Flags
49387                                ;
49388                                ;--------------------------------------------------------------------------
49389
49390                                GenPatch:
49391 00008753 57                      push    di                      ;save offset of buggy code
49392
49393                                  ;mov    bx,[bp+4]
49394 00008754 8B5E04                  mov     bx,[bp+StackVars.sv_pPatch]
49395                                                                  ;jump from buggy code to patch area
49396 00008757 E81900                  call    GenJump
49397
49398 0000875A E80A00                  call    CopyPatch               ;copy replacement code to patch area
49399
49400 0000875D 5B                      pop     bx                      ;offset of buggy code + buggy code
49401 0000875E 01D3                    add     bx,dx                   ;  length = return from patch offset
49402
49403 00008760 E81000                  call    GenJump                 ;jump from patch area back to main-
49404                                  ;mov    [bp+4],di
49405 00008763 897E04                  mov     [bp+StackVars.sv_pPatch],di
49406                                                                  ;  line code, update patch pointer
49407 00008766 C3                      retn
49408
49409                                ;--------------------------------------------------------------------------
49410                                ;
49411                                ; CopyPatch
49412                                ;
49413                                ; Copies patch code to patch location.
49414                                ;
49415                                ; Entry:
49416                                ;   DS:SI = patch code to be copied
49417                                ;   ES    = segment of code to patch
49418                                ;   CX    = length of code to copy
49419                                ;   [bp].sv_pPatch = offset in ES of where to copy patch code
49420                                ;
49421                                ; Exit:
49422                                ;   DI, [bp].sv_pPatch = byte after copied patch code
49423                                ;
49424                                ; Used:
49425                                ;   SI, Flags
49426                                ;
49427                                ;--------------------------------------------------------------------------
49428
49429                                CopyPatch:
49430 00008767 51                      push    cx
49431                                  ;mov    di,[bp+4]
49432 00008768 8B7E04                  mov     di,[bp+StackVars.sv_pPatch] ;patch pointer is the dest offset
49433 0000876B FC                      cld
49434 0000876C F3A4                    rep movsb
49435 0000876E 59                      pop     cx
49436                                  ;mov    [bp+4],di
49437 0000876F 897E04                  mov     [bp+StackVars.sv_pPatch],di ;update net pointer location
49438 00008772 C3                      retn
49439
49440                                ;--------------------------------------------------------------------------
49441                                ;
49442                                ; GenJump
49443                                ;
49444                                ; Generates a rel16 JMP instruction at location 'from' to location 'to'.
49445                                ;
49446                                ; Entry:
49447                                ;   ES:DI = from location (where to put jmp instruction)
49448                                ;   BX    = to location (where to jump to)
49449                                ;
49450                                ; Exit:
49451                                ;   DI = byte after generated jump
49452                                ;
49453                                ; Used:
49454                                ;   AX
49455                                ;
49456                                ;--------------------------------------------------------------------------
49457
49458                                GenJump:
49459 00008773 B0E9                    mov     al,0E9h         ; jmp rel16 opcode
49460 00008775 AA                      stosb
49461
49462 00008776 89D8                    mov     ax,bx           ; calc offset to 'to' location
49463 00008778 29F8                    sub     ax,di
```

```
                        sub     ax,2

                        stosw                   ; output offset

                        retn

                    ;--------------------------------------------------------------------------
                    ;
                    ; ScanCodeSeq
                    ;
                    ; Looks for a pattern pointed to by DS:SI & len DX in ES:200 to ES:200+CX-1
                    ;
                    ; returns in ES:DI the start of the pattern if Zero flag is set
                    ;
                    ;--------------------------------------------------------------------------

                    ScanCodeSeq:
                        mov     di,200h
                    ScanCodeSeq_di:
                        push    cx
                        sub     cx,dx
                        inc     cx
                    scsagain:
                        push    si
                        push    di
                        push    cx
                        mov     cx,dx
                        rep     cmpsb
                        pop     cx
                        pop     di
                        pop     si
                        je      short scsfound
                        inc     di
                        loop    scsagain
                    scsfound:
                        pop     cx
                    vvexit:             ; 18/12/2022
                        retn

                    ;--------------------------------------------------------------------------
                    ;
                    ; VerifyVersion
                    ;
                    ; Checks whether the binary version from ES:0 matches the ASCII version
                    ; from ES:2A.
                    ;
                    ;       Entry: AX = binary version number
                    ;       Exit : Z flag set if version numbers match
                    ;
                    ;--------------------------------------------------------------------------

                    VerifyVersion:
                        mov     si,[es:2Ah]             ; offset of version number
                                                        ;  in ascii
                        mov     bl,10
                        add     si,3                    ; point to last digit

                        call    VVDigit
                        jne     short vvexit
                        call    VVDigit
                        jne     short vvexit
                        cmp     byte [es:si],'.'
                        jne     short vvexit
                        dec     si
                        ;call   VVDigit
                        ; 18/12/2022
                        ;jmp    short VVDigit
                    ;vvexit:
                        ;retn
                    VVDigit:
                        div     bl
                        add     ah,'0'
                        dec     si
                        cmp     [es:si+1],ah
                        mov     ah,0                    ; do not xor or sub we need Z
                        retn

                    ; 20/03/2024 - Retro DOS v5.0 (PCDOS 7.1 IBMDOS.COM)
                    ;%endif

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

                    ; 23/05/2019 - Retro DOS v4.0
                    ; DOSCODE:B702h (MSDOS 6.21, MSDOS.SYS)

                    ; 04/12/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
                    ; DOSCODE:B3E0h (MSDOS 5.0, MSDOS.SYS)

                    exepatch_start:     ; label byte

                        ; The following is the code that'll be layed over the buggy unpack
                        ; code.
                    str1:
                        db  06h                 ;push   es
                        db  8Ch,0D8h            ;mov    ax,ds

                    first_stop equ     $-str1

                        db  2Bh, 0C2h           ;sub    ax,dx

                    first:  ; label    byte

                        db  8Eh,0D8h            ;mov    ds,ax
                        db  8Eh,0C0h            ;mov    es,ax
                        db  0BFh,0Fh,00h        ;mov    di,000FH
                        db  57h                 ;push   di
                        db  0B9h,10h,00h        ;mov    cx,0010H
                        db  0B0h,0FFh           ;mov    al,0FFH
                        db  0F3h,0AEh           ;repz   scasb
                        db  47h                 ;inc    di
                        db  8Bh,0F7h            ;mov    si,di
                        db  5Fh                 ;pop    di
                        db  58h                 ;pop    ax

                    second_stop equ    $-first

                        db  2Bh,0C2h            ;sub    ax,dx

                    second: ; label    byte

                        db  8Eh,0C0h            ;mov    es,ax
                                                ;NextRec:
                        db  0B9h,04h,02h        ;mov    cx,0204h
                                                ;norm_agn:
```

```
49588 000087DF 8BC6                       db  8Bh,0C6h           ;mov    ax,si
49589 000087E1 F7D0                       db  0F7h,0D0h         ;not    ax
49590 000087E3 D3E8                       db  0D3h,0E8h         ;shr    ax,cl
49591 000087E5 7413                       db  74h,13h           ;jz     short SI_ok
49592 000087E7 8CDA                       db  8Ch,0DAh          ;mov    dx,ds
49593 000087E9 83CEF0                     db  83h,0CEh,0F0h     ;or     si,0FFF0H
49594 000087EC 2BD0                       db  2Bh,0D0h          ;sub    dx,ax
49595 000087EE 7308                       db  73h,08h           ;jnc    short SItoDS
49596 000087F0 F7DA                       db  0F7h,0DAh         ;neg    dx
49597 000087F2 D3E2                       db  0D3h,0E2h         ;shl    dx,cl
49598 000087F4 2BF2                       db  2Bh,0F2h          ;sub    si,dx
49599 000087F6 33D2                       db  33h,0D2h          ;xor    dx,dx
49600                                                           ;SItoDS:
49601 000087F8 8EDA                       db  8Eh,0DAh          ;mov    ds,dx
49602                                                           ;SI_ok:
49603 000087FA 87F7                       db  87h,0F7h          ;xchg   si,di
49604 000087FC 1E                         db  1Eh               ;push   ds
49605 000087FD 06                         db  06h               ;push   es
49606 000087FE 1F                         db  1Fh               ;pop    ds
49607 000087FF 07                         db  07h               ;pop    es
49608 00008800 FECD                       db  0FEh,0CDh         ;dec    ch
49609 00008802 75DB                       db  75h,0DBh          ;jnz    short norm_agn
49610 00008804 AC                         db  0ACh              ;lodsb
49611 00008805 92                         db  92h               ;xchg   dx,ax
49612 00008806 4E                         db  4Eh               ;dec    si
49613 00008807 AD                         db  0ADh              ;lodsw
49614 00008808 8BC8                       db  8Bh,0C8h          ;mov    cx,ax
49615 0000880A 46                         db  46h               ;inc    si
49616 0000880B 8AC2                       db  8Ah,0C2h          ;mov    al,dl
49617 0000880D 24FE                       db  24h,0FEh          ;and    al,0FEH
49618 0000880F 3CB0                       db  3Ch,0B0h          ;cmp    al,RPTREC
49619 00008811 7505                       db  75h,05h           ;jne    short TryEnum
49620 00008813 AC                         db  0ACh              ;lodsb
49621 00008814 F3AA                       db  0F3h,0AAh         ;rep stosb
49622
49623                               ;       db  0EBh,07h,90h      ;jmp    short TryNext
49624 00008816 EB06                       db  0EBh,06h          ;jmp    short TryNext
49625
49626                                                           ;TryEnum:
49627 00008818 3CB2                       db  3Ch,0B2h          ;cmp    al,ENMREC
49628 0000881A 756C                       db  75h,6Ch           ;jne    short CorruptExe
49629 0000881C F3A4                       db  0F3h,0A4h         ;rep movsb
49630                                                           ;TryNext:
49631
49632 0000881E 92                         db  92h               ;xchg   dx,ax
49633                               ;       db  8Ah,0C2h          ;mov    al,dl
49634
49635 0000881F A801                       db  0A8h,01h          ;test   al,1
49636 00008821 74B9                       db  74h,0B9h          ;jz     short NextRec
49637 00008823 9090                       db  90h,90h           ;nop,nop
49638
49639                                   last_stop equ $-second
49640                                   size_str1 equ $-str1
49641
49642                                       ; The following is the code that we need to look for in the exe
49643                                       ; file.
49644
49645                                   scan_patch1: ; label byte
49646
49647 00008825 8CC3                       db  8Ch,0C3h          ;mov    bx,es
49648 00008827 8CD8                       db  8Ch,0D8h          ;mov    ax,ds
49649 00008829 2BC2                       db  2Bh,0C2h          ;sub    ax,dx
49650 0000882B 8ED8                       db  8Eh,0D8h          ;mov    ds,ax
49651 0000882D 8EC0                       db  8Eh,0C0h          ;mov    es,ax
49652 0000882F BF0F00                     db  0BFh,0Fh,00h      ;mov    di,000FH
49653 00008832 B91000                     db  0B9h,10h,00h      ;mov    cx,0010H
49654 00008835 B0FF                       db  0B0h,0FFh         ;mov    al,0FFH
49655 00008837 F3AE                       db  0F3h,0AEh         ;repz   scasb
49656 00008839 47                         db  47h               ;inc    di
49657 0000883A 8BF7                       db  8Bh,0F7h          ;mov    si,di
49658 0000883C 8BC3                       db  8Bh,0C3h          ;mov    ax,bx
49659 0000883E 2BC2                       db  2Bh,0C2h          ;sub    ax,dx
49660 00008840 8EC0                       db  8Eh,0C0h          ;mov    es,ax
49661 00008842 BF0F00                     db  0BFh,0Fh,00h      ;mov    di,000FH
49662                                                           ;NextRec:
49663 00008845 B104                       db  0B1h,04h          ;mov    cl,4
49664 00008847 8BC6                       db  8Bh,0C6h          ;mov    ax,si
49665 00008849 F7D0                       db  0F7h,0D0h         ;not    ax
49666 0000884B D3E8                       db  0D3h,0E8h         ;shr    ax,cl
49667 0000884D 7409                       db  74h,09h           ;jz     short SI_ok
49668 0000884F 8CDA                       db  8Ch,0DAh          ;mov    dx,ds
49669 00008851 2BD0                       db  2Bh,0D0h          ;sub    dx,ax
49670 00008853 8EDA                       db  8Eh,0DAh          ;mov    ds,dx
49671 00008855 83CEF0                     db  83h,0CEh,0F0h     ;or     si,0FFF0H
49672                                                           ;SI_ok:
49673 00008858 8BC7                       db  8Bh,0C7h          ;mov    ax,di
49674 0000885A F7D0                       db  0F7h,0D0h         ;not    ax
49675 0000885C D3E8                       db  0D3h,0E8h         ;shr    ax,cl
49676 0000885E 7409                       db  74h,09h           ;jz     short DI_ok
49677 00008860 8CC2                       db  8Ch,0C2h          ;mov    dx,es
49678 00008862 2BD0                       db  2Bh,0D0h          ;sub    dx,ax
49679 00008864 8EC2                       db  8Eh,0C2h          ;mov    es,dx
49680 00008866 83CFF0                     db  83h,0CFh,0F0h     ;or     di,0FFF0H
49681                                                           ;DI_ok:
49682
49683                                   size_scan_patch1 equ $-scan_patch1
49684
49685                                   scan_patch2: ; label byte
49686
49687 00008869 8CC3                       db  8Ch,0C3h          ;mov    bx,es
49688 0000886B 8CD8                       db  8Ch,0D8h          ;mov    ax,ds
49689 0000886D 48                         db  48h               ;dec    ax
49690 0000886E 8ED8                       db  8Eh,0D8h          ;mov    ds,ax
49691 00008870 8EC0                       db  8Eh,0C0h          ;mov    es,ax
49692 00008872 BF0F00                     db  0BFh,0Fh,00h      ;mov    di,000FH
49693 00008875 B91000                     db  0B9h,10h,00h      ;mov    cx,0010H
49694 00008878 B0FF                       db  0B0h,0FFh         ;mov    al,0FFH
49695 0000887A F3AE                       db  0F3h,0AEh         ;repz   scasb
49696 0000887C 47                         db  47h               ;inc    di
49697 0000887D 8BF7                       db  8Bh,0F7h          ;mov    si,di
49698 0000887F 8BC3                       db  8Bh,0C3h          ;mov    ax,bx
49699 00008881 48                         db  48h               ;dec    ax
49700 00008882 8EC0                       db  8Eh,0C0h          ;mov    es,ax
49701 00008884 BF0F00                     db  0BFh,0Fh,00h      ;mov    di,000FH
49702                                                           ;NextRec:
49703 00008887 B104                       db  0B1h,04h          ;mov    cl,4
49704 00008889 8BC6                       db  8Bh,0C6h          ;mov    ax,si
49705 0000888B F7D0                       db  0F7h,0D0h         ;not    ax
49706 0000888D D3E8                       db  0D3h,0E8h         ;shr    ax,cl
49707 0000888F 740A                       db  74h,0Ah           ;jz     short SI_ok
49708 00008891 8CDA                       db  8Ch,0DAh          ;mov    dx,ds
49709 00008893 2BD0                       db  2Bh,0D0h          ;sub    dx,ax
49710 00008895 8EDA                       db  8Eh,0DAh          ;mov    ds,dx
49711 00008897 81CEF0FF                   db  81h,0CEh,0F0h,0FFh
```

```
                                                        ;or     si,0FFF0H
                                                        ;SI_ok:
0000889B 8BC7                           db  8Bh,0C7h    ;mov    ax,di
0000889D F7D0                           db  0F7h,0D0h   ;not    ax
0000889F D3E8                           db  0D3h,0E8h   ;shr    ax,cl
000088A1 740A                           db  74h,0Ah     ;jz     short DI_ok
000088A3 8CC2                           db  8Ch,0C2h    ;mov    dx,es
000088A5 2BD0                           db  2Bh,0D0h    ;sub    dx,ax
000088A7 8EC2                           db  8Eh,0C2h    ;mov    es,dx
000088A9 81CFF0FF                       db  81h,0CFh,0F0h,0FFh
                                                        ;or     di,0FFF0H
                                                        ;DI_ok:

                                    size_scan_patch2 equ $-scan_patch2

                                    scan_patch3: ; label byte

000088AD 8CC3                           db  8Ch,0C3h    ;mov    bx,es
000088AF 8CD8                           db  8Ch,0D8h    ;mov    ax,ds
000088B1 48                             db  48h         ;dec    ax
000088B2 8ED8                           db  8Eh,0D8h    ;mov    ds,ax
000088B4 8EC0                           db  8Eh,0C0h    ;mov    es,ax
000088B6 BF0F00                         db  0BFh,0Fh,00h        ;mov    di,000FH
000088B9 B91000                         db  0B9h,10h,00h        ;mov    cx,0010H
000088BC B0FF                           db  0B0h,0FFh   ;mov    al,0FFH
000088BE F3AE                           db  0F3h,0AEh   ;repz   scasb
000088C0 47                             db  47h         ;inc    di
000088C1 8BF7                           db  8Bh,0F7h    ;mov    si,di
000088C3 8BC3                           db  8Bh,0C3h    ;mov    ax,bx
000088C5 48                             db  48h         ;dec    ax
000088C6 8EC0                           db  8Eh,0C0h    ;mov    es,ax
000088C8 BF0F00                         db  0BFh,0Fh,00h        ;mov    di,000FH
                                                        ;NextRec:
000088CB B104                           db  0B1h,04h    ;mov    cl,4
000088CD 8BC6                           db  8Bh,0C6h    ;mov    ax,si
000088CF F7D0                           db  0F7h,0D0h   ;not    ax
000088D1 D3E8                           db  0D3h,0E8h   ;shr    ax,cl
000088D3 7409                           db  74h,09h     ;jz     short SI_ok
000088D5 8CDA                           db  8Ch,0DAh    ;mov    dx,ds
000088D7 2BD0                           db  2Bh,0D0h    ;sub    dx,ax
000088D9 8EDA                           db  8Eh,0DAh    ;mov    ds,dx
000088DB 83CEF0                         db  83h,0CEh,0F0h       ;or     si,0FFF0H
                                                        ;SI_ok:
000088DE 8BC7                           db  8Bh,0C7h    ;mov    ax,di
000088E0 F7D0                           db  0F7h,0D0h   ;not    ax
000088E2 D3E8                           db  0D3h,0E8h   ;shr    ax,cl
000088E4 7409                           db  74h,09h     ;jz     short DI_ok
000088E6 8CC2                           db  8Ch,0C2h    ;mov    dx,es
000088E8 2BD0                           db  2Bh,0D0h    ;sub    dx,ax
000088EA 8EC2                           db  8Eh,0C2h    ;mov    es,dx
000088EC 83CFF0                         db  83h,0CFh,0F0h       ;or     di,0FFF0H
                                                        ;DI_ok:

                                    size_scan_patch3 equ $-scan_patch3

                                    scan_com: ; label byte

000088EF AC                             db  0ACh        ;lodsb
000088F0 8AD0                           db  8Ah,0D0h    ;mov    dl,al
000088F2 4E                             db  4Eh         ;dec    si
000088F3 AD                             db  0ADh        ;lodsw
000088F4 8BC8                           db  8Bh,0C8h    ;mov    cx,ax
000088F6 46                             db  46h         ;inc    si
000088F7 8AC2                           db  8Ah,0C2h    ;mov    al,dl
000088F9 24FE                           db  24h,0FEh    ;and    al,0FEH
000088FB 3CB0                           db  3Ch,0B0h    ;cmp    al,RPTREC
000088FD 7506                           db  75h,06h     ;jne    short TryEnum
000088FF AC                             db  0ACh        ;lodsb
00008900 F3AA                           db  0F3h,0AAh   ;rep stosb
00008902 EB0790                         db  0EBh,07h,90h        ;jmp    short TryNext
                                                        ;TryEnum:
00008905 3CB2                           db  3Ch,0B2h    ;cmp    al,ENMREC
00008907 756B                           db  75h,6Bh     ;jne    short CorruptExe
00008909 F3A4                           db  0F3h,0A4h   ;rep movsb
                                                        ;TryNext:
0000890B 8AC2                           db  8Ah,0C2h    ;mov    al,dl
0000890D A801                           db  0A8h,01h    ;test   al,1
                                    ;   db  74h,0BAh    ;jz     short NextRec

                                    size_scan_com       equ     $-scan_com

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

                                    ; 23/05/2019 - Retro DOS v4.0
                                    ; DOSCODE:B852h (MSDOS 6.21, MSDOS.SYS)

                                    ; 04/12/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
                                    ; DOSCODE:B530h (MSDOS 5.0, MSDOS.SYS)

                                    ; 21/03/2024 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMDOS.COM)
                                    ; DOSCODE:CB02h (PCDOS 7.1 IBMDOS.COM)

                                    ExePatch:
                                        ; 21/03/2024 - Retro DOS v5.0
                                        ;;28/12/2022 - Retro DOS v4.1
0000890F E80600                         call    ExePackPatch
00008912 36FF16[690D]                   call    word [ss:RationalPatchPtr]
00008917 C3                             retn
                                        ; 28/12/2022
                                        ;jmp    short ExePackPatch

                                    ;--------------------------------------------------------------------------
                                    ;
                                    ; Procedure Name   : ExePackPatch
                                    ;
                                    ; Inputs           : DS                    -> DOSDATA
                                    ;                    ES:0                  -> read in image
                                    ;                    ax:cx = start cs:ip of program
                                    ; Output           :
                                    ;
                                    ;   1. If ES <= 0fffh
                                    ;      2. if exepack signature ('RB') found
                                    ;         3. if common code to patch compares (for 3 diff. versions)
                                    ;            4. if rest of the code & checksum compares
                                    ;               5. overlay buggy code with code in
                                    ;                  doscode:str1.
                                    ;            6. endif
                                    ;         7. endif
                                    ;      8. endif
                                    ;   9. endif
                                    ;
                                    ;
                                    ; Uses                    : NONE
                                    ;
```

```
                                ;--------------------------------------------------------------------------
                                
                                    ; 21/03/2024 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMDOS.COM)
                                    ; 04/12/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
                                    ; 23/05/2019 - Retro DOS v4.0
                                ExePackPatch:
00008918 53                         push    bx
00008919 8CC3                       mov     bx,es                   ; bx has load segment
0000891B 81FBFF0F                   cmp     bx,0FFFh                ; Q: is the load segment > 64K
0000891F 7602                       jbe     short ep_cont           ; N:
00008921 5B                         pop     bx                      ; Y: no need to patch
00008922 C3                         retn
                                ep_cont:
00008923 1E                         push    ds
00008924 06                         push    es
00008925 50                         push    ax
00008926 51                         push    cx
00008927 56                         push    si
00008928 57                         push    di
                                
                                            ; M033 - start
                                            ; exepacked programs have an IP of 12h (>=2)
                                
00008929 83E902                     sub     cx,2                    ; Q: is IP >=2
0000892C 7303                       jnb     short epp_1             ; N: exit
0000892E E9B500                     jmp     ep_notpacked
                                                                    ; ax:cx now points to location of
                                                                    ; 'RB' if this is an exepacked file.
                                            ; M033 - end
                                epp_1:
00008931 89CF                       mov     di,cx
00008933 8EC0                       mov     es,ax
00008935 36893E[8700]               mov     [ss:UNPACK_OFFSET],di ; save pointer to 'RB' in
                                                                  ; unpack_offset
                                
0000893A 26813D5242                 cmp     word [es:di],'RB' ; 4252h
                                    ;jne    ep_notpacked
0000893F 7403                       je      short epp_2
00008941 E9A200                     jmp     ep_notpacked
                                epp_2:
00008944 0E                         push    cs
00008945 1F                         pop     ds                      ; set ds to cs
                                
                                    ;add    di,6Ch
00008946 83C76C                     add     di,PATCH1_COM_OFFSET    ; es:di -> points to place in packed
                                                                    ;          file where we hope to find
                                                                    ;          scan string.
                                
00008949 E8A200                     call    chk_common_str          ; check for match
                                
0000894C 7521                       jnz     short ep_chkpatch2      ; Q: does the patch match
                                                                    ; N: check at patch2_offset
                                                                    ; Y: check for rest of patch string
0000894E BE[2588]                   mov     si,scan_patch1
                                                                    ; ds:si -> scan string
00008951 368B3E[8700]               mov     di,[ss:UNPACK_OFFSET] ; restore di to point to 'RB'
                                
                                    ;add    di,28h
                                    ; 07/12/2022
00008956 83C728                     add     di,PATCH1_OFFSET        ; es:di -> points to place in packed
                                                                    ;          file where we hope to find
                                                                    ;          scan string.
                                    ;;mov   cx,68
                                    ;mov    cx,size_scan_patch1
                                    ; 21/03/2024
00008959 B144                       mov     cl,size_scan_patch1 ; 68
                                
                                    ;mov    bx,142
0000895B BB8E00                     mov     bx,CHKSUM1_LEN
                                    ;mov    ax,0EF4Eh
0000895E B84EEF                     mov     ax,PATCH1_CHKSUM
00008961 E89E00                     call    chk_patchsum            ; check if patch and chk sum compare
00008964 7207                       jc      short ep_done1          ; Q: did we pass the test
                                                                    ; N: exit
                                                                    ; Y: overlay code with new
00008966 BE[BF87]                   mov     si,str1
                                    ;;mov   cx,102
                                    ;mov    cx,size_str1
                                    ; 21/03/2024
00008969 B166                       mov     cl,size_str1 ; 102
                                
0000896B F3A4                       rep     movsb
                                ep_done1:
0000896D EB77                       jmp     short ep_done ; 21/03/2024
                                
                                ep_chkpatch2:
                                    ;mov    di,76h
0000896F BF7600                     mov     di,PATCH2_COM_OFFSET    ; es:di -> possible location of patch
                                                                    ; in another version of unpack
00008972 E87900                     call    chk_common_str          ; check for match
                                
00008975 753D                       jnz     short ep_chkpatch3      ; Q: does the patch match
                                                                    ; N: check for patch3_offset
                                                                    ; Y: check for rest of patch string
                                
00008977 BE[6988]                   mov     si,scan_patch2
                                                                    ; ds:si -> scan string
                                    ;mov    di,32h
0000897A BF3200                     mov     di,PATCH2_OFFSET        ; es:di -> points to place in packed
                                                                    ;          file where we hope to find
                                    ;;mov   cx,68                   ;          scan string.
                                    ;mov    cx,size_scan_patch2
                                    ; 21/03/2024
0000897D B144                       mov     cl,size_scan_patch2 ; 68
                                    ;mov    bx,140
0000897F BB8C00                     mov     bx,CHKSUM2_LEN
                                    ;mov    ax,78B2h
00008982 B8B278                     mov     ax,PATCH2_CHKSUM
00008985 E87A00                     call    chk_patchsum            ; check if patch and chk sum compare
                                
                                                                    ; M046 - Start
                                                                    ; Q: did we pass the test
00008988 7310                       jnc     short ep_patchcode2     ; Y: overlay code with new
                                                                    ; N: try with a different chksum
                                
0000898A BE[6988]                   mov     si,scan_patch2
                                                                    ; ds:si -> scan string
                                    ;;mov   cx,68
                                    ;mov    cx,size_scan_patch2
                                    ; 21/03/2024
0000898D B144                       mov     cl,size_scan_patch2 ; 68
                                    ;mov    bx,129
0000898F BB8100                     mov     bx,CHKSUM2A_LEN
                                    ;mov    ax,1C47h
```

```
49960 00008992 B8471C                   mov     ax,PATCH2A_CHKSUM
49961 00008995 E86A00                   call    chk_patchsum            ; check if patch and chk sum compare
49962                                                                   ; Q: did we pass the test
49963 00008998 724C                     jc      short ep_notpacked      ; N: try with a different chksum
49964                                                                   ; Y: overlay code with new
49965
49966                                   ep_patchcode2:                          ; M046 - End
49967 0000899A BE[BF87]                 mov     si,str1
49968                                   ;;mov   cx,3
49969                                   ;mov    cx,first_stop
49970                                   ; 21/03/2024
49971 0000899D B103                     mov     cl,first_stop ; 3
49972 0000899F F3A4                     rep     movsb
49973 000089A1 B89048                   mov     ax,4890h                ; ax = opcodes for dec ax, nop
49974 000089A4 AB                       stosw
49975                                   ;add    si,2
49976                                   ; 21/03/2024
49977 000089A5 46                       inc     si
49978 000089A6 46                       inc     si
49979                                   ;;mov   cx,20
49980                                   ;mov    cx,second_stop
49981                                   ; 21/03/2024
49982 000089A7 B114                     mov     cl,second_stop ; 20
49983 000089A9 F3A4                     rep     movsb
49984 000089AB AB                       stosw                           ; put in dec ax and nop
49985                                   ;add    si,2
49986                                   ; 21/03/2024
49987 000089AC 46                       inc     si
49988 000089AD 46                       inc     si
49989                                   ;;mov   cx,75
49990                                   ;mov    cx,last_stop
49991                                   ; 21/03/2024
49992 000089AE B14B                     mov     cl,last_stop ; 75
49993 000089B0 F3A4                     rep     movsb
49994 000089B2 EB32                     jmp     short ep_done
49995
49996                                   ep_chkpatch3:
49997                                   ;mov    di,74h
49998 000089B4 BF7400                   mov     di,PATCH3_COM_OFFSET    ; es:di -> possible location of patch
49999                                                                   ; in another version of unpack
50000 000089B7 E83400                   call    chk_common_str          ; check for match
50001
50002 000089BA 752A                     jnz     short ep_notpacked      ; Q: does the patch match
50003                                                                   ; N: exit
50004                                                                   ; Y: check for rest of patch string
50005 000089BC BE[AD88]                 mov     si,scan_patch3
50006                                                                   ; ds:si -> scan string
50007                                   ;mov    di,32h
50008 000089BF BF3200                   mov     di,PATCH3_OFFSET        ; es:di -> points to place in packed
50009                                                                   ;          file where we hope to find
50010                                                                   ;         scan string.
50011                                   ;;mov   cx,66
50012                                   ;mov    cx,size_scan_patch3
50013                                   ; 21/03/2024
50014 000089C2 B142                     mov     cl,size_scan_patch3 ; 66
50015                                   ;mov    bx,139
50016 000089C4 BB8B00                   mov     bx,CHKSUM3_LEN
50017                                   ;mov    ax,4EDEh
50018 000089C7 B8DE4E                   mov     ax,PATCH3_CHKSUM
50019 000089CA E83500                   call    chk_patchsum            ; check if patch and chk sum compare
50020 000089CD 7217                     jc      short ep_notpacked      ; Q: did we pass the test
50021                                                                   ; N: exit
50022                                                                   ; Y: overlay code with new
50023 000089CF BE[BF87]                 mov     si,str1
50024                                   ;;mov   cx,3
50025                                   ;mov    cx,first_stop
50026                                   ; 21/03/2024
50027 000089D2 B103                     mov     cl,first_stop ; 3
50028 000089D4 F3A4                     rep     movsb
50029 000089D6 B048                     mov     al,48h                  ; al = opcode for dec ax
50030 000089D8 AA                       stosb
50031                                   ;add    si,2
50032                                   ; 21/03/2024
50033 000089D9 46                       inc     si
50034 000089DA 46                       inc     si
50035                                   ;;mov   cx,20
50036                                   ;mov    cx,second_stop
50037                                   ; 21/03/2024
50038 000089DB B114                     mov     cl,second_stop ; 20
50039 000089DD F3A4                     rep     movsb
50040 000089DF AA                       stosb                           ; put in dec ax
50041                                   ;add    si,2
50042                                   ; 21/03/2024
50043 000089E0 46                       inc     si
50044 000089E1 46                       inc     si
50045                                   ;;mov   cx,75
50046                                   ;mov    cx,last_stop
50047                                   ; 21/03/2024
50048 000089E2 B14B                     mov     cl,last_stop ; 75
50049 000089E4 F3A4                     rep     movsb
50050
50051                                   ep_notpacked:
50052                                       ;stc
50053                                   ep_done:
50054 000089E6 5F                       pop     di
50055 000089E7 5E                       pop     si
50056 000089E8 59                       pop     cx
50057 000089E9 58                       pop     ax
50058 000089EA 07                       pop     es
50059 000089EB 1F                       pop     ds
50060 000089EC 5B                       pop     bx
50061 000089ED C3                       retn
50062
50063                                   ;--------------------------------------------------------------------------
50064                                   ;
50065                                   ;   Procedure Name : chk_common_str
50066                                   ;
50067                                   ;   Input          : DS = DOSCODE
50068                                   ;                    ; ES:DI points to string in packed file
50069                                   ;
50070                                   ;   Output         ; Z if match else NZ
50071                                   ;
50072                                   ;--------------------------------------------------------------------------
50073
50074                                       ; 23/05/2019 - Retro DOS v4.0
50075                                   chk_common_str:
50076 000089EE BE[EF88]                 mov     si,scan_com
50077                                                                   ; ds:si -> scan string
50078                                   ;mov    cx,32
50079 000089F1 B92000                   mov     cx,size_scan_com
50080
50081 000089F4 F3A6                     repe    cmpsb
50082                                                                   ; M046 - start
50083                                   ; a fourth possible version of these exepacked programs have a
```

```
                                    ; 056h instead of 06Bh. See scan_com above
                                    ;
                                    ;       db  75h, 6Bh            ;jne CorruptExe
                                    ;
                                    ; If the mismatch at this point is due to a 56h instead of 6Bh
                                    ; we shall try to match the rest of the string
                                    ;
000089F6 7409                       jz      short ccs_done
000089F8 26807DFF56                 cmp     byte [es:di-1],56h
000089FD 7502                       jnz     short ccs_done

000089FF F3A6                       repe    cmpsb
                                ccs_done:                           ; M046 - end
00008A01 C3                         retn

                                ;--------------------------------------------------------------------------
                                ;
                                ;   Procedure Name : chk_patchsum
                                ;
                                ;   Input          : DS:SI -> string we're looking for
                                ;                  : ES:DI -> offset in packed file
                                ;                  : CX    = scan length
                                ;                  : BX    = length of check sum
                                ;                  : AX    = value of check sum
                                ;
                                ;   Output         : if patch & check sum compare
                                ;                         NC
                                ;                    else
                                ;                         CY
                                ;
                                ;   Uses           : AX, BX, CX, SI
                                ;
                                ;--------------------------------------------------------------------------

                                    ; 23/05/2019 - Retro DOS v4.0
                                chk_patchsum:
00008A02 57                         push    di

00008A03 F3A6                       repe    cmpsb

00008A05 7517                       jnz     short cp_fail           ; Q: does the patch match
                                                                    ; N: exit
                                                                    ; Y:

                                            ; we do a check sum starting from the location of the
                                            ; exepack signature 'RB' up to 11c/2 bytes, the end of the
                                            ; unpacking code.

00008A07 368B3E[8700]               mov     di,[ss:UNPACK_OFFSET] ; di -> start of unpack code
00008A0C 89D9                       mov     cx,bx                 ; cx = length of check sum

00008A0E 89C3                       mov     bx,ax                 ; save check sum passed to us in bx
00008A10 31C0                       xor     ax,ax
                                ep_chksum:
00008A12 260305                     add     ax,[es:di]
                                    ;add    di,2
                                    ; 01/07/2024 (PCDOS 7.1 IBMDOS.COM)
00008A15 47                         inc     di
00008A16 47                         inc     di
00008A17 E2F9                       loop    ep_chksum

00008A19 5F                         pop     di                    ; restore di

00008A1A 39D8                       cmp     ax,bx                 ; Q: does the check sum match
                                    ;jne    short cp_fail         ; N: exit
                                                                  ; Y:
                                    ; 25/09/2023
                                    ;clc
                                    ;retn
00008A1C 74E3                       je      short ccs_done ; cf=0

                                cp_fail:
00008A1E F9                         stc
00008A1F C3                         retn

                                ; 21/03/2024 - Retro DOS v5.0
                                ;%if 1
                                ; 28/12/2022 - Retro DOS v4.1
                                ;%if 0
                                ;--------------------------------------------------------------------------


                                ; M020 : BEGIN
                                ;
                                ;--------------------------------------------------------------------------
                                ;
                                ; procedure : RationalPatch
                                ;
                                ; A routine (in Ration DOS extender) which is invoked at hardware interrupts
                                ; clobbers CX register on 286 machines. (123 release 3 uses Rational DOS
                                ; extender). This routine identifies Buggy Rational EXEs and fixes the bug.
                                ;
                                ; THE BUG is in the following code sequence:
                                ;
                                ;8b 0e 10 00       mov     cx, ds:[10h]          ; delay count
                                ;90          even                        ; word align
                                ;e2 fe             loop    $                     ; wait         CLOBBERS CX
                                ;e8 xx xx   call    set_A20               ; enable A20
                                ;
                                ; This patch routine replaces the mov & the loop with a far call into a
                                ; routine in DOS data segment which is in low memory (because A20 line
                                ; is off). The routine (RatBugCode) in DOS data saves & restores CX around
                                ; a mov & loop.
                                ;
                                ; Identification of Buggy Rational EXE
                                ; ====================================
                                ;
                                ; (ALL OFFSETS ARE IN THE PROGRAM SECTION - EXCLUDING THE EXE HEADER)
                                ;
                                ; OFFSET                          Contains
                                ; ------                          --------
                                ; 0000h                  100 times Version number in binary
                                ;                   bug exists in version 3.48 thru 3.83 (both inclusive)
                                ;
                                ; 000ah                  the WORDS : 0000h, 0020h, 0000h, 0040h, 0001h
                                ;
                                ; 002ah                  offset where version number is stored in ASCII
                                ;                       e.g. '3.48A'
                                ;
                                ; 0030h                  offset of copyright string. Copyright strings either
                                ;                   start with "DOS/16M Copyright...." or
                                ;                   "Copyright.....". The string contains
                                ;                   "Rational Systems, Inc."
```

```
50208                                   ;
50209                                   ; 0020h                   word : Paragraph offset of the buggy code segment
50210                                   ;                         from the program image
50211                                   ; 0016h                   word : size of buggy code segment
50212                                   ;
50213                                   ;   Buggy code is definite to start after offset 200h in its segment
50214                                   ;
50215                                   ;--------------------------------------------------------------------------
50216
50217                                   ; 23/05/2019 - Retro DOS v4.0
50218                                   ; DOSCODE:B976h (MSDOS 6.21, MSDOS.SYS)
50219
50220                                   ; 04/12/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
50221                                   ; DOSCODE:B654h (MSDOS 5.0, MSDOS.SYS)
50222
50223                                   RScanPattern1:
50224 00008A20 000020000000400001-         db      0, 0, 20h, 0, 0, 0, 40h, 0, 1, 0
50224 00008A29 00
50225
50226                                   RLen1 equ $ - RScanPattern1
50227
50228                                   RScanPattern2:
50229 00008A2A 8B0E100090E2FEE8            db      8Bh, 0Eh, 10h, 00h, 90h, 0E2h, 0FEh, 0E8h
50230
50231                                   RLen2 equ $ - RScanPattern2
50232
50233                                   RScanPattern3:
50234 00008A32 8B0E1000E2FEE8              db      8Bh, 0Eh, 10h, 00h, 0E2h, 0FEh, 0E8h
50235
50236                                   RLen3 equ $ - RScanPattern2
50237
50238                                   ; DOSCODE:B98Fh (MSDOS 6.21, MSDOS.SYS)
50239                                   ; DOSCODE:B66Dh (MSDOS 5.0, MSDOS.SYS)
50240
50241                                   ;--------------------------------------------------------------------------
50242                                   ;
50243                                   ; INPUT : ES = segment where program got loaded
50244                                   ;
50245                                   ;--------------------------------------------------------------------------
50246
50247                                       ; 21/03/2024 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMDOS.COM)
50248                                       ; DOSCODE:CC2Fh (PCDOS 7.1 IBMDOS.COM)
50249
50250                                   RationalPatch:
50251 00008A39 FC                          cld
50252
50253                                   ; 21/03/2024
50254                                   %if 0
50255                                       push    ax
50256                                       push    bx
50257                                       push    cx
50258                                       push    dx
50259                                       push    si
50260                                       push    di
50261                                   %else
50262                                       ; 21/03/2024 (PCDOS 7.1 IBMDOS.COM)
50263                                       ;;;
50264 00008A3A 60                          pusha
50265                                       ;;;
50266                                   %endif
50267 00008A3B 06                          push    es
50268 00008A3C 1E                          push    ds                  ; we use all of them
50269 00008A3D BF0A00                      mov     di,0Ah              ; look for pat1 at offset 0Ah
50270 00008A40 0E                          push    cs
50271 00008A41 1F                          pop     ds
50272
50273 00008A42 BE[208A]                    mov     si,RScanPattern1
50274                                      ;mov    cx,10
50275 00008A45 B90A00                      mov     cx,RLen1
50276 00008A48 F3A6                        rep     cmpsb               ; do we have the pattern ?
50277 00008A4A 754A                        jne     short rpexit
50278 00008A4C 26A10000                    mov     ax,[es:0]
50279 00008A50 3D5C01                      cmp     ax,348              ; is it a buggy version ?
50280 00008A53 7241                        jb      short rpexit
50281 00008A55 3D7F01                      cmp     ax,383              ; is it a buggy version ?
50282 00008A58 773C                        ja      short rpexit
50283
50284 00008A5A E83AFD                      call    VerifyVersion
50285 00008A5D 7537                        jne     short rpexit
50286
50287 00008A5F 268B0E1600                  mov     cx,[es:16h]         ; Length of buggy code seg
50288 00008A64 81E90002                    sub     cx,200h             ; Length we search (we start
50289                                                                  ;  at offset 200h)
50290 00008A68 268E062000                  mov     es,[es:20h]         ; es=buggy code segment
50291 00008A6D BE[2A8A]                    mov     si,RScanPattern2
50292                                      ;mov    dx,8
50293 00008A70 BA0800                      mov     dx,RLen2
50294 00008A73 E809FD                      call    ScanCodeSeq         ; look for code seq with nop
50295 00008A76 740B                        jz      short rpfound
50296
50297 00008A78 BE[328A]                    mov     si,RScanPattern3
50298                                      ;mov    dx,15
50299 00008A7B BA0F00                      mov     dx,RLen3
50300 00008A7E E8FEFC                      call    ScanCodeSeq         ; look for code seq w/o nop
50301 00008A81 7513                        jnz     short rpexit
50302
50303                                   rpfound:
50304
50305                                   ;   we set up a far call into DOS data
50306                                   ;   dx has the length of the code seq we were searching for
50307
50308 00008A83 B09A                        mov     al,9Ah              ; far call opcode
50309 00008A85 AA                          stosb
50310 00008A86 B8[2611]                    mov     ax,RatBugCode
50311 00008A89 AB                          stosw
50312 00008A8A 8CD0                        mov     ax,ss
50313 00008A8C AB                          stosw
50314 00008A8D 89D1                        mov     cx,dx
50315 00008A8F 83E906                      sub     cx,6                ; filler (with NOPs)
50316 00008A92 B090                        mov     al,90h
50317 00008A94 F3AA                        rep     stosb
50318                                   rpexit:
50319 00008A96 1F                          pop     ds
50320 00008A97 07                          pop     es
50321
50322                                   ; 21/03/2024
50323                                   %if 0
50324                                       pop     di
50325                                       pop     si
50326                                       pop     dx
50327                                       pop     cx
50328                                       pop     bx
50329                                       pop     ax
50330                                   %else
```

```
                                    ; 21/03/2024 (PCDOS 7.1 IBMDOS.COM)
                                    ;;;
00008A98 61                         popa
                                    ;;;
                                %endif
00008A99 C3                         retn

                                ; M020 END

                                ;--------------------------------------------------------------------------
                                ; 21/03/2024
                                ;%endif    ; 28/12/2022

                                ;--------------------------------------------------------------------------
                                ;
                                ;   M068
                                ;
                                ;   Procedure Name : IsCopyProt
                                ;
                                ;   Inputs         : DS:100 -> start of com file just read in
                                ;
                                ;   Outputs        : sets the A20OFF_COUNT variable to 10 if
                                ;                    the program loaded in DS:100 uses a MICROSOFT
                                ;                    copy protect scheme that relies on the A20 line
                                ;                    being turned off for it's scheme to work.
                                ;
                                ;                    Note: The int 21 function dispatcher will turn
                                ;                          a20 off, if the A20OFF_COUNT is non-zero
                                ;                          and dec the A20OFF_COUNT before     iretting
                                ;                          to the user.
                                ;
                                ;   Uses           : ES, DI, SI, CX
                                ;
                                ;--------------------------------------------------------------------------

                                ; 23/05/2019 - Retro DOS v4.0

                                CPStartOffset      EQU     0175h
                                CPID1Offset EQU     011Bh
                                CPID2Offset EQU     0173h
                                CPID3Offset EQU     0146h
                                CPID4Offset EQU     0124h
                                ID1         EQU     05343h
                                ID2         EQU     05044h
                                ID3         EQU     0F413h
                                ID4         EQU     08000h

                                ; DOSCODE:B9FAh (MSDOS 6.21, MSDOS.SYS)

                                ; 04/12/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
                                ; DOSCODE:B71Ch (MSDOS 5.0, MSDOS.SYS)

                                ; 21/03/2024 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMDOS.COM)
                                ; DOSCODE:CC90h (PCDOS 7.1 IBMDOS.COM)

                                CPScanPattern:
00008A9A 89264801                   db      89h,26h,48h,01h                 ; mov [148],sp
00008A9E 8C0E4C01                   db      8Ch,0Eh,4Ch,01h                 ; mov [14C],cs
00008AA2 C7064A010001               db      0C7h,06h,4Ah,01h,00h,01h ; mov [14A],100h
00008AA8 8C0E1301                   db      8Ch,0Eh,13h,01h                 ; mov [113],cs
00008AAC B82001                     db      0B8h,20h,01h            ; mov ax,120h
00008AAF BE0001                     db      0BEh,00h,01h            ; mov si,100h

                                CPSPlen     EQU $ - CPScanPattern

                                ; DOSCODE:BA12h (MSDOS 6.21, MSDOS.SYS)
                                ; DOSCODE:B734h (MSDOS 5.0, MSDOS.SYS)
                                ; 21/03/2024
                                ; DOSCODE:CCA8h (PCDOS 7.1 IBMDOS.COM)

                                IsCopyProt:
00008AB2 813E1B014353               cmp     word [CPID1Offset],ID1
00008AB8 752D                       jne     short CP_done

00008ABA 813E73014450               cmp     word [CPID2Offset],ID2
00008AC0 7525                       jne     short CP_done

00008AC2 813E460113F4               cmp     word [CPID3Offset],ID3
00008AC8 751D                       jne     short CP_done

00008ACA 813E24010080               cmp     word [CPID4Offset],ID4
00008AD0 7515                       jne     short CP_done

00008AD2 0E                         push    cs
00008AD3 07                         pop     es
00008AD4 BF[9A8A]                   mov     di,CPScanPattern      ; es:di -> Pattern to find

00008AD7 BE7501                     mov     si,CPStartOffset      ; ds:si -> possible location
                                                                  ; of pattern

00008ADA B91800                     mov     cx,CPSPlen ; 24               ; cx = length of pattern
00008ADD F3A6                       repe    cmpsb
00008ADF 7506                       jnz     short CP_done

00008AE1 36C606[8500]0A             mov     byte [ss:A20OFF_COUNT],0Ah ; M071
                                CP_done:
00008AE7 C3                         retn

                                ;DOSCODE ENDS

                                    ;END

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

                                ;align 2 ; 05/09/2018 (Error!)

                                ; 07/12/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
                                ;align 16 ; 08/09/2018 (OK.)
                                align 2

                                ; 06/08/2018 - Retro DOS v3.0
                                ;==========================================================================
                                ; MSINIT.ASM
                                ;==========================================================================
                                ; 22/04/2019 - Retro DOS v4.0 (MSINIT.ASM, MSDOS 6.0, 1991)
                                ;
                                ; MAIN ENTRY FOR DOS INITIALIZATION
                                ;
                                    ; 15/07/2018 - Retro DOS v3.0
                                    ; (MSDOS 3.3, IBMDOS.COM, 1987)

                                ; temp iret instruction

                                ; 05/12/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
```

```
                                ; DOSCODE:B76Ah (MSDOS 5.0, MSDOS.SYS)

                                initiret: ; MSDOS 6.0
                                SYSBUF:
                                ;IRETT: ; 06/05/2019
00008AE8 CF                         iret

                                ; 22/04/2019 - Retro DOS v4.0

                                ; pointer to the BIOS data segment that will be available just to the
                                ; initialization code

00008AE9 7000                   InitBioDataSeg:    dw 70h ; KERNEL_SEGMENT = 0070h

                                ; 05/12/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
                                ; DOSCODE:B76Dh (MSDOS 5.0, MSDOS.SYS)

                                ; 22/03/2024 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMDOS.COM)
                                ; DOSCODE:CCE1h (PCDOS 7.1 IBMDOS.COM)

                                ; Convert AX from a number of bytes to a number of paragraphs (round up).

                                ParaRound:
00008AEB 83C00F                     add     ax, 15
00008AEE D1D8                       rcr     ax, 1
00008AF0 D1E8                       shr     ax, 1
00008AF2 D1E8                       shr     ax, 1
00008AF4 D1E8                       shr     ax, 1
00008AF6 C3                         retn

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

                                ; 22/03/2024 - Retro DOS v5.0
                                %if 1
                                ; 28/12/2022 - Retro DOS v4.1
                                ;%if 0

                                ;--------------------------------------------------------------------------
                                ;
                                ; Procedure Name : WhatCPUType
                                ;
                                ; Inputs     : none
                                ;
                                ; Outputs    : AL = 0 if CPU <  286
                                ;                 = 1 if CPU == 286
                                ;                 = 2 if CPU >= 386
                                ;
                                ; Regs. Mod.        : AX
                                ;
                                ;--------------------------------------------------------------------------

                                WhatCPUType:
                                    ; 25/04/2019 - Retro DOS v4.0
                                    ;get_cpu_type  ; done with a MACRO which can't be generated > once

                                    ;CPUTYPE.INC (MSDOS 6.0, 1991)

                                ; Note: this must be a macro, and not a subroutine in the BIOS since
                                ;   it is called from both CODE and SYSINITSEG.
                                ;
                                ;------GET_CPU_TYPE-----------------------------------May, 88 by M.Williamson
                                ;  Returns: AX = 0 if 8086 or 8088
                                ;              = 1 if 80286
                                ;              = 2 if 80386

                                    ; 04/11/2022
                                    ; MSDOS 5.0 MSDOS.SYS - DOSCODE:0BB03h

                                    ; 22/03/2024
                                    ; PCDOS 7.1 IBMDOS.COM - DOSCODE:0CCEDh

                                Get_CPU_Type:      ;macro
00008AF7 9C                         pushf
00008AF8 53                         push    bx                  ; preserve bx
00008AF9 31DB                       xor     bx,bx               ; init bx to zero

00008AFB 31C0                       xor     ax,ax               ; 0000 into AX
00008AFD 50                         push    ax                  ; put it on the stack...
00008AFE 9D                         popf                        ; ...then shove it into the flags
00008AFF 9C                         pushf                       ; get it back out of the flags...
00008B00 58                         pop     ax                  ; ...and into ax
00008B01 2500F0                     and     ax,0F000h           ; mask off high four bits
00008B04 3D00F0                     cmp     ax,0F000h           ; was it all 1's?
00008B07 740E                       je      short cpu_8086      ; aye; it's an 8086 or 8088

00008B09 B800F0                     mov     ax,0F000h           ; now try to set the high four bits..
00008B0C 50                         push    ax
00008B0D 9D                         popf
00008B0E 9C                         pushf
00008B0F 58                         pop     ax                  ; ...and see what happens
00008B10 2500F0                     and     ax,0F000h           ; any high bits set ?
00008B13 7401                       jz      short cpu_286       ; nay; it's an 80286
                                cpu_386:                        ; bx starts as zero
00008B15 43                         inc     bx                  ; inc twice if 386
                                cpu_286:                        ; just inc once if 286
00008B16 43                         inc     bx
                                cpu_8086:                       ; don't inc at all if 086
00008B17 89D8                       mov     ax,bx               ; put CPU type value in ax
00008B19 5B                         pop     bx                  ; restore original bx
00008B1A 9D                         popf

                                    ;endm

                                    ; 04/11/2022 (MSDOS 5.0 MSDOS.SYS)
00008B1B C3                         retn    ; 19/09/2023

                                ; ---------------------------------------------------------------------------
                                %endif      ; 28/12/2022

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

                                ; MAIN ENTRY FOR DOS INITIALIZATION

                                    ; 05/12/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
                                    ; DOSCODE:B779h (MSDOS 5.0 MSDOS.SYS)

                                    ; 23/03/2024
                                    ; 22/03/2024 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMDOS.COM)
                                    ; DOSCODE:CCEDh (PCDOS 7.1 IBMDOS.COM)

                                    ; DOSCODE:BA57h (MSDOS 6.22 MSDOS.SYS)
                                    ; BIOSCODE:CF81h (Windows ME IO.SYS)

                                    ; 30/05/2019
```

```
50579                                   ; 22/04/2019 - Retro DOS v4.0
50580                                   ; 07/07/2018 - Retro DOS v3.0
50581                                   ; Retro DOS v2.0 - 03/03/2018
50582                                   ; 03/11/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
50583                                   ; MSDOS 5.0 - MSDOS.SYS, offset 79A9h
50584                                 DOSINIT:
50585                                   ; MSDOS 6.21 - MSDOS.SYS, offset 7C77h
50586                                   ;
50587                                   ; Far call from SYSINIT
50588                                   ; DX = Memory size in paragraphs
50589                                   ; DS:SI = [DEVICE_LIST] (SYSINIT.S)
50590                                   ;         (Retro DOS v2.0, 16/03/2018)
50591                                   ;
50592                                   ; ES:DI = ptr to BIOS communication block (sysinit3.s)
50593                                   ;         (Retro DOS v4.0, 20/04/2019)
50594
50595 00008B1C FA                          CLI
50596 00008B1D FC                          CLD
50597
50598                                   ; 03/11/2022
50599                                   ;push   dx ; 30/05/2019              ; save parameters from BIOS
50600
50601                                   ; 17/12/2022
50602                                   ; 05/12/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
50603                                   ;push   dx ; =*=              ; save parameters from BIOS
50604
50605 00008B1E 56                       push    si
50606 00008B1F 1E                       push    ds
50607 00008B20 57                       push    di                    ;save di (ptr to BiosComBlock)
50608
50609 00008B21 8CC3                     mov     bx,es                 ;bx:di = ptr to BiosComBlock
50610
50611                                 ; First, move the DOS data segment to its final location in low memory
50612
50613                                   ;;;mov  ax,0BF69h ; MSDOS 6.21 MSDOS.SYS, file offset 7C7Fh
50614                                   ;;mov   ax,0BC77h ; MSDOS 5.0 MSDOS.SYS, file offset 79B1h
50615                                   ; 22/03/2024
50616                                   ;mov    ax,0D20Fh ; PCDOS 7.1 IBMDOS.COM, file offset 92FFh
50617 00008B23 B8[DF8F]                 mov     ax,MEMSTRT            ; get offset of end of init code
50618
50619                                   ; 05/12/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
50620 00008B26 83C00F                   add     ax,15                 ; round to nearest paragraph
50621                                   ;and    ax,~15  ; 0FFF0h      ; boundary
50622                                   ; 12/04/2024
50623 00008B29 24F0                     and     al,0F0h
50624
50625 00008B2B 89C6                     mov     si,ax                 ; si = offset of DOSDATA in current
50626                                                                 ; code segment
50627                                   ; 05/12/2022
50628                                   ; 30/04/2019 - Retro DOS v4.0
50629                                   ;xor    si,si
50630
50631                                   ;mov    ax,cs
50632                                   ;mov    ds,ax                 ; ds = current code segment
50633                                                                 ; DS:SI now points to dosdata
50634                                   ; 22/03/2024
50635 00008B2D 0E                       push    cs
50636 00008B2E 1F                       pop     ds
50637
50638                                   ;mov    es,[cs:0BA49h] ; MSDOS 6.21 IO.SYS, offset 7C8Eh
50639                                   ;mov    es,[cs:InitBioDataSeg]; First access to DosDataSg in
50640                                                                 ;  BData segment. Cannot use
50641                                                                 ;  getdseg macro here!!!
50642                                   ; 17/12/2022
50643 00008B2F 8E06[E98A]               mov     es,[InitBioDataSeg]
50644                                   ; 07/12/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
50645                                   ;mov    es,[cs:InitBioDataSeg]  ; ds = cs !
50646
50647                                   ;mov    es,[es:3]
50648 00008B33 268E060300               mov     es,[es:DosDataSg]     ; Get free location in low memory
50649
50650 00008B38 31FF                     xor     di,di                 ; ES:DI now points to RAM data
50651
50652                                   ;mov    cx,4970  ; Offset 0BA78h in MSDOS 6.21 MSDOS.SYS)
50653                                   ;mov    cx,4976  ; 25/05/2019
50654                                   ; 22/03/2024
50655                                   ;mov    cx,4934  ; PCDOS 7.1 IBMDOS.COM
50656                                   ; 05/12/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
50657                                   ;mov    cx,4962
50658                                   ;mov    cx,MSDAT001E          ; get end of dosdata = size of dosdata
50659 00008B3A B94613                   mov     cx,DOSDATASIZE ; = 4962 for MSDOS 5.0 MSDOS.SYS
50660                                                          ; = 4934 for PCDOS 7.1 IBMDOS.COM ; 01/07/2024
50661 00008B3D F3A4                     rep     movsb                 ; move data to final location
50662
50663 00008B3F 5F                       pop     di                    ; restore ptr to BiosComBlock
50664 00008B40 1F                       pop     ds                    ; restore parms from BIOS
50665 00008B41 5E                       pop     si
50666                                   ; 17/12/2022
50667                                   ;pop    dx ; 30/05/2019
50668                                   ; 05/12/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
50669                                   ;pop    dx ; =*=
50670
50671 00008B42 06                       push    es
50672 00008B43 1E                       push    ds
50673 00008B44 07                       pop     es                    ; es:si -> device chain
50674 00008B45 1F                       pop     ds                    ; ds points to dosdata
50675
50676                                 ;SR;
50677                                 ;We get a ptr to the BIOS exchange data block. This has been setup right
50678                                 ;now so that the EXEC call knows when SysInit is present to do the special
50679                                 ;lie table handling for device drivers. This can be expanded later on to
50680                                 ;establish a communication block from the BIOS to the DOS.
50681
50682                                   ;mov    [1040h],di     ; Offset 0BA87h in MSDOS 6.21 MSDOS.SYS)
50683                                   ;mov    [1042h],bx
50684 00008B46 893E[2313]               mov     [BiosDataPtr],di
50685 00008B4A 891E[2513]               mov     [BiosDataPtr+2],bx    ; save ptr to BiosComBlock
50686
50687 00008B4E 2E8C1E[0700]             mov     [cs:DosDSeg],ds               ; set pointer to dosdata in code seg
50688
50689                                   ; Set the segment of Lowint23/24/28Addr in msctrlc.asm to dosdata
50690
50691 00008B53 2E8C1E[F95D]             mov     [cs:LowInt23Addr+2],ds; set pointers in code seg
50692 00008B58 2E8C1E[FD5D]             mov     [cs:LowInt24Addr+2],ds
50693 00008B5D 2E8C1E[015E]             mov     [cs:LowInt28Addr+2],ds
50694
50695                                   ;mov    [346h],dx      ; MSDOS 6.21 DOSDATA addresses
50696                                   ;mov    [584h],sp
50697                                   ;mov    [586h],ss
50698 00008B62 8916[4603]               mov     [ENDMEM],dx    ; =*=
50699 00008B66 8926[8405]               mov     [USER_SP],sp
50700 00008B6A 8C16[8605]               mov     [USER_SS],ss
50701
50702                                   ;mov    ax,ds          ; set up ss:sp to dosdata:dskstack
```

```
                                    ;mov    ss,ax
                                    ; 01/07/2024
00008B6E 1E                         push    ds
00008B6F 17                         pop     ss

                                    ;mov    sp,920h         ; MSDOS 6.21 DOSDATA address
                                    ;mov    sp,offset dosdata:dskstack
00008B70 BC[2009]                   mov     sp,DSKSTACK     ; 920h ; PCDOS 7.1 IBMDOS.COM ; 22/03/2024

                                ;M023
                                ; Init patch ptrs to default values

                                ; 22/03/2024
                                %if 0
                                    ;mov    word [1212h],RetExePatch
                                    ;mov    word [1214h],RetExePatch
                                    ;mov    word [61h],RetExePatch
                                    mov     word [FixExePatch],RetExePatch       ; M023
                                    ; 28/12/2022 - Retro DOS v4.1
                                    ;mov    word [RationalPatchPtr],RetExePatch ; M023
                                    mov     word [ChkCopyProt],RetExePatch       ; M068
                                %else
                                    ; 22/03/2024 (PCDOS 7.1 IBMDOS.COM)
                                    ;;;
00008B73 B8[9B72]                   mov     ax,RetExePatch
00008B76 A3[670D]                   mov     [FixExePatch],ax
00008B79 A3[690D]                   mov     [RationalPatchPtr],ax ; 25/03/2024
00008B7C A3[6100]                   mov     [ChkCopyProt],ax
                                    ;;;
                                %endif

                                ; 22/03/2024
                                %if 1
                                ; 05/12/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
                                ;%if 0      ; 19/09/2023

                                ; Setup to call 386 Rational DOS Extender patch routine if running on
                                ; a 386 or later. Unlike other patches, this is not dependent on MS-DOS
                                ; running in the HMA.

00008B7F E875FF                     call    WhatCPUType     ; get cpu type (0 < 286,1==286,2 >= 386)
00008B82 3C02                       cmp     al,2            ;   386 or later?
00008B84 B8[BD85]                   mov     ax,Rational386Patch
00008B87 7303                       jae     short di_set_patch
00008B89 B8[9B72]                   mov     ax,RetExePatch ; < 386, don't need this patch
                                di_set_patch:
00008B8C A3[3E13]                   mov     [Rational386PatchPtr],ax ; patch routine or RET instr.

                                %endif
                                    ; Set up the variable temp_dosloc to point to the dos code segment

00008B8F 8CC8                       mov     ax,cs           ; ax = current segment of DOS code

                                    ; ax now holds segment of DOS code
00008B91 A3[A30A]                   mov     [TEMP_DOSLOC],ax   ; store temp location of DOS

00008B94 8C06[4A00]                 mov     word [NULDEV+2],es ; nuldev -> points to device chain
00008B98 8936[4800]                 mov     word [NULDEV],si
                                ;SR;
                                ; There are some locations in the Win386 instance data structures
                                ; which need to be set up with the DOS data segment. First, initialize
                                ; the segment part of the instance table pointer in the SIS.

                                    ;;mov   [0FF2h],ds ; [Win386_Info+14+2]
                                    ;mov    [0EF1h],ds ; PCDOS 7.1 IBMDOS.COM ; 22/03/2024
00008B9C 8C1E[F10E]                 mov     [Win386_Info+Win386_SIS.Instance_Data_Ptr+2],ds

                                ; Now initialize the segment part of the pointer to the data in each
                                ; instance table entry.

00008BA0 56                         push    si              ; preserve pointer to device chain
                                    ; 18/12/2022
                                    ; cx = 0
00008BA1 B107                       mov     cl,7
                                    ;mov    cx,7            ; There are 7 entries in the instance table
                                                            ; M019
                                    ;;mov   si,0FF6h ; offset (dosdata:Instance_Table+2)
                                    ;mov    si,0EF9h ; PCDOS 7.1 IBMDOS.COM ; 22/03/2024
00008BA3 BE[F90E]                   mov     si,Instance_Table+2 ; point si to segment field
                                Instance_init_loop:
00008BA6 8C1C                       mov     [si],ds         ; set offset in instance entry
                                    ;add    si,6
00008BA8 83C606                     add     si,size_of_Win386_IIS ; move on to next entry
00008BAB E2F9                       loop    Instance_init_loop

                                ;Initialize the WIN386 2.xx instance table with the DOS data segment value

                                    ; 18/12/2022
00008BAD B105                       mov     cl,5
                                    ;mov    cx,5            ; There are five entries in the instance table

                                    ;mov    si,(offset dosdata:OldInstanceJunk) + 6
                                    ;;mov   si,11EDh        ; point si to segment field
                                    ;mov    si,1102h ; PCDOS 7.1 IBMDOS.COM ; 22/03/2024
00008BAF BE[0211]                   mov     si,OldInstanceJunk+6
                                OldInstance_init_loop:
00008BB2 8C1C                       mov     [si],ds         ; set offset in instance entry
00008BB4 83C606                     add     si,6            ; move on to next entry
00008BB7 E2F9                       loop    OldInstance_init_loop
00008BB9 5E                         pop     si              ; restore pointer to device chain

                                ; End of WIN386 2.xx compatibility bullshit

                                ; 05/12/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
                                %if 0
                                    ; 30/04/2019
                                    ;push   es
                                    ;pop    ds
                                                    ; ds:si points to console device

                                    ; 24/04/2019 - Retro DOS v4.0

                                    ; 15/07/2018
                                    ; MSDOS 3.3 (IBMDOS.COM, 1987)
                                    ; (Set INT 2Ah handler address to an 'IRET')

                                    ; need crit vector inited to use deviocall
                                    ;push   ds                  ; preserve segment of device chain
                                    push    es ; 30/04/2019

                                %endif
                                    ; 05/12/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
00008BBA 06                         push    es
                                    ; 17/12/2022
```

```
50827                                   ;pop    ds
50828                                   ;push   ds
50829
50830 00008BBB 31C0                     xor     ax,ax
50831 00008BBD 8ED8                     mov     ds,ax                   ; point DS to int vector table
50832 00008BBF B8[E88A]                 mov     ax,initiret
50833                                   ;mov    [0A8h],ax  ; [2Ah*4]
50834 00008BC2 A3A800                   mov     [addr_int_ibm],ax
50835 00008BC5 8CC8                     mov     ax,cs
50836                                   ;mov    [0AAh],ax  ; [(2Ah*4)+2]
50837 00008BC7 A3AA00                   mov     [addr_int_ibm+2],ax
50838 00008BCA 1F                       pop     ds                      ; restore segment of device chain
50839
50840 00008BCB E84702                   call    CHARINIT                ; initialize console driver
50841 00008BCE 56                       push    si                      ; save pointer to header
50842
50843 00008BCF 16                       push    ss                      ; move pointer to dos data...
50844 00008BD0 07                       pop     es                      ; ...into ES
50845
50846                                   ;initialize sft for file 0 (CON)
50847
50848                                       ; 07/07/2018 - Retro DOS v3.0
50849                                   ; 24/04/2019 - Retro DOS v4.0
50850                                   ;mov    di,SFTABL+6             ; SFT0_SFTable ; 22/03/2024
50851 00008BD1 BF[D200]                 MOV     DI,SFTABL+SFT.SFTable ; Point to sft 0
50852 00008BD4 B80300                   MOV     AX,3
50853 00008BD7 AB                       STOSW                   ; Refcount
50854                                      ;DEC       AL
50855                                   ; 22/03/2024 - Retro DOS v5.0
50856 00008BD8 48                       dec     ax
50857 00008BD9 AB                       STOSW                   ; Access rd/wr, compatibility
50858 00008BDA 30C0                     XOR     AL,AL
50859 00008BDC AA                       STOSB                   ; attribute
50860                                   ;mov    al,0C3h
50861 00008BDD B0C3                     mov     al,devid_device_EOF|devid_device|ISCIN|ISCOUT
50862 00008BDF AB                       STOSW                   ; flags
50863 00008BE0 89F0                     mov     ax,si
50864 00008BE2 AB                           stosw               ; device pointer in devptr
50865 00008BE3 8CD8                         mov     ax,ds
50866 00008BE5 AB                           stosw
50867 00008BE6 31C0                         xor     ax,ax   ; 0
50868
50869                                   ; 22/03/2024
50870                                   %if 0
50871                                       stosw               ; firclus
50872                                       stosw               ; time
50873                                       stosw               ; date
50874                                       dec     ax      ; -1
50875                                       stosw               ; size
50876                                       stosw
50877                                       inc     ax      ; 0
50878                                       stosw               ; position
50879                                       stosw
50880                                   %else
50881                                       ; 22/03/2024 (PCDOS 7.1 IBMDOS.COM)
50882                                       ;;;
50883 00008BE8 83C720                       add     di,32           ; SFTABL+SFT.SFTable + 43
50884 00008BEB AB                           stosw                   ; SFT0_SFTable + 43 ; .sf_fclus32
50885 00008BEC AB                           stosw                   ; SF_ENTRY.sf_fclus32+2
50886 00008BED 83C7DE                       add     di,-34          ; 0FFDEh ; SFTABL+SFT.SFTable + 13
50887 00008BF0 AB                           stosw                   ; SFT0_SFTable + 13 ; .sf_time
50888 00008BF1 AB                           stosw                   ; SF_ENTRY.sf_date
50889 00008BF2 48                           dec     ax              ; -1
50890 00008BF3 AB                           stosw                   ; SFT0_SFTable + 17 ; .sf_size
50891 00008BF4 AB                           stosw                   ; SF_ENTRY.sf_size + 2
50892 00008BF5 40                           inc     ax              ; 0
50893 00008BF6 AB                           stosw                   ; SFT0_SFTable + 21 ; .sf_position
50894 00008BF7 AB                           stosw                   ; SF_ENTRY.sf_position + 2
50895                                       ;;;
50896                                   %endif
50897
50898                                       ;add    di,7            ; SFT0_SFTable + 32 ; 22/03/2024
50899 00008BF8 83C707                       add     di,SF_ENTRY.sf_name-SF_ENTRY.sf_cluspos
50900                                                               ; point at name
50901                                       ;add    si,10
50902 00008BFB 83C60A                       add     si,SYSDEV.NAME ; sdevname
50903                                                               ; point to name
50904 00008BFE B90400                       mov     cx,4
50905 00008C01 F3A5                         rep     movsw           ; name
50906 00008C03 B103                         mov     cl,3
50907 00008C05 B020                         mov     al," "
50908 00008C07 F3AA                         rep     stosb           ; extension
50909
50910 00008C09 5E                           pop     si              ; get back pointer to header
50911
50912                                                               ; mark device as CON I/O
50913                                       ; 15/07/2018
50914                                          ;OR BYTE [SI+4],ISCIN|ISCOUT ; or byte [si+4],3
50915 00008C0A 804C0403                     OR      BYTE [SI+SYSDEV.ATT],ISCIN|ISCOUT
50916                                       ; 12/03/2018
50917                                       ;mov    [ss:32h],si
50918 00008C0E 368936[3200]                 MOV      [SS:BCON],SI
50919                                       ;mov    [ss:34h],ds
50920 00008C13 368C1E[3400]                     MOV      [SS:BCON+2],DS
50921
50922                                       ; initialize each device until the clock device is found
50923
50924                                   CHAR_INIT_LOOP:
50925 00008C18 C534                             LDS     SI,[SI]                 ; AUX device
50926 00008C1A E8F801                       call    CHARINIT
50927                                               ;15/07/2018
50928                                       ;test   byte [SI+4],8
50929 00008C1D F6440408                     TEST     BYTE [SI+SYSDEV.ATT],ISCLOCK
50930 00008C21 74F5                             JZ      SHORT CHAR_INIT_LOOP
50931                                       ; 12/03/2018
50932                                       ;mov    [ss:2Eh],si
50933 00008C23 368936[2E00]                     MOV      [SS:BCLOCK],SI
50934                                       ;mov    [ss:30h],ds
50935 00008C28 368C1E[3000]                     MOV      [SS:BCLOCK+2],DS
50936                                           ;MOV       BP,MEMSTRT ; Retro DOS 3.0 ; ES:BP points to DPB
50937
50938                                       ;mov    bp,4970             ; bp = pointer to free mem
50939                                       ;mov    bp,4976  ; 25/05/2019 - Retro DOS v4.0
50940                                       ; 05/12/2022 - Retro DOS v4.0 (Modified MSDOS 5.0, MSDOS.SYS)
50941                                       ;mov    bp,4962 ; (MSDOS 5.0 MSDOS.SYS)
50942                                       ; 22/03/2024 - Retro DOS v5.0
50943                                       ;mov    bp,4934 ; (PCDOS 7.1 IBMDOS.COM)
50944 00008C2D BD4613                      mov     bp,MSDAT001E            ; es:bp points to dpb area
50945
50946 00008C30 36892E[2600]                 mov     [ss:DPBHEAD],bp             ; set offset of pointer to DPB's
50947 00008C35 368C06[2800]                 mov     [ss:DPBHEAD+2],es       ; set segment of pointer to DPB's
50948
50949                                   PERDRV:
50950                                       ;lds    si,[SI+SYSDEV.NEXT] ; 15/07/2018
```

```
50951 00008C3A C534                         LDS SI,[SI]               ; Next device
50952 00008C3C 83FEFF                       CMP SI,-1  ; 0FFFFh
50953                                   ;JZ     SHORT CONTINIT
50954                                   ; 23/03/2024
50955                                   ;;;
50956 00008C3F 7503                     jnz     short PERDRV2
50957 00008C41 E99C00                   jmp     CONTINIT
50958                                 PERDRV2:
50959                                   ;;;
50960
50961 00008C44 E8CE01                       call       CHARINIT
50962
50963                                   ; Retro DOS v2.0 - 16/03/2018 (NOTE for 'CHARINIT' return):
50964                                   ; [CALLUNIT] = Number of drives for (Disk) Block Dev Driver ([DRVMAX])
50965                                   ;           (..When the command is 'DSK$INIT', as in 'CHARINIT')
50966                                   ; [CALLBPB] = [DEVCALL.COUNT] = Address of the BPB (DEVCALL offset 18)
50967                                   ; (REF: MSDOS 3.3 MSBIO2.ASM, MSDATA.INC, MSDISK.ASM, MSBIO1.ASM)
50968                                   ; (.. !DSK$IN' in MSBIO1.ASM)
50969                                   ; DEVCALL.MEDIA = CALLUNIT (DEVCALL offset 13)
50970
50971                                       ; 15/07/2018
50972                                   ;test  word [SI+4],8000h            ; DEVTYP
50973                                       ; 17/12/2022
50974                                   ;test  byte [SI+5],80h
50975 00008C47 F6440580                 test   byte [SI+SYSDEV.ATT+1],(DEVTYP>>8) ; 80h
50976                                   ;TEST  word [SI+SYSDEV.ATT],DEVTYP ; 8000h
50977 00008C4B 75ED                     JNZ     SHORT PERDRV               ; Skip any other character devs
50978
50979 00008C4D 368A0E[6703]                 MOV CL,[SS:CALLUNIT] ; 12/03/2018
50980 00008C52 30ED                     XOR     CH,CH
50981                                       ; 07/07/2018
50982                                   ;MOV    [SI+10],CL            ; Number of units in name field
50983 00008C54 884C0A                   mov     [si+SYSDEV.NAME],cl   ; sdevname
50984 00008C57 368A16[4600]             MOV     DL,[SS:NUMIO] ; 15/03/2018
50985 00008C5C 30F6                     XOR     DH,DH
50986 00008C5E 36000E[4600]             ADD     [SS:NUMIO],CL  ; 12/03/2018
50987 00008C63 1E                       PUSH    DS
50988 00008C64 56                           PUSH    SI
50989 00008C65 36C51E[6C03]             LDS     BX,[SS:CALLBPB]       ; 12/03/2018
50990
50991                                 PERUNIT:
50992 00008C6A 8B37                         MOV     SI,[BX]                 ; DS:SI Points to BPB
50993 00008C6C 43                           INC     BX
50994 00008C6D 43                           INC     BX                      ; On to next BPB
50995                                   ; 15/12/2022
50996                                   ; 07/07/2018
50997                                       ;mov        [ES:BP+DPB.DRIVE],DL
50998 00008C6E 26885600                 MOV     [ES:BP],DL
50999                                   ; 05/12/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
51000                                   ;;mov   [ES:BP+0],DL
51001                                   ;mov    [ES:BP+DPB.DRIVE],DL
51002
51003                                   ;MOV    [ES:BP+1],DH
51004 00008C72 26887601                 MOV     [ES:BP+DPB.UNIT],DH
51005 00008C76 53                           PUSH    BX
51006 00008C77 51                           PUSH    CX
51007 00008C78 52                           PUSH    DX
51008
51009                                   ; 23/03/2024 (PCDOS 7.1 IBMDOS.COM)
51010                                   ;;;
51011 00008C79 BA5241                   mov     dx,4152h ; 'RA'       ; signature ('AR') for FAT32 extended BPB/DPB
51012 00008C7C 31C9                     xor     cx,cx  ; 0
51013                                   ;mov    [es:bp+29],cx
51014 00008C7E 26894E1D                 mov     [es:bp+DPB.NEXT_FREE],cx
51015 00008C82 394C0B                   cmp     [si+A_BPB.SECTORSPERFAT],cx ; 16 bit FAT size = 0 ?
51016                                   ;cmp    [si+11],cx     ; BPB_FATSz16
51017 00008C85 7514                     jnz     short PERUNIT2 ; FAT (FAT12 or FAT16) -old DPB-
51018                                   ;mov    [es:bp+57],cx  ; FAT32 -new- DPB
51019 00008C87 26894E39                 mov     [es:bp+DPB.FAT32_NXTFREE],cx
51020                                   ;mov    [es:bp+59],cx
51021 00008C8B 26894E3B                 mov     [es:bp+DPB.FAT32_NXTFREE+2],cx
51022 00008C8F 49                       dec     cx  ; 0FFFFh   ; -1
51023                                   ;mov    [es:bp+31],cx
51024 00008C90 26894E1F                 mov     [es:bp+DPB.FREE_CNT],cx
51025                                   ;mov    [es:bp+33],cx
51026 00008C94 26894E21                 mov     [es:bp+DPB.FREE_CNT_HW],cx
51027 00008C98 B95845                   mov     cx,4558h ; 'XE'       ; signature ('EX') for FAT32 extended BPB/DPB
51028                                 PERUNIT2:
51029                                   ;;;
51030
51031                                       ;invoke    $SETDPB
51032 00008C9B E8C787                       CALL       _$SETDPB             ; build DPB!
51033
51034                                   ; 07/07/2018
51035                                   ;MOV    AX,[ES:BP+2]
51036 00008C9E 268B4602                 mov     ax,[ES:BP+DPB.SECTOR_SIZE]
51037                                       ; 12/03/2018
51038 00008CA2 363B06[3600]             CMP     AX,[SS:MAXSEC]        ; Q:is this the largest sector so far
51039 00008CA7 7604                     JBE      SHORT NOTMAX         ; N:
51040 00008CA9 36A3[3600]               MOV     [SS:MAXSEC],AX        ; Y: save it in maxsec
51041                                 NOTMAX:
51042                                   ; set the next dpb field in the currently built bpb
51043                                   ; and mark as never accessed
51044
51045                                   ; 24/04/2019
51046 00008CAD 89E8                     mov     ax,bp                 ; get pointer to DPB
51047                                   ;;add   ax,33
51048                                   ;add    ax,61  ; next DPB (PCDOS 7.1 DPB size = 61) ; 23/03/2024
51049 00008CAF 83C03D                   add     ax,DPBSIZ             ; advance pointer to next DPB
51050                                                                 ; set seg & offset of next DPB
51051                                   ;mov    [es:bp+25],ax
51052 00008CB2 26894619                 mov     [es:bp+DPB.NEXT_DPB],ax
51053                                   ;mov    [es:bp+27],es
51054 00008CB6 268C461B                 mov     [es:bp+DPB.NEXT_DPB+2],es
51055                                                                 ; mark as never accessed
51056                                   ;mov    byte [es:bp+24],0FFh
51057 00008CBA 26C64618FF               mov     byte [es:bp+DPB.FIRST_ACCESS],-1
51058
51059 00008CBF 5A                       POP      DX
51060 00008CC0 59                           POP     CX
51061 00008CC1 5B                           POP     BX
51062 00008CC2 8CD8                         MOV     AX,DS                   ; save segment of bpb array
51063 00008CC4 5E                           POP     SI
51064 00008CC5 1F                           POP     DS
51065                                                                 ; ds:si -> device header
51066                                                                 ; store it in the corresponding dpb
51067                                   ; 07/07/2018
51068                                   ;MOV    [ES:BP+19],SI ; 24/04/2019
51069 00008CC6 26897613                 mov     [ES:BP+DPB.DRIVER_ADDR],si
51070                                   ;MOV    [ES:BP+21],DS ; 24/04/2019
51071 00008CCA 268C5E15                 mov     [ES:BP+DPB.DRIVER_ADDR+2],ds
51072
51073 00008CCE 1E                       PUSH    DS                    ; save pointer to device header
51074 00008CCF 56                       PUSH    SI
```

```
00008CD0 FEC6                       INC     DH                      ; inc unit #
00008CD2 FEC2                       INC     DL                      ; inc drive #
00008CD4 8ED8                       MOV     DS,AX                   ; restore segment of BPB array
                                    ;;add   bp,33 ; 24/04/2019
                                    ;add    bp,61 ; 23/03/2024 (PCDOS 7.1)
00008CD6 83C53D                     ADD     BP,DPBSIZ               ; advance pointer to next dpb
00008CD9 E28F                       LOOP    PERUNIT                 ; process all units in each driver

00008CDB 5E                         POP      SI                     ; restore pointer to device header
00008CDC 1F                         POP      DS
00008CDD E95AFF                     JMP     PERDRV                  ; process all drivers in chain

                                CONTINIT:
                                    ; 24/04/2019
                                    ;;sub   bp,33                   ; set link in last DPB to -1
                                    ;sub    bp,61 ; 23/03/2024 (PCDOS 7.1)
00008CE0 83ED3D                     sub     bp,DPBSIZ               ; back up to last dpb
                                                                    ; set last link offset & segment
                                ; 23/03/2024 - Retro DOS v5.0
                                %if 0
                                    ;mov    word [bp+25],0FFFFh
                                    mov     word [bp+DPB.NEXT_DPB],-1
                                    ;mov    word [bp+27],0FFFFh
                                    mov     word [bp+DPB.NEXT_DPB+2],-1
                                %else
                                    ; 23/03/2024 (PCDOS 7.1 IBMDOS.COM)
                                    ;;;
00008CE3 B8FFFF                     mov     ax,0FFFFh ; -1
                                    ;mov    word [bp+25],ax
00008CE6 894619                     mov     word [bp+DPB.NEXT_DPB],ax ; -1
                                    ;mov    word [bp+27],ax
00008CE9 89461B                     mov     word [bp+DPB.NEXT_DPB+2],ax ; -1
                                    ;;;
                                %endif
                                    ;;add   bp,33
                                    ;add    bp,61 ; 23/03/2024 (PCDOS 7.1)
00008CEC 83C53D                     add     BP,DPBSIZ               ; advance to free memory again
                                                                    ; the DPB chain is done.
00008CEF 16                         push    ss
00008CF0 1F                         pop     ds

00008CF1 89E8                       mov     ax,bp
00008CF3 E8F5FD                     call    ParaRound               ; round up to segment

00008CF6 8CDA                       mov     dx,ds                   ; dx = dosdata segment
00008CF8 01C2                       add     dx,ax                   ; dx = ds+ax first free segment

00008CFA BB0F00                     mov     bx,0Fh

                                    ; 24/05/2019
                                    ;mov    cx,[ENDMEM]
                                    ; 05/12/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
                                    ; 17/12/2022
                                    ;mov    cx,[ENDMEM]
                                                                    ; set seg inpacketto dosdata
00008CFD 8C1E[A203]                 mov     [DSKCHRET+3],ds ; mov [DOSSEG_INIT],ds

                                ; Patch in the segments of the interrupt vectors with current code segment.
                                ; Also patch in the segment of the pointers in the dosdata area.
                                ;
                                ; Note: Formerly, temp_dosloc was initialized to -1 until after these
                                ; calls were done. The procedure patch_misc_segments is called multiple
                                ; times, and relies on temp_dosloc being initialized to -1 as a flag
                                ; for the first invocation. Thus, we must set it to -1 for this call.

00008D01 52                         push    dx                      ; preserve first free segment

00008D02 A1[A30A]                   mov     ax,[TEMP_DOSLOC]        ; ax = segment to patch in
00008D05 8EC0                       mov     es,ax                   ; es = segment of DOS
00008D07 C706[A30A]FFFF             mov     word [TEMP_DOSLOC],-1 ; -1 means first call to patch_misc_segments

00008D0D E8BC01                     call    patch_vec_segments      ; uses AX as doscode segment
00008D10 E8F101                     call    patch_misc_segments     ; patch in segments for sharer and
                                                                    ; other tables with seg in ES.
                                    ; 17/12/2022
                                    ; cx = 0
00008D13 8C06[A30A]                 mov     [TEMP_DOSLOC],es        ; put back segment of dos code

00008D17 5A                         pop     dx                      ; restore first free segment

                                ; We shall now proceed to set the offsets of the interrupt vectors handled
                                ; by DOS to their appropriate values in DOSCODE. In case the DOS loads in
                                ; HIMEM the offsets also will be patched to their appropriate values in the
                                ; low_mem_stub by seg_reinit.

                                    ;xor     ax,ax ; 0
                                    ;mov     ds,ax
                                    ;mov     es,ax
                                    ; 17/12/2022
                                    ; cx = 0
                                    ;xor     cx,cx ; 0
00008D18 8ED9                       mov     ds,cx
00008D1A 8EC1                       mov     es,cx

                                    ; set the segment of int 24 vector that was
                                    ; left out by patch_vec_segments above.

                                    ; 17/12/2022
                                ; 05/12/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
                                ;%if 0
                                    ; 24/05/2019
                                    ;;mov    di,90h
                                    ;;mov    di,4*int_fatal_abort
                                    ;mov     di,addr_int_fatal_abort
00008D1C BF9200                     mov     di,addr_int_fatal_abort+2 ; 24/05/2019

00008D1F 36A1[A30A]                 mov     ax,[ss:TEMP_DOSLOC]
                                    ;mov     [di+2],ax  ; int 24h segment
00008D23 8905                       mov     [di],ax ; 24/05/2019

                                    ;;mov    di,82h
                                    ;mov     di,INTBASE+2

                                ;%endif
                                    ; 17/12/2022
                                    ; 05/12/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
                                    ;;mov    di,90h
                                    ;;mov    di,4*int_fatal_abort
                                    ;mov     di,addr_int_fatal_abort
                                    ;mov     ax,[ss:TEMP_DOSLOC]
                                    ;mov     [di+2],ax  ; int 24h segment
                                    ;;mov    di,82h
                                    ;mov     di,INTBASE+2

```

```
                                    ; set default divide trap offset
                                    ;mov    word ptr ds:[0],offset doscode:divov
00008D25 C7060000[D55F]             mov     word [0],DIVOV
                                    ; set vectors 20-28 and 2a-3f to point to iret.
                                    ;mov    di,80h
00008D2B BF8000                     mov     di,INTBASE
                                    ;mov    ax,offset doscode:irett
00008D2E B8[CB02]                   mov     ax,IRETT
                                    ; 17/12/2022
                                    ; cx = 0
00008D31 B109                       mov     cl,9
                                    ;mov    cx,9                    ; set 9 offsets (skip 2 between each)
                                                                    ;   sets offsets for ints 20h-28h
                                iset1:
00008D33 AB                         stosw
                                    ;add    di,2
                                    ; 20/09/2023
00008D34 47                         inc     di
00008D35 47                         inc     di
00008D36 E2FB                       loop    iset1
00008D38 83C704                     add     di,4                    ; skip vector 29h
                                ;   mov     cx,6                    ; set 6 offsets (skip 2 between each)
                                ;                                   ;   sets offsets for ints 2Ah-2Fh
                                ;iset2:
                                ;   stosw
                                ;   add     di,2
                                ;   loop    iset2
                                ; 30h & 31H is the CPM call entry point whose segment address is set up by
                                ; patch_vec_segments above. So skip it.
                                ;   add     di,8                    ; skip vector 30h & 31h
                                    ;;;
                                    ; 06/05/2019 - Retro DOS v4.0
                                    ;mov    cx,5                    ; set offsets for int 2Ah-2Eh
                                    ; 17/12/2022
00008D3B B105                       mov     cl,5 ; 28/06/2019
                                    ; 05/12/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
                                    ;mov    cx,6
                                iset2:
00008D3D AB                         stosw
                                    ;add    di,2
                                    ; 20/09/2023
00008D3E 47                         inc     di
00008D3F 47                         inc     di
00008D40 E2FB                       loop    iset2
                                    ; 05/12/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
                                    ; 17/12/2022
00008D42 83C70C                     add     di,12                   ; skip vectors 2Fh, 30h & 31h
                                    ;add    di,8
                                    ;;;
                                    ; 17/12/2022
00008D45 B10E                       mov     cl,14
                                    ;mov    cx,14                   ; set 14 offsets (skip 2 between each)
                                                                    ;   sets offsets for ints 32h-3Fh
                                iset3:
00008D47 AB                         stosw
                                    ;add    di,2
                                    ; 20/09/2023
00008D48 47                         inc     di
00008D49 47                         inc     di
00008D4A E2FB                       loop    iset3
                                ;if installed
                                    ; set the offset of int2f handler
                                    ;mov    word [0BCh],INT2F
00008D4C C706BC00[3A07]             mov     word [02Fh*4],INT2F
                                    ; set segment to doscode as we have to do int 2f to check for XMS
00008D52 36A1[A30A]                 mov     ax,[ss:TEMP_DOSLOC]     ; get segment of doscode
                                    ;mov    [0BEh],ax
00008D56 A3BE00                     mov     [(02Fh*4)+2],ax
                                ;endif
                                    ; set up entry point call at vectors 30-31h. Note the segment of the
                                    ; long jump will be patched in by seg_reinit
                                    ;mov    byte [C0h],0EAh
00008D59 C606C000EA                 mov     byte [ENTRYPOINT],mi_long_jmp
                                    ;mov    byte [C1h],CALL_ENTRY
00008D5E C706C100[CC02]             mov     word [ENTRYPOINT+1],CALL_ENTRY
00008D64 C7068000[C502]             mov     word [addr_int_abort],QUIT    ; INT 20h
00008D6A C7068400[F102]             mov     word [addr_int_command],COMMAND ; INT 21h
00008D70 C70688000001               mov     word [addr_int_terminate],100h       ; INT 22h
00008D76 89168A00                   mov     word [addr_int_terminate+2],dx
00008D7A C7069400[2D05]             mov     word [addr_int_disk_read],ABSDRD   ; INT 25h
00008D80 C7069800[DF05]             mov     word [addr_int_disk_write],ABSDWRT ; INT 26h
00008D86 C7069C00[3E72]             mov     word [addr_int_keep_process],STAY_RESIDENT ; INT 27h
00008D8C 16                         push    ss
00008D8D 1F                         pop     ds
                                    ; 24/05/2019
                                    ;push   ss
                                    ;pop    es
                                    ; 05/12/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
                                    ; 17/12/2022
                                    ;push   ss
                                    ;pop    es
00008D8E 52                         push    dx                      ; remember address of arena
00008D8F 42                         inc     dx                      ; leave room for arena header
                                    ;mov    [330h],dx
00008D90 8916[3003]                 mov     [CurrentPDB],dx                 ; set current pdb
00008D94 31FF                       xor     di,di                   ; point es:di at end of memory
00008D96 8EC2                       mov     es,dx                   ; ...where psp will be
00008D98 31C0                       xor     ax,ax
                                    ;mov    cx,80h                  ; psp is 128 words
                                    ; 17/12/2022
00008D9A B180                       mov     cl,128 ; 28/06/2019
                                    ; 05/12/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
                                    ;mov    cx,128
00008D9C F3AB                       rep     stosw                   ; zero out psp area
```

```
00008D9E A1[4603]                       mov     ax,[ENDMEM]

                                    ; 17/12/2022
                                    ; cx = 0
00008DA1 E82189                     call    SETMEM                      ; build psp at dx; ax is memory size

                                    ; ds, es now point to PSP

00008DA4 16                         push    ss
00008DA5 1F                         pop     ds

                                    ;mov    di,24
00008DA6 BF1800                     mov     di,PDB.JFN_TABLE        ; es:di -> pdb_jfn_table in psp
00008DA9 31C0                       xor     ax,ax
00008DAB AB                         stosw
00008DAC AA                         stosb                           ; 0,1 and 2 are con device
00008DAD B0FF                       mov     al,0FFh
                                    ;mov    cx,FILPERPROC-3 ; 17
                                    ; 17/12/2022
                                    ; cx = 4
00008DAF B111                       mov     cl,FILPERPROC-3 ; 17
00008DB1 F3AA                       rep     stosb                   ; rest are unused

00008DB3 16                         push    ss
00008DB4 07                         pop     es
                                                                    ; must be set to print messages
00008DB5 8C1E[2C00]                 mov     [SFT_ADDR+2],ds

                                ; after this point the char device functions for con will work for
                                ; printing messages

                                    ; 24/04/2019 - Retro DOS v4.0

                                ; 12/05/2019
                                ;
                                ;write_version_msg:
                                ;
                                ;   ;if     (not ibm)
                                ;   ;mov    si,offset doscode:header
                                ;   mov     si,HEADER
                                ;outmes:
                                ;   ;lods   cs:byte ptr [si]
                                ;   cs
                                ;   lodsb
                                ;   cmp     al,"$"
                                ;   je      short outdone
                                ;   call    OUTT
                                ;   jmp     short outmes
                                ;outdone:
                                ;   push    ss                      ; out stomps on segments
                                ;   pop     ds
                                ;   push    ss
                                ;   pop     es
                                ;   ;endif

                                    ; at this point es is dosdata

                                    ; Fill in the segment addresses of sysinitvar and country_cdpg
                                    ; in sysinittable (ms_data.asm)

                                    ;mov    si,0D28h
00008DB9 BE[580D]                   mov     si,SysInitTable

                                    ; 17/12/2022
                                    ; ds = es = ss

                                    ; 23/03/2024
                                    ;;;
                                    ; 17/12/2022
                                    ;; 05/12/2022 (MSDOS 5.0 MSDOS.SYS compatibility)

                                    ;;mov   [es:si+6],es
                                    ;mov    [es:si+SYSI_EXT.Country_Tab+2],es
                                    ;;mov   [es:si+2],es
                                    ;mov    [es:si+SYSI_EXT.SysInitVars+2],es

00008DBC 8C4406                     mov     [si+SYSI_EXT.Country_Tab+2],es
00008DBF 8C4402                     mov     [si+SYSI_EXT.SysInitVars+2],es

                                    ; buffhead -> dosdata:hashinitvar

                                    ;mov    [es:BUFFHEAD+2],es      ; BUGBUG - unused, remove this
00008DC2 8C06[3A00]                 mov     [BUFFHEAD+2],es
                                    ;mov    si,offset dosdata:hashinitvar ; and all other references
                                    ;mov    si,6Dh
00008DC6 BE[6D00]                   mov     si,HASHINITVAR
                                    ;mov    [es:BUFFHEAD],si
00008DC9 8936[3800]                 mov     [BUFFHEAD],si

00008DCD 5A                             pop     dx                      ; restore address of arena

                                    ;mov    [032Ch+2],dx
00008DCE 8916[2E03]                     mov     [DMAADD+2],dx

                                    ;mov    [es:arena_head],dx
00008DD2 8916[2400]                 mov     [arena_head],dx
                                    ;;;

00008DD6 8EDA                           mov     ds,dx

                                    ;mov    byte [0],'Z'
00008DD8 C60600005A                 mov      byte [ARENA.SIGNATURE],arena_signature_end
                                        ;mov       word [1],0
00008DDD C70601000000               mov      word [ARENA.OWNER],arena_owner_system

00008DE3 36A1[4603]                     mov     ax,[ss:ENDMEM]
00008DE7 29D0                       sub     ax,dx
00008DE9 48                             dec     ax
                                    ;mov    [3],ax  ; 23/03/2024
00008DEA A30300                         mov     [ARENA.SIZE],ax

                                    ; point to sft 0

                                    ;mov    di,offset dosdata:sftabl + sftable
                                    ;mov    di,SFTABL+6
00008DED BF[D200]                   mov     di,SFTABL+SFT.SFTable
00008DF0 B80300                         mov     ax,3
00008DF3 AB                             stosw                       ; adjust refcount

                                    ; es:di is shared data area i.e., es:di -> dosdata:sysinttable

                                    ;mov    di,offset dosdata:sysinittable
                                    ;;mov   di,0D28h
                                    ;mov    di,0D58h        ; 23/03/2024 (PCDOS 7.1)
```

```
00008DF4 BF[580D]                   mov     di,SysInitTable

00008DF7 42                         inc     dx                      ; advance dx from arena to psp
00008DF8 8EDA                       mov     ds,dx                   ; point ds to psp

                                                                    ; pass the address os seg_reinit
                                                                    ; in dx
00008DFA BA[688E]                   mov     dx,seg_reinit
00008DFD B9[BF87]                   mov     cx,exepatch_start
00008E00 81E9[0000]                 sub     cx,_$STARTCODE          ; cx = (doscode - exepatch) - dosinit

00008E04 B8[E88A]                   mov     ax,SYSBUF
00008E07 2D[0000]                   sub     ax,_$STARTCODE          ; ax = size of doscode - dosinit

00008E0A 368B26[8405]                   mov     sp,[ss:USER_SP]           ; use ss override for next 2
00008E0F 368E16[8605]                   mov     ss,[ss:USER_SS]

00008E14 CB                             retf

                                ;
                                ; END OF DOSINIT
                                ;
                                ;--------------------------------------------------------------------------

                                CHARINIT:
                                    ; 11/04/2024
                                    ; 23/03/2024 - Retro DOS v5.0
                                    ; 24/04/2019 - Retro DOS v4.0
                                    ; 07/07/2018 - Retro DOS v3.0
                                    ;mov    byte [ss:035Ah],26 ; 1Ah ; MSDOS 6.22
00008E15 36C606[5A03]19                 MOV BYTE [SS:DEVCALL_REQLEN],DINITHL ; 25 ; PCDOS 7.1
                                    ;mov    byte [ss:035Bh],0
00008E1B 36C606[5B03]00                 MOV BYTE [SS:DEVCALL_REQUNIT],0
                                    ;mov    byte [ss:035Ch],0
00008E21 36C606[5C03]00                 MOV BYTE [SS:DEVCALL_REQFUNC],DEVINIT
                                    ;mov    word [ss:035Bh],0
00008E27 36C706[5D03]0000               MOV WORD [SS:DEVCALL_REQSTAT],0
00008E2E 06                             PUSH    ES
00008E2F 53                             PUSH    BX
00008E30 50                             PUSH    AX
00008E31 BB[5A03]                       MOV BX,DEVCALL
                                        ;PUSH   CS
00008E34 16                         PUSH    SS ; 30/04/2019
00008E35 07                             POP ES
00008E36 E85CC4                         CALL    DEVIOCALL2
00008E39 58                         POP     AX
00008E3A 5B                             POP BX
00008E3B 07                             POP ES
00008E3C C3                             RETN

                                ; 25/04/2019 - Retro DOS v4.0

                                ;--------------------------------------------------------------------------
                                ;
                                ;   check_XMM: routine to check presence of XMM driver
                                ;
                                ;   Exit:   Sets up the XMM entry point in XMMcontrol in DOSDATA
                                ;
                                ;   USED:   none
                                ;
                                ;--------------------------------------------------------------------------

                                check_XMM: ; proc near
                                ;
                                ; determine whether or not an XMM driver is installed
                                ;
00008E3D 50                         push    ax
                                    ;mov    ax,(XMM_MULTIPLEX<<8)+XMM_INSTALL_CHECK
00008E3E B80043                     mov     ax,4300h
00008E41 CD2F                       int     2Fh
                                            ; - Multiplex - XMS - INSTALLATION CHECK
                                            ; Return: AL = 80h XMS driver installed
                                            ; AL <> 80h no driver
00008E43 3C80                       cmp     al,80h                  ; Q: installed
00008E45 751D                       jne     short cXMM_no_driver    ; N: set error, quit
                                ;
                                ; get the XMM control functions entry point, save it, we
                                ; need to call it later.
                                ;
00008E47 53                         push    bx
00008E48 52                         push    dx
00008E49 1E                         push    ds
00008E4A 06                         push    es
                                    ;mov    ax,(XMM_MULTIPLEX<<8)+XMM_FUNCTION_ADDR
00008E4B B81043                     mov     ax,4310h
00008E4E CD2F                       int     2Fh
                                            ; - Multiplex - XMS - GET DRIVER ADDRESS
                                            ; Return: ES:BX -> driver entry point

00008E50 2E8E1E[0700]               mov     ds,[cs:DosDSeg]

00008E55 891E[7B10]                 mov     [XMMcontrol],bx
00008E59 8C06[7D10]                 mov     [XMMcontrol+2],es
                                cXMMexit:
00008E5D F8                         clc
00008E5E 07                         pop     es
00008E5F 1F                         pop     ds
00008E60 5A                         pop     dx
00008E61 5B                         pop     bx
00008E62 58                         pop     ax
00008E63 C3                         retn                            ; done
                                ;
                                ; set carry if XMM driver not present
                                ;
                                cXMM_no_driver:
00008E64 F9                         stc
00008E65 58                         pop     ax
00008E66 C3                         retn

                                ;--------------------------------------------------------------------------
                                ;
                                ; Procedure Name : seg_reinit
                                ;
                                ; Inputs     : ES has final dos code location
                                ;              AX = 0 / 1
                                ;
                                ; Outputs    : Patch in the sharer and other tables with seg in ES
                                ;              if AX =0
                                ;                 if first entry
                                ;                   patch segment & offset of vectors with stub
                                ;                   and stub with segment in ES
                                ;                 else
                                ;                   patch stub with segment in ES
                                ;
```

```
                                ;             else if AX = 1
                                ;                  patch segment of vectors with segment in ES
                                ;
                                ; NOTE              : This routine can be called at most twice!
                                ;
                                ; Regs Mod.  : es, ax, di, cx, bx
                                ;-----------------------------------------------------------------------------

00008E67 00                     num_entry: db      0              ; keeps track of the # of times this routine
                                                                  ; has been called. (0 or 1)

                                    ; 04/11/2022 - Retro DOS v4.0 (ref: MSDOS 5.0)
                                    ; MSDOS 5.0 MSDOS.SYS - DOSCODE:0BAB7h
                                    ; 25/05/2019 - Retro DOS v4.0 (ref: MSDOS 6.21)
                                    ; MSDOS 6.21 MSDOS.SYS - DOSCODE:0BDA5h

                                    ; 23/03/2024 - Retro DOS v5.0 (ref: PCDOS 7.1)
                                    ; PCDOS 7.1 IBMDOS.COM - DOSCODE:0D07Eh

                                seg_reinit: ; proc  far
00008E68 1E                         push    ds

00008E69 2E8E1E[0700]               mov     ds,[cs:DosDSeg]

00008E6E E89300                     call    patch_misc_segments   ; patch in segments for sharer and
                                                                  ; other tables with seg in ES.

                                    ; 17/12/2022
                                    ; cx = 0
00008E71 39C8                       cmp     ax,cx ; 0
                                    ;cmp    ax,0
00008E73 754A                       jne     short patch_vec_seg   ; patch vectors with segment in es
                                    ; 23/03/2024
                                    ;or     ax, ax
                                    ;jnz    short patch_vec_seg

                                    ; 23/03/2024
                                    ;cmp    [cs:num_entry],al ; 0
                                    ; 17/12/2022
00008E75 2E380E[678E]               cmp     [cs:num_entry],cl ; 0
                                    ;cmp    byte [cs:num_entry],0 ; Q: is it the first call to this
00008E7A 7508                       jne     short second_entry    ; N: just patch the stub with
                                                                  ;    segment in ES
                                                                  ; Y: patch the vectors with stub
00008E7C 8CD8                       mov     ax,ds
00008E7E E84B00                     call    patch_vec_segments    ; patch the segment of vectors
00008E81 E8CA00                     call    patch_offset          ; patch the offsets of vectors
                                                                  ; with those in the stub.
                                    ; 17/12/2022
                                    ; cx = 0
                                second_entry:
00008E84 8CC0                       mov     ax,es                 ; patch the stub with segment in es

                                    ;mov    di,OFFSET DOSDATA:DOSINTTABLE
                                    ;;mov   di,1062h       ; (same table addr for MSDOS 5.0 and MSDOS 6.21)
                                    ;mov    di,0F7Ah ; 23/03/2024 ; PCDOS 7.1
00008E86 BF[7A0F]                   mov     di,DOSINTTABLE

                                    ; 17/12/2022
                                    ; cx = 0
                                    ;;mov   cx,9
                                    ;mov    cl,9
                                    ; 23/03/2024 - Retro DOS v5.0 (PCDOS 7.1 IBMDOS.COM)
                                    ;;;
                                    ;mov    cx,8
00008E89 B108                       mov     cl,8
                                    ;;;
00008E8B 1E                         push    ds
00008E8C 07                         pop     es                    ; es:di -> DOSINTTABLE

                                dosinttabloop:
                                    ;add    di,2
                                    ; 19/06/2023
00008E8D 47                         inc     di
00008E8E 47                         inc     di
00008E8F AB                         stosw
00008E90 E2FB                       loop    dosinttabloop

                                ; For ROMDOS, this routine will only be called when the DOS wants to
                                ; use the HMA, so we don't want to check CS

                                ;ifndef ROMDOS
00008E92 3D00F0                     cmp     ax,0F000h             ; Q: is the DOS running in the HMA
00008E95 722D                       jb      short sr_done         ; N: done
                                ;endif
00008E97 E8A3FF                     call    check_XMM             ; Y: set up the XMS entry point
00008E9A 7228                       jc      short sr_done         ; failed to set up XMS do not do
                                                                  ; A20 toggling in the stub.
                                    ; 17/12/2022
                                    ; cx = 0
00008E9C E82A01                     call    patch_in_nops         ; enable the stub to check A20 state
                                ; M021-
                                    ;;mov   byte [1211h],1
                                    ;mov    byte [0D66h],1 1 ; 23/03/2024 ; PCDOS 7.1
00008E9F C606[660D]01               mov     byte [DosHasHMA],1    ; set flag telling DOS control of HMA

                                                                  ; set pointer to the routine that
                                                                  ; patches buggy exepacked code.
                                    ;mov    [FixExePatch],offset DOSCODE:ExePatch
00008EA4 C706[670D][0F89]           mov     word [FixExePatch],ExePatch
                                                                  ; M068: set pointer to the routine
                                                                  ; M068: that detects copy protected
                                                                  ; M068: apps
                                    ;mov    [ChkCopyProt],offset DOSCODE:IsCopyProt
00008EAA C706[6100][B28A]           mov     word [ChkCopyProt],IsCopyProt

                                    ; 23/03/2024 - PCDOS 7.1 IBMDOS.COM - DOSCODE:0D0C7h
                                    ;            (MSDOS 6.22 MSDOS.SYS - DOSCODE:0BDF1h)
00008EB0 E844FC                     call    WhatCPUType
00008EB3 3C01                       cmp     al,1
00008EB5 750D                       jne     short sr_done  ; we need Rational Patch only on 286 systems
00008EB7 C706[690D][398A]           mov     word [RationalPatchPtr],RationalPatch

                                    ; 19/09/2023
00008EBD EB05                       jmp     short sr_done

                                    ; 23/03/2024
                                patch_vec_seg:                          ; patch vectors with segment in es
00008EBF 8CC0                       mov     ax,es
00008EC1 E80800                     call    patch_vec_segments    ; patch in DOSCODE for the segments
                                                                  ; NOTE we don't have to patch the
                                                                  ; offsets as they have been already
                                                                  ; set to the doscode offsets at
                                                                  ; DOSINIT.
```

```
                                sr_done:
00008EC4 2EC606[678E]01             mov     byte [cs:num_entry],1
00008ECA 1F                         pop     ds
00008ECB CB                         retf    ; ! far return !

                                ;--------------------------------------------------------------------------
                                ;
                                ; Procedure Name : patch_vec_segments
                                ;
                                ; Inputs     : ax -> has segment address to patch in
                                ;              ds -> DOSDATA
                                ;
                                ; Outputs    : Patches in AX as the segment for the following vectors:
                                ;
                                ;                   0,20-28,3a-3f
                                ;
                                ; Regs. Mod.        : DI,CX,DX,AX
                                ;
                                ;--------------------------------------------------------------------------

                                patch_vec_segments:

00008ECC 06                         push    es

00008ECD 31C9                       xor     cx,cx ; 0
00008ECF 8EC1                       mov     es,cx

                                    ;mov    di,82h
00008ED1 BF8200                     mov     di,INTBASE+2            ; di -> segment of int 20h vector

00008ED4 26A30200                   mov     [es:2],ax               ; segment of default divide trap handler

                                                                    ; set vectors 20h & 21h
                                    ; 04/11/2022
                                    ;mov    cx,2
                                    ; 17/12/2022
                                    ;mov    cl,2
                                ps_set1:
00008ED8 AB                         stosw   ; int 20h segment
                                    ;add    di,2
                                    ; 17/12/2022
00008ED9 47                         inc     di
00008EDA 47                         inc     di
                                    ;loop   ps_set1

                                    ; 17/12/2022
00008EDB AB                         stosw   ; int 21h segment
                                    ;inc    di
                                    ;inc    di

                                    ;add    di,4                    ; skip int 22h vector
00008EDC 83C706                     add     di,6 ; *

00008EDF AB                         stosw                           ; set int 23h
00008EE0 83C706                     add     di,6                    ; skip int 24h

                                                                    ; set vectors 25h-28h and 2Ah-3Fh
                                    ; 04/11/2022
                                    ;mov    cx,4                    ; set 4 segments
                                    ; 17/12/2022
00008EE3 B104                       mov     cl,4
                                ps_set2:
00008EE5 AB                         stosw
                                    ;add    di,2
                                    ; 17/12/2022
00008EE6 47                         inc     di
00008EE7 47                         inc     di
00008EE8 E2FB                       loop    ps_set2

00008EEA 83C704                     add     di,4                    ; skip int 29h vector (fast con) as it may
                                                                    ;   already be set.
                                    ; 04/11/2022
                                    ;mov    cx,6                    ; set 6 segs (skip 2 between each)
                                    ; 17/12/2022
00008EED B106                       mov     cl,6                    ;  set segs for ints 2Ah-2Fh
                                ps_set3:
00008EEF AB                         stosw
                                    ;add    di,2
                                    ; 17/12/2022
00008EF0 47                         inc     di
00008EF1 47                         inc     di
00008EF2 E2FB                       loop    ps_set3

                                ; 30h & 31H is the CPM call entry point whose segment address is set up by
                                ; below. So skip it.

00008EF4 83C708                     add     di,8                    ; skip vector 30h & 31h

                                    ; 04/11/2022
                                    ;mov    cx,14                   ; set 14 segs (skip 2 between each)
                                    ; 17/12/2022
00008EF7 B10E                       mov     cl,14                   ;  sets segs for ints 32h-3Fh
                                ps_set4:
00008EF9 AB                         stosw
                                    ;add    di,2
                                    ; 17/12/2022
00008EFA 47                         inc     di
00008EFB 47                         inc     di
00008EFC E2FB                       loop    ps_set4

                                ; set offset of int2f

                                ;if installed
                                ;   mov     word ptr es:[02fh * 4],offset doscode:int2f
                                ;endif
                                    ;mov    [es:0C3h],ax
00008EFE 26A3C300                   mov     [es:ENTRYPOINT+3],ax
                                    ; 17/12/2022
                                    ; cx = 0
00008F02 07                         pop     es
00008F03 C3                         retn

                                ;--------------------------------------------------------------------------
                                ;
                                ; Procedure Name : patch_misc_segments
                                ;
                                ; Inputs     : es = segment to patch in
                                ;              ds = dosdata
                                ;
                                ; outputs    : patches in the sharer and other tables in the dos
                                ;              with right dos code segment in es
                                ;
                                ; Regs Mod   : DI,SI,CX
                                ;
```

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

                                patch_misc_segments:

00008F04 53                         push    bx
00008F05 06                         push    es
00008F06 50                         push    ax

00008F07 8CC0                       mov     ax,es                   ; ax - > DOS segment

00008F09 1E                         push    ds
00008F0A 07                         pop     es                      ; es -> DOSDATA

                                ; initialize the jump table for the sharer...

                                    ;mov    di,offset dosdata:jshare
                                    ;mov    di,90h
00008F0B BF[9000]                   mov     di,JShare
                                    ;;mov   bx,[0AAAh]
                                    ;mov    bx,[0AA3h] ; 23/03/2024 ; PCDOS 7.1
00008F0E 8B1E[A30A]                 mov     bx,[TEMP_DOSLOC]        ; bx = location to which the share
                                                                    ; table was patched during the first
                                                                    ; call to this routine
00008F12 B90F00                     mov     cx,15
                                jumptabloop:
                                    ;add    di,2                    ; skip offset
                                    ; 17/12/2022
00008F15 47                         inc     di
00008F16 47                         inc     di
00008F17 83FBFF                     cmp     bx,-1 ; 0FFFFh          ; Q: is this called for the 1st time
00008F1A 7405                       je      short share_patch       ; Y: patch in sharer table
                                                                    ; N:
00008F1C 263B1D                     cmp     bx,[es:di]              ; Q: has share been installed
00008F1F 7501                       jne     short no_share_patch    ; Y: don't patch in sharer table
                                share_patch:
00008F21 AB                         stosw                           ; drop in segment
                                no_share_patch:
00008F22 E2F1                       loop    jumptabloop
                                                                    ; BUGBUG patching the country info
                                                                    ; with dosdata can be done inline
                                                                    ; in dosinit.
                                                                    ; for dos 3.3 country info
                                                                    ; table address

                                    ;mov    si,offset dosdata:country_cdpg
                                    ;mov    si,122Ah
00008F24 BE[2A12]                   mov     si,COUNTRY_CDPG
                                                                    ; initialize double word
                                                                    ; pointers with dosdata in ds
                                    ;mov    [si+4Fh],ds
                                    ;mov    [si+54h],ds
                                    ;mov    [si+59h],ds
                                    ;mov    [si+5Eh],ds
                                    ;mov    [si+80h],ds
                                    ;mov    [si+63h],ds
00008F27 8C5C4F                     mov     [si+DOS_CCDPG.ccUcase_ptr+2],ds
00008F2A 8C5C54                     mov     [si+DOS_CCDPG.ccFileUcase_ptr+2],ds
00008F2D 8C5C59                     mov     [si+DOS_CCDPG.ccFileChar_ptr+2],ds
00008F30 8C5C5E                     mov     [si+DOS_CCDPG.ccCollate_ptr+2],ds
00008F33 8C9C8000                   mov     [si+DOS_CCDPG.ccMono_ptr+2],ds
00008F37 8C5C63                     mov     [si+DOS_CCDPG.ccDBCS_ptr+2],ds

                                                                    ; fastopen routines are in doscode
                                                                    ; so patch with doscode seg in ax

                                    ;mov    si,offset dosdata:fastopentable
                                    ;mov    si,0D30h
00008F3A BE[3C11]                   mov     si,FastOpenTable

                                    ; 17/12/2022
                                    ; bx = [TEMP_DOSLOC]
00008F3D 83FBFF                     cmp     bx,-1
                                    ;cmp    word [TEMP_DOSLOC],-1 ; Q: first time
00008F40 7405                       je      short fast_patch        ; Y: patch segment
                                    ;mov    cx,[TEMP_DOSLOC]
                                                                    ; Q: has fastopen patched in it's
                                                                    ;    segment
                                    ; 17/12/2022
00008F42 3B5C04                     cmp     bx,[si+fastopen_entry.name_caching+2]
                                    ;;cmp   cx,[si+4]
                                    ;cmp    cx,[si+fastopen_entry.name_caching+2]
00008F45 7503                       jne     short no_fast_patch     ; Y: don't patch in doscode seg

                                fast_patch:
                                    ;mov    [si+4],ax
00008F47 894404                     mov     [si+fastopen_entry.name_caching+2],ax
                                no_fast_patch:
                                    ; 17/12/2022
                                    ; cx = 0
00008F4A 58                         pop     ax
00008F4B 07                         pop     es
00008F4C 5B                         pop     bx

00008F4D C3                         retn

                                ;--------------------------------------------------------------------------
                                ;
                                ; Procedure Name : patch_offset
                                ;
                                ; Inputs     : NONE
                                ;
                                ; Outputs    : Patches in the offsets in the low_mem_stub for vectors
                                ;              0,20-28,3a-3f, and 30,31
                                ;
                                ;
                                ; Regs. Mod  : AX,DI,CX
                                ;--------------------------------------------------------------------------

                                patch_offset:
00008F4E 06                         push    es              ; preserve es

00008F4F 31C0                       xor     ax,ax
00008F51 8EC0                       mov     es,ax
                                                            ; set default divide trap address
                                    ;mov    word ptr es:[0],offset dosdata:ldivov
                                    ;;mov   word [es:0],108Ah
                                    ;mov    word [es:0],0F9Eh ; 23/03/2024 ; PCDOS 7.1
00008F53 26C7060000[9E0F]           mov     word [es:0],ldivov

                                    ;mov    di,80h
00008F5A BF8000                     mov     di,INTBASE      ; di-> offset of int 20 handler
                                    ;mov    ax,offset dosdata:lirett
                                    ;mov    ax,10DAh
00008F5D B8[6E10]                   mov     ax,lirett
```

```
                                                        ; set vectors 20h & 21h to point to iret.
                                    ; 17/12/2022
                                    ; cx = 0

                                    ;mov    cx,2            ; set 2 offsets (skip 2 between each)
                                po_iset1:
00008F60 AB                         stosw   ; int 20h offset
                                    ;add    di,2 ; *
                                    ;loop   po_iset1
                                    ; 17/12/2022
00008F61 47                         inc     di
00008F62 47                         inc     di
00008F63 AB                         stosw   ; int 21h offset

                                    ;add    di,4            ; skip vector 22h
                                    ; 17/12/2022
00008F64 83C706                     add     di,6 ; *

00008F67 AB                         stosw                   ; set offset of 23h
                                    ;add    di,6            ; skip 24h
                                    ; 19/09/2023
00008F68 83C712                     add     di,18           ; skip 23h segment and int 24-25-26-27h

                                                            ; set vectors 25-28 and 2a-3f to iret.
                                    ; 04/11/2022
                                    ;mov    cx,4            ; set 4 offsets (skip 2 between each)
                                    ; 19/09/2023
                                    ; 17/12/2022
                                    ;mov    cl,4            ; sets offsets for ints 25h-28h
                                po_iset2:
00008F6B AB                         stosw           ; set offset for int 28h ; 19/09/2023
                                    ;add    di,2
                                    ; 19/09/2023
                                    ; 17/12/2022
                                    ;inc    di
                                    ;inc    di
                                    ; 19/09/2023
                                    ;loop   po_iset2

                                    ;add    di,4            ; skip vector 29h
                                    ; 19/09/2023
00008F6C 83C706                     add     di,6    ; skip int 28h segment and int 29h ; 19/09/2023

                                    ; 04/11/2022
                                    ;mov    cx,6            ; set 6 offsets (skip 2 between each)
                                    ; 17/12/2022
                                    ;mov    cl,6            ; sets offsets for ints 2Ah-2Fh
00008F6F B105                       mov     cl,5            ; sets offsets for ints 2Ah-2Eh
                                po_iset3:
00008F71 AB                         stosw
                                    ;add    di,2
                                    ; 17/12/2022
00008F72 47                         inc     di
00008F73 47                         inc     di
00008F74 E2FB                       loop    po_iset3

                                ; 30h & 31H is the CPM call entry point whose offset address is set up by
                                ; below. So skip it.

                                    ;add    di,8            ; skip vector 30h & 31h
                                    ; 17/12/2022
00008F76 83C70C                     add     di,12           ; skip vector 2Fh, 30h & 31h

                                    ; 04/11/2022
                                    ;mov    cx,14           ; set 14 offsets (skip 2 between each)
                                                            ;  sets offsets for ints 32h-3Fh
                                    ; 17/12/2022
00008F79 B10E                       mov     cl,14 ; 26/06/2019
                                po_iset4:
00008F7B AB                         stosw
                                    ;add    di,2
                                    ; 17/12/2022
00008F7C 47                         inc     di
00008F7D 47                         inc     di
00008F7E E2FB                       loop    po_iset4

                                ;if installed
                                    ;mov    word ptr es:[02fh * 4],offset dosdata:lint2f
                                    ;;mov   word [es:0BCh],10C6h ; (MSDOS 5.0 & 6.21)
                                    ;mov    word [es:0BCh],0FDAh ; PCDOS 7.1 ; 23/03/2024
00008F80 26C706BC00[DA0F]           mov     word [es:(2Fh*4)],lint2f
                                ;endif

                                ; set up entry point call at vectors 30-31h
                                    ;mov    byte [es:0C0h],0EAh
00008F87 26C606C000EA               mov     byte [es:ENTRYPOINT],mi_long_jmp
                                    ;mov    word [es:0C1h],10D0h ;; 0FE4h ; 23/03/2024 (PCDOS 7.1)
00008F8D 26C706C100[E40F]           mov     word [es:ENTRYPOINT+1],lcall_entry

                                                                    ; 19/09/2023
                                    ;mov    word [es:80h],1094h ;; 0FA8h ; 23/03/2024
00008F94 26C7068000[A80F]           mov     word [es:addr_int_abort],lquit              ; int 20h
                                    ;mov    word [es:84h],109Eh ;; 0FB2h
00008F9B 26C7068400[B20F]           mov     word [es:addr_int_command],lcommand  ; int 21h
                                    ;mov    word [es:94h],10A8h ;; 0FBCh
00008FA2 26C7069400[BC0F]           mov     word [es:addr_int_disk_read],labsdrd ; int 25h
                                    ;mov    word [es:98h],10B2h ;; 0FC6h
00008FA9 26C7069800[C60F]           mov     word [es:addr_int_disk_write],labsdwrt      ; int 26h
                                    ;mov    word [es:9Ch],10BCh ;; 0FD0h
00008FB0 26C7069C00[D00F]           mov     word [es:addr_int_keep_process],lstay_resident      ; int 27h

                                    ; 17/12/2022
                                    ; CX = 0
00008FB7 07                         pop     es              ; restore es
00008FB8 C3                         retn

                                ;--------------------------------------------------------------------------
                                ;
                                ;   Procedure Name :        patch_in_nops
                                ;
                                ;   Entry          :        ES -> DOSDATA
                                ;
                                ;   Regs Mod       :        cx, di
                                ;
                                ;   Description:
                                ;           This routine patches in 2 nops at the offsets specified in
                                ;   patch_table. This basically enables the low mem stub to start
                                ;   making XMS calls.
                                ;
                                ;--------------------------------------------------------------------------

                                    ; 04/11/2022
                                    ; MSDOS 5.0 MSDOS.SYS - DOSCODE:0BC50h
```

```
52067
52068                                   ; 23/03/2024
52069                                   ; MSDOS 6.22 MSDOS.SYS - DOSCODE:0BF42h
52070                                   ; 23/03/2024 - Retro DOS v5.0
52071                                   ; PCDOS 7.1 IBMDOS.COM - DOSCODE:0D1E9h
52072
52073                                 patch_table:       ; label byte
52074                                     ;dw    offset dosdata:i0patch
52075                                     ;dw    offset dosdata:i20patch
52076                                     ;dw    offset dosdata:i21patch
52077                                     ;dw    offset dosdata:i25patch
52078                                     ;dw    offset dosdata:i26patch
52079                                     ;dw    offset dosdata:i27patch
52080                                     ;dw    offset dosdata:i2fpatch
52081                                     ;dw    offset dosdata:cpmpatch
52082 00008FB9 [9E0F]                   dw      i0patch
52083 00008FBB [A80F]                   dw      i20patch
52084 00008FBD [B20F]                   dw      i21patch
52085 00008FBF [BC0F]                   dw      i25patch
52086 00008FC1 [C60F]                   dw      i26patch
52087 00008FC3 [D00F]                   dw      i27patch
52088 00008FC5 [DA0F]                   dw      i2fpatch
52089 00008FC7 [E40F]                   dw      cpmpatch
52090
52091                                 patch_table_size equ ($-patch_table)/2
52092
52093                                 patch_in_nops:
52094 00008FC9 50                       push    ax
52095 00008FCA 56                       push    si
52096 00008FCB BE[B98F]                 mov     si,patch_table
52097 00008FCE B89090                   mov     ax,9090h ; nop, nop
52098                                   ; 17/12/2022
52099                                   ; cx = 0
52100                                   ;mov    cx,8
52101                                   ;mov    cx,patch_table_size ; 8
52102 00008FD1 B108                     mov     cl,patch_table_size ; 8
52103                                 pin_loop:
52104 00008FD3 2E8B3C                   mov     di,[cs:si]
52105 00008FD6 AB                       stosw
52106                                   ;add    si,2
52107                                   ; 17/12/2022
52108 00008FD7 46                       inc     si
52109 00008FD8 46                       inc     si
52110 00008FD9 E2F8                     loop    pin_loop
52111 00008FDB 5E                       pop     si
52112 00008FDC 58                       pop     ax
52113 00008FDD C3                       retn
52114
52115                                 ; 05/12/2022 (MSDOS 5.0 MSDOS.SYS compatibility)
52116                                 ; ---------------------------------------------------------------------------
52117                                 ; MSDOS 5.0 - MSDOS.SYS offset BC77h, file offset 7EA7h
52118                                 ; ---------------------------------------------------------------------------
52119                                 ; 23/03/2024
52120                                 ; (MSDOS 6.22 MSDOS.SYS - DOSCODE:0BF69h)
52121                                 ; PCDOS 7.1 IBMDOS.COM - DOSCODE:0D20Fh
52122
52123                                     ; 05/12/2022 - temporary ; (paragraph alinment)
52124                                 DOSCODE_END:
52125
52126                                     ; 23/03/2024
52127                                     ;;times 9 db 0  ; db 9 dup(0)
52128                                     ;; 18/12/2022
52129                                     ;dw    0       ; times 2 db 0
52130
52131                                     ; 23/03/2024
52132                                     ;times  7 db 0  ; MSDOS 6.22 MSDOS.SYS
52133
52134                                     ; 23/03/2024 - Retro DOS v5.0
52135                                     ;db    0       ; PCDOS 7.1 IBMDOS.COM
52136                                     ; 01/07/2024
52137                                     ;times  12 db 0
52138
52139                                     ; 01/07/2024 - Retro DOS v5.0
52140                                     ; PCDOS 7.1 IBMDOS.COM - DOSCODE:0D0Fh
52141 00008FDE 00                       db      0
52142
52143                                 ;align 16
52144                                     ; DOSCODE:BC80h       (MSDOS 5.0 MSDOS.SYS file offset 7EB0h)
52145                                     ; MSDOS.SYS file offset: 32432 (start of DOSDATA)
52146
52147                                     ; 23/03/2024
52148                                     ; PCDOS 7.1 IBMDOS.COM - DOSCODE:0D210h
52149                                     ; (MSDOS 6.22 MSDOS.SYS - DOSCODE:0BF70h)
52150
52151                                 ; ---------------------------------------------------------------------------
52152
52153                                 ;memstrt label word
52154                                 ; ---------------------------------------------------------------------------
52155                                 ; MSDOS 6.21 - MSDOS.SYS offset BF69h, file offset 8189h
52156                                 ; ---------------------------------------------------------------------------
52157
52158                                 MEMSTRT: ; 25/04/2019 - Retro DOS v4.0
52159
52160                                 ; if not ROMDOS, then we close the dos code segment, otherwise we close
52161                                 ; the dos initialization segment
52162
52163                                 ;ifndef ROMDOS
52164
52165                                 ;doscode ends
52166
52167                                 ;else
52168
52169                                 ;;dosinitseg ends
52170
52171                                 ;endif ; ROMDOS
52172
52173                                 ;============================================================================
52174
52175                                 ; DPUBLIC <ParaRound, cXMM_no_driver, cXMMexit, char_init_loop, charinit>
52176                                 ; DPUBLIC <check_XMM, continit, dosinttabloop, endlist>
52177                                 ; DPUBLIC <initiret, iset1, iset2, jumptabloop, nxtentry>
52178                                 ; DPUBLIC <notmax,  patch_offset, perdrv>
52179                                 ; DPUBLIC <perunit, po_iset1, po_iset2, po_iset3>
52180                                 ; DPUBLIC <ps_set1, ps_set2, ps_set3, seg_reinit>
52181                                 ; DPUBLIC <sr_done, version_fake_table, xxx>
52182
52183                                 ;; burası doscode sonu
52184
52185                                 ;============================================================================
52186                                 ; DOSDATA
52187                                 ;============================================================================
52188                                 ; 29/04/2019 - Retro DOS 4.0
52189                                 ; 24/03/2024 - Retro DOS 5.0
52190
```

```
                                ;[BITS 16]

                                ;[ORG 0]

                                ; 25/04/2019 - Retro DOS v4.0

                                ;============================================================================
                                ; DOSDATA - MSDOS 6.21 - MSDOS.SYS Offset 0BF70h, file offset 8190h
                                ;============================================================================

                                ;align 16
                                    ; DOSDATA (MSDOS.SYS kernel DATA) segment starts here...
                                    ; (4970 bytes for MSDOS 6.21)
                                    ; (4976 bytes for Retro DOS v4.0, 25/05/2019 modification.)

                                ;============================================================================
                                ; MSCONST.ASM (MSDOS 6.0, 1991)
                                ;============================================================================
                                ; 03/11/2022 - Retro DOS 4.0 (Modified MSDOS 5.0 MSDOS.SYS)
                                ; 25/04/2019 - Retro DOS 4.0 (MSDOS 6.21)
                                ; 16/07/2018 - Retro DOS 3.0

                                ;Break <Initialized data and data used at DOS initialization>
                                ;----------------------------------------------------------------------------

                                ; We need to identify the parts of the data area that are relevant to tasks
                                ; and those that are relevant to the system as a whole. Under 3.0, the system
                                ; data will be gathered with the system code. The process data under 2.x will
                                ; be available for swapping and under 3.0 it will be allocated per-process.
                                ;
                                ; The data that is system data will be identified by [SYSTEM] in the comments
                                ; describing that data item.

                                ;DOSDATA SEGMENT

                                ; 04/11/2022
                                ;[ORG 0]
                                ; ---------------------------------------------------------------------------
                                ; 04/11/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
                                ; ---------------------------------------------------------------------------
                                ; DOSDATA segment start offset from beginning of MSDOS.SYS file: 32432 (7EB0h)
                                ; (3DD0h+7EB0h = 0BC80h) - for MSDOS 5.0 kernel file -
                                ; ---------------------------------------------------------------------------

                                ; 04/11/2022

                                ;DOSDATA:0000h

00008FDF 90                     align 16

                                ; ---------------------------------------------------------------------------
                                ; 06/12/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
                                ; ---------------------------------------------------------------------------

                                segment .data  vstart=0 ; 06/12/2022

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

                                ; 06/12/2022
                                ;DOSDATASTART equ $
                                DOSDATASTART:

                                ;hkn; add 4 bytes to get correct offsets since jmp has been removed in START

                                    ;; 03/11/2022
                                    ;jmp    DOSINIT        ; MSDOS 5.0 - MSDOS.SYS (DOSDATA:0000h)

                                    ; 04/11/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
                                    ;db     4 dup (?)
00000000 00<rep 4h>                 times  4 db 0

                                    ; 29/04/2019 - Retro DOS v4.0 modification
                                    ;dw     _$STARTCODE ; DOSCODE offset and/or size of DOSDATA
                                    ;dw     0

                                    ;EVEN

                                ;align 2

                                ; WANGO!!! The following word is used by SHARE and REDIR to determin data
                                ; area compatability. This location must be incremented EACH TIME the data
                                ; area here gets mucked with.
                                ;
                                ; Also, do NOT change this position relative to DOSDATA:0.

                                MSCT001S:   ; LABEL BYTE

                                DataVersion:
00000004 0100                       dw      1       ;AC000; [SYSTEM] version number for DOS DATA

                                ;hkn; add 8 bytes to get correct offsets since BugTyp, BugLev and "BUG " has
                                ;hkn; been removed to DOSCODE above

                                ;M044
                                ; First part of save area for saving last para of Window memory

                                WinoldPatch1:       ; db 8 dup (?) ;M044
00000006 00<rep 8h>                 times  8 db 0

                                    ; MSDOS 6.21 DOSDATA:000Eh
                                MYNUM:                      ; Offset 000Eh
0000000E 0000                       dw      0       ; [SYSTEM] A number that goes with MYNAME
                                FCBLRU:             ; [SYSTEM] LRU count for FCB cache
00000010 0000                       dw      0
                                OpenLRU:
00000012 0000                       dw      0       ; [SYSTEM] LRU count for FCB cache opens
                                OEM_HANDLER:
00000014 FFFFFFFF                   dd      -1      ; [SYSTEM] Pointer to OEM handler code

                                ;   BUGBUG - who uses LeaveAddr?  What if we want to rework the
                                ;;                  way that we leave DOS???? - jgl

                                LeaveAddr:
00000018 [F603]                     dw      LeaveDOS  ; <<OFFSET DOSCODE:LeaveDOS>> ; [SYSTEM]
                                RetryCount:
0000001A 0300                       dw      3       ; [SYSTEM] Share retries
                                RetryLoop:
0000001C 0100                       dw      1       ; [SYSTEM] Share retries
                                LastBuffer:
0000001E FFFFFFFF                   dd      -1      ; [SYSTEM] Buffer queue recency pointer
                                CONTPOS:
00000022 0000                       dw      0       ; [SYSTEM] location in buffer of next read
                                arena_head:
```

```
00000024 0000                           dw      0       ; [SYSTEM] Segment # of first arena in memory

                                ;; 16/07/2018
                                ;;*****************************************************************************
                                ;; NOTE: INT 21H AH=52H !  (http://stanislavs.org/helppc/int_21-52.html)
                                ;;*****************************************************************************
                                ;; INT 21,52 - Get Pointer to DOS "INVARS" (Undocumented)
                                ;;
                                ;;  AH = 52h
                                ;;
                                ;;  on return:
                                ;;  ES:BX = pointer to DOS "invars", a table of pointers used by DOS.
                                ;;          Known "invars" fields follow (varies with DOS version):
                                ;;
                                ;;  Offset Size             Description
                                ;;
                                ;;   -12   word   sharing retry count (DOS 3.1-3.3)
                                ;;   -10   word   sharing retry delay  (DOS 3.1-3.3)
                                ;;    -8   dword  pointer to current disk buffer (DOS 3.x)
                                ;;    -4   word   pointer in DOS code segment of unread CON input;
                                ;;                0 indicates no unread input (DOS 3.x)
                                ;;    -2   word   segment of first Memory Control Block (MCB)
                                ;;    00   dword  pointer to first DRIVE PARAMETER TABLE (A:) in chain
                                ;;    04   dword  pointer to DOS System File Table (SFT)
                                ;;    08   dword  pointer to $CLOCK device driver
                                ;;    0C   dword  pointer to CON device driver
                                ;;    10   byte   number of logical drives in system
                                ;;    11   word   maximum bytes/block of any block device
                                ;;    13   dword  pointer to DOS cache buffer header
                                ;;    17 18bytes  NUL device header, first 4 bytes of device header
                                ;;                point to the next device in device chain
                                ;;
                                ;;*****************************************************************************

                                ; The following block of data is used by SYSINIT.
                                ; Do not change the order or size of this block

                                ;SYSINITVAR:
                                ;-----------------------------------------------------------------------------
                                SYSINITVARS:
                                DPBHEAD:
00000026 00000000                   dd      0       ; [SYSTEM] Pointer to head of DPB-FAT list
                                SFT_ADDR:
0000002A [CC000000]                 dd      SFTABL  ; [SYSTEM] Pointer to first SFT table
                                BCLOCK:
0000002E 00000000                   dd      0       ; [SYSTEM] The CLOCK device
                                BCON:
00000032 00000000                   dd      0       ; [SYSTEM] Console device entry points
                                MAXSEC:
00000036 8000                       dw      128     ; [SYSTEM] Maximum allowed sector size
                                BUFFHEAD:
00000038 00000000                   dd      0       ; [SYSTEM] Pointer to head of buffer queue
                                CDSADDR:
0000003C 00000000                   dd      0       ; [SYSTEM] Pointer to curdir structure table
                                SFTFCB:
00000040 00000000                   dd      0       ; [SYSTEM] pointer to FCB cache table
                                KEEPCOUNT:
00000044 0000                       dw      0       ; [SYSTEM] count of FCB opens to keep
                                NUMIO:
00000046 00                         db      0       ; [SYSTEM] Number of disk tables
                                CDSCOUNT:
00000047 00                         db      0       ; [SYSTEM] Number of CDS structures in above

                                ; A fake header for the NUL device
                                NULDEV:
00000048 00000000                   dd      0       ; [SYSTEM] Link to rest of device list
                                    ;dw     8004h
0000004C 0480                       dw      DEVTYP|ISNULL ; [SYSTEM] Null device attributes
0000004E [CD0D]                     dw      SNULDEV ; [SYSTEM] Strategy entry point
00000050 [D30D]                     dw      INULDEV ; [SYSTEM] Interrupt entry point
00000052 4E554C2020202020          db      "NUL     " ; [SYSTEM] Name of null device
                                SPLICES:
0000005A 00                         db      0       ; [SYSTEM] TRUE => splices being done

                                Special_Entries:
0000005B 0000                       dw      0       ; [SYSTEM] address of special entries ;AN000;
                                UU_IFS_DOS_CALL:
0000005D 00000000                   dd      0       ; [SYSTEM] entry for IFS DOS service ;AN000;
                                ;
                                ; UU_IFS_HEADER:
                                ;   dd      0       ; [SYSTEM] IFS header chain ;AN000;

                                ChkCopyProt:
00000061 0000                       dw      0       ; M068
                                A20OFF_PSP:
00000063 0000                       dw      0       ; M068
                                BUFFERS_PARM1:
00000065 0000                       dw      0       ; [SYSTEM] value of BUFFERS= ,m      ;AN000;
                                BUFFERS_PARM2:
00000067 0000                       dw      0       ; [SYSTEM] value of BUFFERS= ,n ;AN000;
                                BOOTDRIVE:
00000069 00                         db      0       ; [SYSTEM] the boot drive ;AN000;
                                DDMOVE:
0000006A 00                         db      0       ; [SYSTEM] 1 if we need DWORD move ;AN000;
                                EXT_MEM_SIZE:
0000006B 0000                       dw      0       ; [SYSTEM] extended memory size      ;AN000;

                                HASHINITVAR: ; LABEL   WORD       ; AN000;
                                ;
                                ; Replaced by next two declarations
                                ;
                                ;UU_BUF_HASH_PTR:
                                ;   dd      0       ; [SYSTEM] buffer Hash table addr
                                ;UU_BUF_HASH_COUNT:
                                ;   dw      1       ; [SYSTEM] number of Hash entries

                                BufferQueue:
0000006D 00000000                   dd      0       ; [SYSTEM] Head of the buffer Queue
                                DirtyBufferCount:
00000071 0000                       dw      0       ; [SYSTEM] Count of Dirty buffers in the Que
                                                    ; BUGBUG ---- change to byte
                                SC_CACHE_PTR:
00000073 00000000                   dd      0       ; [SYSTEM] secondary cache pointer
                                SC_CACHE_COUNT:
00000077 0000                       dw      0       ; [SYSTEM] secondary cache count
                                BuffInHMA:
00000079 00                         db      0       ; Flag to indicate that buffs are in HMA
                                LoMemBuff:
0000007A 00000000                   dd      0       ; Ptr to intermediate buffer
                                                    ;  in Low mem when buffs are in HMA
                                ;
                                ; All variables which have UU_ as prefix can be reused for other
                                ; purposes and can be renamed. All these variables were used for
                                ; EMS support of Buffer Manager. Now they are useless for Buffer
```

```
                                ; manager ---- MOHANS
                                ;
                                    ;I_am  UU_BUF_EMS_FIRST_PAGE,3,<0,0,0>
                                UU_BUF_EMS_FIRST_PAGE:
0000007E 000000                     db     0,0,0  ; holds the first page above 640K
                                
                                    ;;I_am UU_BUF_EMS_NPA640,WORD,<0> ; holds the number of pages
                                ;UU_BUF_EMS_NPA640:                        ; above 640K
                                ;   dw     0
                                
                                CLOFATENTRY:
00000081 FFFF                       dw     -1     ; M014: Holds the data that
                                                   ; is used in pack/unpack rts.
                                                   ; in fat.asm if cluster 0 is specified.
                                                   ; SR;
                                IoStatFail:
00000083 00                         db     0      ; IoStatFail has been added to
                                                   ; record a fail on an I24
                                                   ; issued from IOFUNC on a status call.
                                
                                ;***I_am    UU_BUF_EMS_MODE,BYTE,<-1>     ; EMS mode     ;AN000;
                                ;***I_am    UU_BUF_EMS_HANDLE,BYTE        ; buffer EMS handle ;AN000;
                                ;***I_am    UU_BUF_EMS_PAGE_FRAME,WORD ,<-1>; EMS page frame # ;AN000;
                                ;***I_am    UU_BUF_EMS_SEG_CNT,WORD,<1>   ; EMS seg count       ;AN000;
                                ;***I_am    UU_BUF_EMS_PFRAME,WORD        ; EMS page frame seg address ;AN000;
                                ;***I_am    UU_BUF_EMS_RESERV,WORD,<0>    ; reserved     ;AN000;
                                ;
                                ;***I_am    UU_BUF_EMS_MAP_BUFF,1,<0>     ; this is not used to save the
                                                                           ; state of the        buffers page.
                                                                           ; This one byte is retained to
                                                                           ; keep the size of this data
                                                                           ; block the same.;
                                ALLOCMSAVE:
00000084 00                         db     0      ; M063: temp var. used to
                                                   ; M063: save alloc method in
                                                   ; M063: msproc.asm
                                A20OFF_COUNT:
00000085 00                         db     0      ; M068: indiactes the # of
                                                   ; M068: int 21 calls for
                                                   ; M068: which A20 is off
                                DOS_FLAG:
00000086 00                         db     0      ; see DOSSYM.INC for Bit
                                                   ; definitions
                                UNPACK_OFFSET:
00000087 0000                       dw     0      ; saves pointer to the start
                                                   ; of unpack code in exepatch.
                                                   ; asm.
                                UMBFLAG:
00000089 00                         db     0      ; M003: bit 0 indicates the
                                                   ; M003: link state of the UMBs
                                                   ; M003: whether linked or not
                                                   ; M003: to the DOS arena chain
                                SAVE_AX:
0000008A 0000                       dw     0      ; M000: temp varibale to store ax
                                                   ; M000: in msproc.asm
                                UMB_HEAD:
0000008C FFFF                       dw     -1     ; M000: this is initialized to
                                                   ; M000: the first umb arena by
                                                   ; M000: BIOS sysinit.
                                START_ARENA:
0000008E 0100                       dw     1      ; M000: this is the first arena
                                                   ; M000: from which DOS will
                                                   ; M000: start its scan for alloc.
                                
                                ; End of SYSINITVar block
                                ;--------------------------------------------------------------------------
                                
                                ; 25/04/2019 - Retro DOS v4.0
                                
                                ; 16/07/2018
                                ; MSDOS 3.3 (& MDOS 6.0)
                                
                                ;
                                ; Sharer jump table
                                ;
                                
                                ;PUBLIC     JShare
                                    ;EVEN
                                
                                ;JShare     LABEL   DWORD
                                ;   DW      OFFSET DOSCODE:BadCall, 0
                                ;   DW      OFFSET DOSCODE:OKCall,  0  ; 1   MFT_enter
                                ;   DW      OFFSET DOSCODE:OKCall,  0  ; 2   MFTClose
                                ;   DW      OFFSET DOSCODE:BadCall, 0  ; 3   MFTCluU
                                ;   DW      OFFSET DOSCODE:BadCall, 0  ; 4   MFTCloseP
                                ;   DW      OFFSET DOSCODE:BadCall, 0  ; 5   MFTCloN
                                ;   DW      OFFSET DOSCODE:BadCall, 0  ; 6   set_block
                                ;   DW      OFFSET DOSCODE:BadCall, 0  ; 7   clr_block
                                ;   DW      OFFSET DOSCODE:OKCall,  0  ; 8   chk_block
                                ;   DW      OFFSET DOSCODE:BadCall, 0  ; 9   MFT_get
                                ;   DW      OFFSET DOSCODE:BadCall, 0  ; 10  ShSave
                                ;   DW      OFFSET DOSCODE:BadCall, 0  ; 11  ShChk
                                ;   DW      OFFSET DOSCODE:OKCall , 0  ; 12  ShCol
                                ;   DW      OFFSET DOSCODE:BadCall, 0  ; 13  ShCloseFile
                                ;   DW      OFFSET DOSCODE:BadCall, 0  ; 14  ShSU
                                
                                align 2
                                
                                JShare:
00000090 [3407]0000                 DW      BadCall,0
00000094 [3807]0000             MFT_enter:  DW      OKCall, 0  ; 1   MFT_enter
00000098 [3807]0000             MFTClose:   DW      OKCall, 0  ; 2   MFTClose
0000009C [3407]0000             MFTCluU:            DW      BadCall,0  ; 3   MFTCluU
000000A0 [3407]0000             MFTCloseP:  DW      BadCall,0  ; 4   MFTCloseP
000000A4 [3407]0000             MFTCloN:    DW      BadCall,0  ; 5   MFTCloN
000000A8 [3407]0000             set_block:  DW      BadCall,0  ; 6   set_block
000000AC [3407]0000             clr_block:  DW      BadCall,0  ; 7   clr_block
000000B0 [3807]0000             chk_block:  DW      OKCall, 0  ; 8   chk_block
000000B4 [3407]0000             MFT_get:    DW      BadCall,0  ; 9   MFT_get
000000B8 [3407]0000             ShSave:             DW      BadCall,0  ; 10  ShSave
000000BC [3407]0000             ShChk:              DW      BadCall,0  ; 11  ShChk
000000C0 [3807]0000             ShCol:              DW      OKCall, 0  ; 12  ShCol
000000C4 [3407]0000             ShCloseFile:        DW      BadCall,0  ; 13  ShCloseFile
000000C8 [3407]0000             ShSU:               DW      BadCall,0  ; 14  ShSU
                                
                                
                                ;=============================================================================
                                ; CONST2.ASM (MSDOS 6.0, 1991)
                                ;=============================================================================
                                ; 25/04/2019 - Retro DOS 4.0
                                ; 16/07/2018 - Retro DOS 3.0
                                
                                ;Break <Initialized data and data used at DOS initialization>
                                ;--------------------------------------------------------------------------
```

```
                                ; We need to identify the parts of the data area that are relevant to tasks
                                ; and those that are relevant to the system as a whole.  Under 3.0, the system
                                ; data will be gathered with the system code.  The process data under 2.x will
                                ; be available for swapping and under 3.0 it will be allocated per-process.
                                ;
                                ; The data that is system data will be identified by [SYSTEM] in the comments
                                ; describing that data item.

                                ;DOSDATA SEGMENT WORD PUBLIC 'DATA'

                                ;
                                ; Table of routines for assignable devices
                                ;
                                ; MSDOS allows assignment if the following standard devices:
                                ;   stdin  (usually CON input)
                                ;   stdout (usually CON output)
                                ;   auxin  (usually AUX input)
                                ;   auxout (usually AUX output)
                                ;   stdlpt (usually PRN output)
                                ;
                                ; SPECIAL NOTE:
                                ;   Status of a file is a strange idea. We choose to handle it in this manner:
                                ;   If we're not at end-of-file, then we always say that we have a character.
                                ;   Otherwise, we return ^Z as the character and set the ZERO flag. In this
                                ;   manner we can support program written under the old DOS (they use ^Z as EOF
                                ;   on devices) and programs written under the new DOS (they use the ZERO flag
                                ;   as EOF).

                                ; Default SFTs for boot up

                                            ;PUBLIC SFTABL

                                SFTABL:       ; LABEL   DWORD           ; [SYSTEM] file table
000000CC FFFF                               DW -1                   ; [SYSTEM] link to next table
000000CE FFFF                               DW -1                   ; [SYSTEM] link seg to next table
                                            ;dw 5 ; 24/03/2024
000000D0 0500                               DW sf_default_number  ; [SYSTEM] Number of entries in table
                                            ;times 295 db 0 ; MSDOS 6.0 ; PCDOS 7.1  ; 24/03/2024
000000D2 00<rep 127h>                       times (sf_default_number*sf_entry_size) db 0

                                ; the next two variables relate to the position of the logical stdout/stdin
                                ; cursor. They are only meaningful when stdin/stdout are assigned to the
                                ; console.
                                            ; DOSDATA:01F9h (MSDOS 6.21) ; PCDOS 7.1 ; 24/03/2024
000001F9 00                     CARPOS:             db 0                ; [SYSTEM] cursor position in stdin
000001FA 00                     STARTPOS:   db 0                  ; [SYSTEM] position of cursor at beginning
                                                                  ;          of buffered input call
000001FB 00<rep 80h>            INBUF:              times 128 db 0        ; [SYSTEM] general device input buffer
0000027B 00<rep 83h>            CONBUF:             times 131 db 0        ; [SYSTEM] The rest of INBUF and console buffer
                                            ; DOSDATA:02FEh (MSDOS 6.21) ; PCDOS 7.1
000002FE 00                     PFLAG:              db 0                  ; [SYSTEM] printer echoing flag
000002FF 00                     VERFLG:             db 0                  ; [SYSTEM] Initialize with verify off
00000300 03                     CHARCO:             db 00000011b ; 3      ; [SYSTEM] Allows statchks every 4 chars...
                                switch_character:
00000301 2F                     chSwitch:   db '/'                ; UNUSED - obsolete datum, can be reused
00000302 00                     AllocMethod:        db 0                  ; [SYSTEM] how to alloc first(best)last
00000303 00                     fShare:             db 0                  ; [SYSTEM] TRUE => sharing installed
00000304 01                     DIFFNAM:    db 1                  ; [SYSTEM] Indicates when MYNAME has changed
00000305 20<rep 10h>            MYNAME:             times 16 db 20h             ; [SYSTEM] My network name

                                ; The following table is a list of addresses that the sharer patches to be
                                ; PUSH AX to enable the critical sections

                                            ; DOSDATA:0315h (MSDOS 6.21) ; PCDOS 7.1 ; 24/03/2024

                                ;PUBLIC     CritPatch

                                CritPatch:  ; LABEL WORD

                                ;IRP sect,<critDisk,critDevice>

                                ;IF (NOT REDIRECTOR) AND (NOT SHAREF)
                                ;
                                ;SR; Change code patch address to a variable in data segment
                                ;
                                ;       dw OFFSET DOSDATA: redir_patch
                                ;       dw OFFSET DOSDATA: redir_patch
                                ;
                                ;;hkn       Short_Addr  E&sect
                                ;;hkn       Short_Addr  L&sect
                                ;
                                ;ELSE
                                ;   DW      0
                                ;   DW      0
                                ;ENDIF
                                ;ENDM
                                ;   DW      0

                                    ; 25/07/2019 - Retro DOS v4.0 (MSDOS 6.21)

00000315 [650D]                     dw      redir_patch
00000317 [650D]                     dw      redir_patch
00000319 [650D]                     dw      redir_patch
0000031B [650D]                     dw      redir_patch

0000031D 0000                       dw      0

                                ; WARNING!!! PRINT and PSPRINT *REQUIRE* ErrorMode to precede INDOS.
                                ; Also, IBM server 1.0 requires this also.

                                    ;db     90h ; 24/03/2024

                                    ;EVEN                   ; Force swap area to start on word boundry

0000031F 90                     align 2
                                    ;PUBLIC SWAP_START

                                ; 10/03/2024 - Retro DOS v5.0
                                ; PCDOS 7.1 IBMDOS.COM - DOSDATA:0320h
                                SWAP_START: ; LABEL BYTE
00000320 00                     ERRORMODE:  db 0            ; Flag for INT 24 processing
00000321 00                     INDOS:              db 0            ; DOS status for interrupt processing
00000322 FF                     WPERR:              db -1           ; Write protect error flag
00000323 00                     EXTERR_LOCUS:       db 0            ; Extended Error Locus
00000324 0000                   EXTERR:             dw 0            ; Extended Error code

                                ;WARNING Following two bytes Accessed as word in $GetExtendedError
00000326 00                     EXTERR_ACTION:      db 0            ; Extended Error Action
00000327 00                     EXTERR_CLASS:       db 0            ; Extended Error Class
                                ; end warning

00000328 00000000               EXTERRPT:   dd 0            ; Extended Error pointer
```

```
52687 0000032C 80000000                DMAADD:             dd 80h          ; User's disk transfer address (disp/seg)
52688 00000330 0000                    CurrentPDB: dw 0            ; Current process identifier
52689 00000332 0000                    ConC_Spsave:        dw 0            ; saved SP before ^C
52690 00000334 0000                    exit_code:  dw 0            ; exit code of last proc.
52691 00000336 00                      CURDRV:             db 0            ; Default drive (init A)
52692 00000337 00                      CNTCFLAG:   db 0            ; ^C check in dispatch disabled
52693                                  ;                           ; F.C. 2/17/86
52694 00000338 00                      CPSWFLAG:   db 0            ; Code Page Switching Flag  DOS 4.00
52695 00000339 00                      CPSWSAVE:   db 0            ; copy of above in case of ABORT
52696                                  ;align 2
52697                                  ; 10/03/2024 - Retro DOS v5.0
52698                                  ; PCDOS 7.1 IBMDOS.COM - DOSDATA:033Ah
52699                                  SWAP_ALWAYS:        ; 05/08/2018
52700 0000033A 0000                    USER_IN_AX: dw 0            ; User INPUT AX value (used for
52701                                                              ;   extended error type stuff.
52702                                                              ;   NOTE: does not have Correct value on
52703                                                              ;   1-12, OEM, Get/Set CurrentPDB,
52704                                                              ;   GetExtendedError system calls)
52705 0000033C 0000                    PROC_ID:    dw 0            ; PID for sharing (0 = local)
52706 0000033E 0000                    USER_ID:    dw 0            ; Machine for sharing (0 = local)
52707 00000340 0000                    FirstArena: dw 0            ; first free block found
52708 00000342 0000                    BestArena:  dw 0            ; best free block found
52709 00000344 0000                    LastArena:  dw 0            ; last free block found
52710 00000346 0000                    ENDMEM:             dw 0            ; End of memory used in DOSINIT
52711 00000348 0000                    LASTENT:    dw 0            ; Last entry for directory search
52712 0000034A 00                      FAILERR:    db 0            ; NZ if user did FAIL on I 24
52713 0000034B 00                      ALLOWED:    db 0            ; Allowed I 24 answers (see allowed_)
52714 0000034C 00                      NoSetDir:   db 0            ; true -> do not set directory
52715 0000034D 00                      DidCTRLC:   db 0            ; true -> we did a ^C exit
52716 0000034E 00                      SpaceFlag:  db 0            ; true -> embedded spaces are allowed in FC
52717
52718                                  ; Warning!  The following items are accessed as a WORD in TIME.ASM
52719                                      ;EVEN
52720 0000034F 90                      align 2
52721                                              ; DOSDATA:0350h (MSDOS 6.21)
52722 00000350 00                      DAY:                db 0            ; Day of month
52723 00000351 00                      MONTH:              db 0            ; Month of year
52724 00000352 0000                    YEAR:               dw 0            ; Year (with century)
52725 00000354 FFFF                    DAYCNT:             dw -1           ; Day count from beginning of year
52726 00000356 00                      WEEKDAY:    db 0            ; Day of week
52727                                  ; end warning
52728
52729 00000357 00                      CONSWAP:    db 0            ; TRUE => console was swapped during device read
52730 00000358 01                      IDLEINT:    db 1            ; TRUE => idle int is allowed
52731 00000359 00                      fAborting:  db 0            ; TRUE => abort in progress
52732
52733                                  ; Combination of all device call parameters
52734                                      ;PUBLIC DEVCALL         ;
52735                                  ;DEVCALL SRHEAD    <>              ; basic header for disk packet
52736                                  DEVCALL: ; 08/08/2018
52737 0000035A 00                      DEVCALL_REQLEN:  db 0               ;Length in bytes of request block
52738 0000035B 00                      DEVCALL_REQUNIT: db 0               ;Device unit number
52739 0000035C 00                      DEVCALL_REQFUNC: db 0               ;Type of request
52740 0000035D 0000                    DEVCALL_REQSTAT: dw 0               ;Status Word
52741 0000035F 00<rep 8h>                       times 8 db 0               ;Reserved for queue links
52742
52743                                      ;PUBLIC CALLUNIT
52744                                  CALLUNIT: ; LABEL   BYTE   ; unit number for disk
52745                                  CALLFLSH: ; LABEL   WORD   ;
52746 00000367 00                      CALLMED:    db 0            ; media byte
52747                                  CALLBR:       ; LABEL   DWORD     ;
52748                                      ;PUBLIC CALLXAD         ;
52749                                  CALLXAD:  ; LABEL   DWORD ;
52750 00000368 00                      CALLRBYT:   db 0            ;
52751                                      ;PUBLIC CALLVIDM        ;
52752                                  CALLVIDM: ; LABEL   DWORD ;
52753 00000369 00<rep 3h>                  times 3 db 0   ;
52754                                      ;PUBLIC CallBPB                ;
52755                                  CALLBPB:  ; LABEL   DWORD ;
52756                                  CALLSCNT:                   ;
52757 0000036C 0000                            dw 0                ;
52758                                      ;PUBLIC CALLSSEC        ;
52759                                  CALLSSEC: ; LABEL   WORD  ;
52760 0000036E 0000                            dw 0                ;
52761 00000370 00000000                CALLVIDRW:  dd 0            ;
52762                                  ;MSDOS 6.0
52763 00000374 00000000                CALLNEWSC:  dd 0            ; starting sector for >32mb
52764 00000378 00000000                CALLDEVAD:  dd 0            ; stash for device entry point
52765
52766                                  ; Same as above for I/O calls     ;
52767                                                              ;
52768                                      ;PUBLIC IOCall          ;
52769                                  ;IOCALL     SRHEAD  <>              ;
52770                                  IOCALL:     ; 07/08/2018
52771 0000037C 00                      IOCALL_REQLEN:     db 0             ;Length in bytes of request block
52772 0000037D 00                      IOCALL_REQUNIT:    db 0             ;Device unit number
52773 0000037E 00                      IOCALL_REQFUNC: db 0                ;Type of request
52774 0000037F 0000                    IOCALL_REQSTAT: dw 0                ;Status Word
52775 00000381 00<rep 8h>                  times 8 db 0             ;Reserved for queue links
52776                                  IOFLSH:       ; LABEL   WORD      ;
52777                                         ;PUBLIC  IORCHR          ;
52778                                  IORCHR:       ; LABEL   BYTE      ;
52779 00000389 00                      IOMED:              db 0           ;
52780 0000038A 00000000                IOXAD:              dd 0           ;
52781 0000038E 0000                    IOSCNT:             dw 0           ;
52782 00000390 0000                    IOSSEC:             dw 0           ;
52783
52784                                  ; Call struct for DSKSTATCHK      ;
52785 00000392 0E                      DSKSTCALL:  db DRDNDHL     ; = 14
52786 00000393 00                                  db 0
52787 00000394 05                      DSKSTCOM:   db DEVRDND     ; = 5
52788 00000395 0000                    DSKSTST:    dw 0            ;
52789 00000397 00<rep 8h>                  times 8 db 0            ;
52790 0000039F 00                      DSKCHRET:   db 0            ;
52791
52792                                  ;hkn; short_addr has been changed to provide offset in DOSCODE.
52793                                  ;hkn; deviobuf is in DATA seg (DOSDATA)
52794                                  ;hkn   short_addr  DEVIOBUF       ;
52795
52796 000003A0 [BC03]                  DEVIOBUF_PTR        dw DEVIOBUF
52797 000003A2 0000                    DOSSEG_INIT dw 0            ; DOS segment set at Init
52798 000003A4 0100                    DSKSTCNT:   dw 1            ;
52799 000003A6 0000                                dw 0            ;
52800
52801 000003A8 00                      CreatePDB:  db 0            ; flag for creating a process
52802
52803                                  ;MSDOS 6.0
52804                                  Lock_Buffer:        ; LABEL  DWORD ;MS. DOS Lock Buffer for Ext Lock
52805 000003A9 00000000                            dd 0            ;MS. position
52806 000003AD 00000000                            dd 0            ;MS. length
52807
52808                                  ;hkn; the foll. was moved from dosmes.asm.
52809
52810                                  ; 29/01/2024 - Retro DOS v5.0 (PCDOS v7.1 IBMDOS.COM)
```

```
                                ; DOSDATA:03B1h
000003B1 90                     IOCTL_drvnum:      db 90h ; nop

                                    ;EVEN
                                align 2                         ; needed to maintain offsets

                                            ; DOSDATA:03B2h (MSDOS 6.21)
                                USERNUM:
000003B2 0000                               dw 0            ; 24 bit user number
000003B4 00                                 db 0
                                ;IF IBM
                                ;IF IBMCOPYRIGHT
                                ; 24/03/2024 (PCDOS v7.1 IBMDOS.COM)
000003B5 00                     OEMNUM:             DB 0            ; 8 bit OEM number
                                ;ELSE
                                ;OEMNUM:    DB 0FFh         ; 8 bit OEM number
                                ;ENDIF
                                ;ELSE
                                ;OEMNUM:    DB 0FFh         ; (MSDOS 6.22) ; 24/03/2024
                                ;ENDIF

                                ;============================================================================
                                ; MS_DATA.ASM (MSDOS 6.0, 1991)
                                ;============================================================================
                                ; 25/04/2019 - Retro DOS 4.0

                                ; Retro DOS v4.0 NOTE: (by Erdogan Tan, 25/04/2019)
                                ; ---------------------------------------------------------
                                ; This data section which was named as uninitialized data
                                ; (as overlayed by initialization code) but follows
                                ; initialized data section from DOSDATA:03B6h address
                                ; (in otherwords, the method is different than MSDOS 3.3,
                                ; and there is not overlaying..)
                                ; *********************************************************
                                ; Reference: MSDOS 6.21 kernel DOSDATA section (4970 bytes)
                                ; follows DOSCODE section in the kernel file (MSDOS.SYS)
                                ; (it is located at offset 0BF70h, file offset 0BF70h-3DE0h)
                                ; but starts from offset 0 (ORG 0) and ends at offset 1370h.
                                ; TIMEBUF is at offset 03B6h.
                                ; *********************************************************

                                ;Break <Uninitialized data overlayed by initialization code>
                                ;-----------------------------------------------------------------------------
                                ;DOSDATA    SEGMENT WORD PUBLIC 'DATA'
                                ; Init code overlaps with data area below

                                ;   ORG     0

                                MSDAT001S:  ; label byte

                                ; DOSDATA:03B6h    ; MSDOS 6.21 (MSDOS.SYS, file offset 0BF70h-3DE0h+3B6h)
                                TIMEBUF: ;  times 6 db 0
000003B6 0000<rep 3h>               times 3 dw     0            ; Time read from clock device
000003BC 0000                   DEVIOBUF:   dw     0            ; Buffer for I/O under file assignment

                                ; The following areas are used as temp buffer in EXEC system call

                                ; DOSDATA:03BEh
                                OPENBUF: ;times 64  dw    0
000003BE 00<rep 80h>                times  128 db 0             ; buffer for name operations
                                RENBUF:
0000043E 00<rep 80h>                times  128 db 0             ; buffer for rename destination

                                ; Buffer for search calls
                                SEARCHBUF:
000004BE 00<rep 35h>                times  53  db 0             ; internal search buffer
                                DUMMYCDS:  ;times 88 db 0
000004F3 00<rep 58h>                times  curdirLen db 0

                                ; End of contiguous buffer

                                ; Temporary directory entry for use by many routines. Device directory
                                ; entries (bogus) are built here.

                                ; DOSDATA:054Bh

                                DEVFCB:     ; LABEL   BYTE              ; Uses NAME1, NAME2, combined

                                ; WARNING..  do not alter position of NAME1 relative to DEVFCB
                                ; without first examining BUILD_DEVICE_ENT. Look carefully at DOS_RENAME
                                ; as well as it is the only guy who uses NAME2 and DESTSTART.

                                NAME1:
0000054B 00<rep Ch>                     times      12 db  0        ; File name buffer
                                NAME2:
00000557 00<rep Dh>                 times  13 db  0             ;
                                DESTSTART:
00000564 0000                       dw     0                    ;
                                    ;DB ((SIZE DIR_ENTRY) - ($ - DEVFCB)) DUP (?)
                                    ;times  5  db  0
00000566 00<rep 5h>                 times  ((dir_entry.size)-($-DEVFCB)) db 0

                                ; End Temporary directory entry.

0000056B 00                     ATTRIB:     db     0            ; storage for file attributes
                                EXTFCB:
0000056C 00                         db     0            ; TRUE => extended FCB in use
                                SATTRIB:
0000056D 00                         db     0            ; Storage for search attributes
                                OPEN_ACCESS:
0000056E 00                         db     0            ; access of open system call
                                FOUNDDEL:
0000056F 00                         db     0            ; true => file was deleted
                                FOUND_DEV:
00000570 00                         db     0            ; true => search found a device
                                FSPLICE:
00000571 00                         db     0            ; true => do a splice in transpath
                                FSHARING:
00000572 00                         db     0            ; TRUE => no redirection
                                SECCLUSPOS:
00000573 00                         db     0            ; Position of first sector within cluster
00000574 00                     TRANS:      db     0            ;
00000575 00                     READOP:     db     0            ;
                                THISDRV:
00000576 00                         db     0            ;
                                CLUSFAC:
00000577 00                         db     0            ;
                                CLUSSPLIT:
00000578 00                         db     0            ;
                                INSMODE:
00000579 00                         db     0            ; true => insert mode in buffered read
0000057A 00                     CMETA:      db     0            ; count of meta'ed components found
0000057B 00                     VOLID:      db     0            ;
                                EXIT_TYPE:
```

```
                                    db      0               ; type of exit...
                                    ; 24/03/2024
                                    db      0   ; PCDOS 7.1 - DOSDATA:057Dh
                                    ;db     90h ; MSDOS 6.22 - DOSDATA:057Dh

                                    ;EVEN

                                align 2

                                ; DOSDATA:057Eh

                                ; WARNING - the following two items are accessed as a word

                                CREATING:
                                    db      0               ; true => creating a file
                                DELALL:     db      0               ; = 0 iff BUGBUG
                                                                    ; = DIRFREE iff BUGBUG
                                EXITHOLD:
                                    dd      0               ; Temp location for proc terminate
                                USER_SP:
                                    dw      0               ; User SP for system call
                                USER_SS:
                                    dw      0               ; User SS for system call
                                CONTSTK:
                                    dw      0               ;
                                THISDPB:
                                    dd      0               ;
                                CLUSSAVE:
                                    dw      0               ;
                                CLUSSEC:
                                    dd      0               ;>32mb                  AC0000
                                PREREAD:
                                    dw      0               ; 0 means preread; 1 means optional
                                FATBYT:     dw      0               ; Used by ALLOCATE
                                FATBYTE:
                                    dw      0               ; Used by $SLEAZEFUNC
                                ; DOSDATA:059Ah
                                DEVPT:      dd      0               ;
                                THISSFT:
                                    dd      0               ; Address of user SFT
                                THISCDS:
                                    dd      0               ; Address of current CDS
                                THISFCB:
                                    dd      0               ; Address of user FCB
                                SFN:dw      -1              ; SystemFileNumber found for accessfile
                                JFN:dw      0               ; JobFileNumber found for accessfile
                                PJFN:       dd      0               ; PointerJobFileNumber found for accessfile
                                WFP_START:
                                    dw      0               ;
                                REN_WFP:
                                    dw      0               ;
                                CURR_DIR_END:
                                    dw      0               ;
                                NEXTADD:
                                    dw      0               ;
                                LASTPOS:
                                    dw      0               ;
                                CLUSNUM:
                                    dw      0               ;
                                DIRSEC:     dd      0               ;>32mb                  AC0000
                                DIRSTART:
                                    dw      0               ;
                                SECPOS:     dd      0               ;>32mb Position of first sector accessed
                                VALSEC:     dd      0               ;>32mb Number of valid (previously written)
                                                                   ; sectors
                                BYTSECPOS:
                                    dw      0               ; Position of first byte within sector
                                BYTPOS: ;times      4 db 0          ; Byte position in file of access
                                        times       2 dw 0
                                BYTCNT1:
                                    dw      0               ; No. of bytes in first sector
                                BYTCNT2:
                                    dw      0               ; No. of bytes in last sector
                                SECCNT:     dw      0               ; No. of whole sectors
                                ; DOSDATA:05D8h
                                ENTFREE:
                                    dw      0               ;
                                ENTLAST:
                                    dw      0               ;
                                NXTCLUSNUM:
                                    dw      0               ;
                                GROWCNT:
                                    dd      0               ;
                                CURBUF:     dd      0               ;
                                CONSFT:     dd      0               ; SFT of console swapped guy.
                                SAVEBX:     dw      0               ;
                                SAVEDS:     dw      0               ;

                                RESTORE_TMP:
                                    dw      0               ; return address for restore world
                                NSS:dw      0
                                NSP:dw      0
                                ; DOSDATA:05F4h
                                EXTOPEN_FLAG:
                                    dw      0               ;FT. extended open input flag ;AN000;
                                EXTOPEN_ON:
                                    db      0               ;FT. extended open conditional flag ;AN000;
                                EXTOPEN_IO_MODE:
                                    dw      0               ;FT. extended open io mode    ;AN000;
                                SAVE_DI:
                                    dw      0               ;FT. extended open saved DI   ;AN000;
                                SAVE_ES:
                                    dw      0               ;FT. extended open saved ES   ;AN000;
                                SAVE_DX:
                                    dw      0               ;FT. extended open saved DX   ;AN000;
                                SAVE_CX:
                                    dw      0               ;FT. extended open saved CX   ;AN000;
                                SAVE_BX:
                                    dw      0               ;FT. extended open saved BX   ;AN000;
                                SAVE_SI:
                                    dw      0               ;FT. extended open saved SI   ;AN000;
                                SAVE_DS:
                                    dw      0               ;FT. extended open saved DS   ;AN000;

                                ; DOSDATA:0607h

                                ; HIGH_SECTOR is a hack to allow passing 32-bit sector numbers where
                                ; we used to just pass 16 bits in a register. Now High_SECTOR holds
                                ; the high 16, the low 16 are still in the register.

                                HIGH_SECTOR:
                                    dw      0               ;>32mb higher sector #        ;AN000;
                                    ; 25/09/2023
                                OffsetMagicPatch:
                                    ; 24/03/2024 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMDOS.COM)
```

```
53059 00000609 [4013]                     dw      MagicPatch      ;scottq 8/6/92
53060                                     ; 06/12/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
53061                                     ;dw     0
53062                                                             ;see dos\mpatch.asm
53063                                   DISK_FULL:
53064 0000060B 00                         db      0               ;>32mb indicating disk full when 1 ;AN000;
53065                                   TEMP_VAR:
53066 0000060C 0000                       dw      0               ; temporary variable for everyone ;AN000;
53067                                   TEMP_VAR2:
53068 0000060E 0000                       dw      0               ; temporary variable 2 for everyone ;AN000;
53069 00000610 00                     DrvErr:     db      0               ; used to save drive error    ;AN000;
53070                                   DOS34_FLAG:
53071 00000611 0000                       dw      0               ; common flag for DOS 3.4     ;AN000;
53072                                   NO_FILTER_PATH:
53073 00000613 00000000                   dd      0               ; pointer to original path    ;AN000;
53074                                   NO_FILTER_DPATH:
53075 00000617 00000000                   dd      0               ; pointer to original path of destination ;AN000;
53076                                   ; M008
53077                                   AbsRdWr_SS:
53078 0000061B 0000                       dw      0               ; INT 25/26 user stack segment
53079                                   AbsRdWr_SP:
53080 0000061D 0000                       dw      0               ; INT 25/26 user stack offset
53081
53082                                     ; I_am  UU_Callback_flag,BYTE,<0>  ; Unused
53083                                   ; M008
53084                                     ; 24/03/2024
53085                                     ; MSDOS 5.0 MSDOS.SYS - DOSDATA:061Fh
53086                                     ; MSDOS 6.22 MSDOS.SYS - DOSDATA:061Fh
53087                                     ; 01/01/2024
53088                                     ; PCDOS 7.1 IBMDOS.COM - DOSDATA:061Fh
53089 0000061F 00                         db      0
53090
53091                                   ; make those pushes fast!!!
53092                                   ;EVEN
53093
53094                                   align 2
53095
53096                                   StackSize   equ 180h  ; 384       ; gross but effective
53097
53098                                   ;StackSize  equ 300h  ;   768     ; This is a "trial" change IBM hasn't
53099                                   ;                             ; made up their minds about
53100
53101                                   ; WARNING!!!! DskStack may grow into AUXSTACK due to interrupt service.
53102                                   ; This is NO problem as long as AUXSTACK comes immediately before DSKSTACK
53103
53104                                   RENAMEDMA:  ; LABEL   BYTE ; See DOS_RENAME
53105
53106 00000620 00<rep 180h>                       times      StackSize db   0       ; db 384 dup(0) ;  PCDOS 7.1
53107                                   AUXSTACK:                   ; LABEL   BYTE
53108
53109 000007A0 00<rep 180h>                       times      StackSize db   0       ; db 384 dup(0) ;  PCDOS 7.1
53110                                   DSKSTACK:                   ; LABEL   BYTE
53111                                       ; 24/03/2024
53112                                       ; 01/01/2024 - Retro DOS v5.0 (PCDOS 7.1 IBMDOS.COM)
53113                                       ; (PCDOS 7.1 IBMDOS.COM - DOSDATA:0920h)
53114 00000920 402349424D3A31322E-        db '@#IBM:12.01.2003.build_1.32#@ IBMDOS.COM(USA)',0
53114 00000929 30312E323030332E62-
53114 00000932 75696C645F312E3332-
53114 0000093B 23402049424D444F53-
53114 00000944 2E434F4D2855534129-
53114 0000094D 00
53115 0000094E 00<rep 152h>               times  StackSize-($-DSKSTACK) db 0  ; db 338 dup(0) ; PCDOS 7.1
53116                                     ;times  StackSize db   0       ;
53117                                   IOSTACK:                    ; LABEL   BYTE
53118
53119                                   ; DOSDATA:0AA0h
53120
53121                                   ; patch space for Boca folks.
53122                                   ; Say What????!!! This does NOT go into the swappable area!
53123                                   ; NOTE: We include the decl of ibmpatch in ms-dos even though it is not needed.
53124                                   ;       This allows the REDIRector to work on either IBM or MS-DOS.
53125
53126                                   IBMPATCH: ; label byte
53127                                   PRINTER_FLAG:
53128 00000AA0 00                         db      0               ; [SYSTEM] status of PRINT utility
53129                                   VOLCHNG_FLAG:
53130                                     ;db     0               ; [SYSTEM] true if volume label created
53131                                     ; 24/03/2024 (PCDOS 7.1 IBMDOS.COM)
53132 00000AA1 FF                       db      0FFh  ; -1
53133                                   VIRTUAL_OPEN:
53134 00000AA2 00                         db      0               ; [SYSTEM] non-zero if we opened a virtual file
53135
53136                                   ; 24/03/2024 - Retro DOS v5.0
53137                                   ; PCDOS 7.1 IBMDOS.COM
53138                                   %if 0
53139
53140                                   ; Following 4 variables moved to MSDATA.asm from MSTABLE.asm (P4986)
53141
53142                                       ; DOSDATA:0AA3h ; MSDOS 6.22 ; MSDOS 5.0
53143                                   FSeek_drive:
53144                                       db      0               ;AN000; fastseek drive #
53145                                   FSeek_firclus:
53146                                       dw      0               ;AN000; fastseek first cluster #
53147                                   FSeek_logclus:
53148                                       dw      0               ;AN000; fastseek logical cluster #
53149                                   FSeek_logsave:
53150                                       dw      0               ;AN000; fastseek returned log clus #
53151
53152                                   %endif
53153
53154                                   ; DOSDATA:0AAAh ; MSDOS 6.22 ; MSDOS 5.0
53155                                   ; DOSDATA:0AA3h    ; PCDOS 7.1 ; 24/03/2024
53156
53157                                   TEMP_DOSLOC:
53158 00000AA3 FFFF                       dw      -1              ;stores the temporary location of dos
53159                                                             ;at SYSINIT time.
53160                                   ; 10/03/2024
53161                                   ;SWAP_END:  ; LABEL   BYTE
53162
53163                                   ; THE FOLLOWING BYTE MUST BE HERE, IMMEDIATELY FOLLOWING SWAP_END. IT CANNOT
53164                                   ; BE USED. If the size of the swap data area is ODD, it will be rounded up
53165                                   ; to include this byte.
53166
53167                                       ; 24/03/2024
53168                                       ; 05/01/2024 (PCDOS 7.1 IBMDOS.COM)
53169                                       ;db     0       ; DOSDATA:0AACh (MSDOS 5.0-6.22 MSDOS.SYS)
53170
53171                                   ; DOSDATA:0AADh
53172
53173                                   ;hkn;       DB      (512+80+32-(SWAP_END-ibmpatch)) DUP (?)
53174
53175                                   ; ----------------------------------------------------------------------------
53176                                   ; 05/01/2024 - Retro DOS v5.0
53177                                   ; PCDOS 7.1 IBMDOS.COM - DOSDATA:0AA5h
```

```
                                    ; 24/01/2024
                                LNE_COUNT:
00000AA5 0000                       dw      0       ; long name entry count (for file)

00000AA7 00<rep Eh>                 times 14 db 0

                                ENTLAST_PREV:
00000AB5 0000                       dw      0       ; previous ENTLAST (for long name search !?)

00000AB7 00<rep 24h>                times 36 db 0

                                absdrw_extd:
00000ADB 00                         db      0
                                DIRSTART_HW:
00000ADC 0000                       dw      0
                                CLUSNUM_HW:
00000ADE 0000                       dw      0
                                NXTCLUSNUM_HW:
00000AE0 0000                       dw      0
                                LASTPOS_HW:
00000AE2 0000                       dw      0
                                FATBYT_HW:
00000AE4 0000                       dw      0
                                DESTSTART_HW:
00000AE6 0000                       dw      0
                                CLUSTNUM_HW:
00000AE8 0000                       dw      0
                                CLUSDATA_HW:        ; cluster data (0 = release, -1 = allocate) ; 30/01/2024
00000AEA 0000                       dw      0
                                CCONTENT_HW:        ; UNPACK output
00000AEC 0000                       dw      0
                                ROOTCLUST_HW:
00000AEE 0000                       dw      0
                                CCOUNT_HW:  ; 09/02/2024 ; for ALLOCATE
00000AF0 0000                       dw      0
                                CLUSTERS_HW:
00000AF2 0000                       dw      0
00000AF4 0000                       dw      0
00000AF6 0000                       dw      0
                                CLSKIP_HW:
00000AF8 0000                       dw      0

                                ; 10/03/2024 - Retro DOS v5.0
                                ; (PCDOS 7.1 IBMDOS.COM)
                                SWAP_END:  ; LABEL   BYTE

                                ;DOSDATA    ENDS

                                ;============================================================================
                                ; DOSTAB.ASM (MSDOS 6.0, 1991)
                                ;============================================================================
                                ; 27/04/2019 - Retro DOS 4.0
                                ; 16/07/2018 - Retro DOS 3.0

                                ;DOSDATA Segment

                                ; DOSDATA:0AADh (MSDOS 6.21, MSDOS.SYS)

                                ; DOSDATA:0AFAh (PCDOS 7.1, IBMDOS.COM) ; 05/01/2024

                                ;
                                ; upper case table
                                ; ---------------------------------------------------------------------------
                                UCASE_TAB:  ; label   byte
00000AFA 8000                       dw      128
00000AFC 809A45418E418F80           db      128,154,069,065,142,065,143,128
00000B04 4545454949498E8F           db      069,069,069,073,073,073,142,143
00000B0C 9092924F994F5555           db      144,146,146,079,153,079,085,085
00000B14 59999A9B9C9D9E9F           db      089,153,154,155,156,157,158,159
00000B1C 41494F55A5A5A6A7           db      065,073,079,085,165,165,166,167
00000B24 A8A9AAABACADAEAF           db      168,169,170,171,172,173,174,175
00000B2C B0B1B2B3B4B5B6B7           db      176,177,178,179,180,181,182,183
00000B34 B8B9BABBBCBDBEBF           db      184,185,186,187,188,189,190,191
00000B3C C0C1C2C3C4C5C6C7           db      192,193,194,195,196,197,198,199
00000B44 C8C9CACBCCCDCECF           db      200,201,202,203,204,205,206,207
00000B4C D0D1D2D3D4D5D6D7           db      208,209,210,211,212,213,214,215
00000B54 D8D9DADBDCDDDEDF           db      216,217,218,219,220,221,222,223
00000B5C E0E1E2E3E4E5E6E7           db      224,225,226,227,228,229,230,231
00000B64 E8E9EAEBECEDEEEF           db      232,233,234,235,236,237,238,239
00000B6C F0F1F2F3F4F5F6F7           db      240,241,242,243,244,245,246,247
00000B74 F8F9FAFBFCFDFEFF           db      248,249,250,251,252,253,254,255
                                ;
                                ; file upper case table
                                ; ---------------------------------------------------------------------------
                                FILE_UCASE_TAB:    ; label  byte
00000B7C 8000                       dw      128
00000B7E 809A45418E418F80           db      128,154,069,065,142,065,143,128
00000B86 4545454949498E8F           db      069,069,069,073,073,073,142,143
00000B8E 9092924F994F5555           db      144,146,146,079,153,079,085,085
00000B96 59999A9B9C9D9E9F           db      089,153,154,155,156,157,158,159
00000B9E 41494F55A5A5A6A7           db      065,073,079,085,165,165,166,167
00000BA6 A8A9AAABACADAEAF           db      168,169,170,171,172,173,174,175
00000BAE B0B1B2B3B4B5B6B7           db      176,177,178,179,180,181,182,183
00000BB6 B8B9BABBBCBDBEBF           db      184,185,186,187,188,189,190,191
00000BBE C0C1C2C3C4C5C6C7           db      192,193,194,195,196,197,198,199
00000BC6 C8C9CACBCCCDCECF           db      200,201,202,203,204,205,206,207
00000BCE D0D1D2D3D4D5D6D7           db      208,209,210,211,212,213,214,215
00000BD6 D8D9DADBDCDDDEDF           db      216,217,218,219,220,221,222,223
00000BDE E0E1E2E3E4E5E6E7           db      224,225,226,227,228,229,230,231
00000BE6 E8E9EAEBECEDEEEF           db      232,233,234,235,236,237,238,239
00000BEE F0F1F2F3F4F5F6F7           db      240,241,242,243,244,245,246,247
00000BF6 F8F9FAFBFCFDFEFF           db      248,249,250,251,252,253,254,255

                                ; 24/03/2024 - Retro DOS v5.0
                                ; PCDOS 7.1 IBMDOS.COM
                                %if 0
                                    ; MSDOS 5.0-6.22 MSDOS.SYS - DOSDATA:0BB1h
                                ;
                                ; file char list
                                ; ---------------------------------------------------------------------------
                                FILE_CHAR_TAB:     ; label  byte
                                    dw      22                          ; length
                                    db      1,0,255                     ; include all
                                    db      0,0,20h                     ; exclude 0 - 20h
                                    db      2,14,'."/\[]:|<>+=;,'          ; exclude 14 special
                                    ;db     24 dup (?)                  ; reserved
                                    times   24 db 0
                                %endif

                                    ; MSDOS 5.0-6.22 MSDOS.SYS - DOSDATA:0BE1h

                                    ; 24/03/2024
                                    ; PCDOS 7.1 IBMDOS.COM - DOSDATA:0BFEh
```

```
                                ;
                                ; collate table
                                ; --------------------------------------------------------------------------
                                COLLATE_TAB:       ; label   byte
00000BFE 0001                       dw      256
00000C00 0001020304050607           db      0,1,2,3,4,5,6,7
00000C08 08090A0B0C0D0E0F           db      8,9,10,11,12,13,14,15
00000C10 10111213141516 17          db      16,17,18,19,20,21,22,23
00000C18 18191A1B1C1D1E1F           db      24,25,26,27,28,29,30,31
00000C20 2021222324252627           db      " ","!",'"',"#","$","%","&","'"
00000C28 28292A2B2C2D2E2F           db      "(",")","*","+",",","-",".","/"
00000C30 3031323334353637           db      "0","1","2","3","4","5","6","7"
00000C38 38393A3B3C3D3E3F           db      "8","9",":",";","<","=",">","?"
00000C40 4041424344454647           db      "@","A","B","C","D","E","F","G"
00000C48 48494A4B4C4D4E4F           db      "H","I","J","K","L","M","N","O"
00000C50 5051525354555657           db      "P","Q","R","S","T","U","V","W"
00000C58 58595A5B5C5D5E5F           db      "X","Y","Z","[","\","]","^","_"
00000C60 6041424344454647           db      "`","A","B","C","D","E","F","G"
00000C68 48494A4B4C4D4E4F           db      "H","I","J","K","L","M","N","O"
00000C70 5051525354555657           db      "P","Q","R","S","T","U","V","W"
00000C78 58595A7B7C7D7E7F           db      "X","Y","Z","{","|","}","~",127
00000C80 4355454141414143           db      "C","U","E","A","A","A","A","C"
00000C88 4545454949494141           db      "E","E","E","I","I","I","A","A"
00000C90 4541414F4F4F5555           db      "E","A","A","O","O","O","U","U"
00000C98 594F552424242424           db      "Y","O","U","$","$","$","$","$"
00000CA0 41494F554E4EA6A7           db      "A","I","O","U","N","N",166,167
00000CA8 3FA9AAABAC212222           db      "?",169,170,171,172,"!",'"','"'
00000CB0 B0B1B2B3B4B5B6B7           db      176,177,178,179,180,181,182,183
00000CB8 B8B9BABBBCBDBEBF           db      184,185,186,187,188,189,190,191
00000CC0 C0C1C2C3C4C5C6C7           db      192,193,194,195,196,197,198,199
00000CC8 C8C9CACBCCCDCECF           db      200,201,202,203,204,205,206,207
00000CD0 D0D1D2D3D4D5D6D7           db      208,209,210,211,212,213,214,215
00000CD8 D8D9DADBDCDDDEDF           db      216,217,218,219,220,221,222,223
00000CE0 E053                       db      224,"S"
00000CE2 E2E3E4E5E6E7               db      226,227,228,229,230,231
00000CE8 E8E9EAEBECEDEEEF           db      232,233,234,235,236,237,238,239
00000CF0 F0F1F2F3F4F5F6F7           db      240,241,242,243,244,245,246,247
00000CF8 F8F9FAFBFCFDFEFF           db      248,249,250,251,252,253,254,255
                                ;
                                ; ----------------------------------------------<MSKK01>----------------------
                                ;
                                ; MSDOS 5.0-6.22 MSDOS.SYS - DOSDATA:0CE3h
                                ; 24/03/2024
                                ; PCDOS 7.1 IBMDOS.COM - DOSDATA:0D00h
                                ;
                                ; 29/04/2019
                                ;
                                ; dbcs is not supported in DOS 3.3
                                ;            DBCS_TAB        CC_DBCS <>
                                ;
                                ; DBCS for DOS 4.00                          2/12/KK
                                ;
                                DBCS_TAB:   ; label byte          ;AN000;  2/12/KK
                                ; ----------------------------------------------<MSKK01>----------------------
                                ;ifdef      DBCS
                                ; ifdef       JAPAN
                                ;           dw      6               ; <MSKK01>
                                ;           db      081h,09fh       ; <MSKK01>
                                ;           db      0e0h,0fch       ; <MSKK01>
                                ;           db      0,0             ; <MSKK01>
                                ;
                                ;           db      0,0,0,0,0,0,0,0,0,0   ; <MSKK01>
                                ; endif
                                ; ifdef       TAIWAN
                                ;           dw      4               ; <TAIWAN>
                                ;           db      081h,0FEh       ; <TAIWAN>
                                ;           db      0,0             ; <TAIWAN>
                                ;
                                ;           db      0,0,0,0,0,0,0,0,0,0,0,0
                                ; endif
                                ; ifdef   KOREA                            ; Keyl
                                ;                dw      4                ; <KOREA>
                                ;                db      0A1h,0FEh        ; <KOREA>
                                ;                db      0,0              ; <KOREA>
                                ;
                                ;           db      0,0,0,0,0,0,0,0,0,0,0,0
                                ;  endif
                                ;else
00000D00 0000                       dw      0               ;AN000;  2/12/KK   max number
                                    ;db     16 dup(0)       ;AN000;  2/12/KK
00000D02 00<rep 10h>                times   16 db 0
                                ;
                                ;           dw      6               ;  2/12/KK
                                ;           db      081h,09Fh       ;  2/12/KK
                                ;           db      0E0h,0FCh       ;  2/12/KK
                                ;           db      0,0             ;  2/12/KK
                                ;
                                ;endif
                                ; ----------------------------------------------<MSKK01>----------------------
                                ;
                                ; 24/03/2024 - Retro DOS v5.0
                                ; PCDOS 7.1 IBMDOS.COM - DOSDATA:0D12h
                                %if 1
                                IBMDOSVERSION:
                                    ;db 7
                                    ;db 10                 ; MSVERSION
00000D12 07                         db MAJOR_VERSION
00000D13 0A                         db MINOR_VERSION
                                ;
00000D14 C8A6                   YRTAB:      db 200,166      ; Leap Year
00000D16 C8A5C8A5C8A5               db 200,165,200,165,200,165
                                ;
00000D1C 1F                     MONTAB:     db 31
                                february:
00000D1D 1C1F1E1F1E1F1F1E1F-        db 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31
00000D26 1E1F
                                %endif
                                ;
                                ; 24/03/2024 - Retro DOS v5.0
                                ; PCDOS 7.1 IBMDOS.COM - DOSDATA:0D28h
                                %if 1
                                ;
                                ; file char list
                                ; --------------------------------------------------------------------------
                                FILE_CHAR_TAB:      ; label  byte
00000D28 1600                       dw      22                          ; length
00000D2A 0100FF                     db      1,0,255                     ; include all
00000D2D 000020                     db      0,0,20h                     ; exclude 0 - 20h
00000D30 020E2E222F5C5B5D3A-        db      2,14,'."/\[]:|<>+=;,'          ; exclude 14 special
00000D39 7C3C3E2B3D3B2C
                                    ;db     24 dup (?)                  ; reserved
00000D40 00<rep 18h>                times   24 db 0
                                %endif
```

```
                                ; 24/03/2024 - Retro DOS v5.0
                                ; PCDOS 7.1 IBMDOS.COM - DOSDATA:0D58h
                                %if 1
                                SysInitTable:
00000D58 [2600]                     dw      DPBHEAD         ; SYSINITVARS
00000D5A 0000                       dw      0
00000D5C [2A12]                     dw      COUNTRY_CDPG
00000D5E 0000                       dw      0
00000D60 000000                     db      0,0,0
                                TEMPSEG:
00000D63 0000                       dw      0
                                redir_patch:
00000D65 00                         db      0
                                DosHasHMA:
00000D66 00                         db      0
                                FixExePatch:
00000D67 0000                       dw      0
                                RationalPatchPtr:
00000D69 0000                       dw      0
                                %endif

                                ; MSDOS 5.0-6.22 MSDOS.SYS - DOSDATA:0CF5h
                                ; 24/03/2024
                                ; PCDOS 7.1 IBMDOS.COM - DOSDATA:0D6Bh

                                ; --------------------------------------------------------------------------
                                ;
                                ;CASE MAPPER ROUTINE FOR 80H-FFH character range, DOS 3.3
                                ;       ENTRY: AL = Character to map
                                ;       EXIT:  AL = The converted character
                                ; Alters no registers except AL and flags.
                                ; The routine should do nothing to chars below 80H.
                                ; --------------------------------------------------------------------------
                                ; Example:

                                MAP_CASE:
                                ;Procedure MAP_CASE,FAR
00000D6B 3C80                       CMP     AL,80h
                                    ;JAE    short Map1      ;Map no chars below 80H ever
                                    ;RETF
                                    ; 24/03/2024 (PCDOS 7.1 IBMDOS.COM)
                                    ;;;
00000D6D 720C                       jb      short L_RET     ;Map no chars below 80H ever
                                    ;;;
                                Map1:
00000D6F 2C80                       SUB     AL,80h          ;Turn into index value
00000D71 1E                         PUSH    DS
00000D72 53                         PUSH    BX
00000D73 BB[FC0A]                   MOV     BX,UCASE_TAB+2
                                FINISH:
00000D76 0E                         PUSH    CS              ;Move to DS
00000D77 1F                         POP     DS
00000D78 D7                         XLAT                    ;Get upper case character
00000D79 5B                         POP     BX
00000D7A 1F                         POP     DS
                                L_RET:
00000D7B CB                         RETF

                                ;EndProc MAP_CASE

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

                                ; 24/03/2024 - Retro DOS v5.0
                                ; PCDOS 7.1 IBMDOS.COM
                                %if 0

                                ; The variables for ECS version are moved here for the same data alignments
                                ; as IBM-DOS and MS-DOS.

                                InterChar:
                                    db      0       ; Interim character flag ( 1= interim)  ;AN000;
                                ;------- NOTE: NEXT TWO BYTES SOMETIMES USED AS A WORD !! -------------------
                                DUMMY:      ; LABEL   WORD
                                InterCon:
                                    db      0       ; Console in Interim mode ( 1= interim) ;AN000;
                                SaveCurFlg:
                                    db      0       ; Print, do not advance cursor flag     ;AN000;

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

                                ; 17/01/2024 - Retro DOS v5.0
                                ;TEMPSEG:  dw      0       ;hkn; used to store ds.
                                ;redir_patch:
                                ;     db    0

                                ; DOSDATA:0D0Dh

                                Mark1:      ; label byte

                                ;IF2
                                ;   IF ((OFFSET MARK1) GT (OFFSET MSVERSION) )
                                ;           %OUT !DATA CORRUPTION!MARK1 OFFSET TOO BIG. RE-ORGANIZE DATA.
                                ;   ENDIF
                                ;ENDIF

                                      times 5 db 0

                                ;##########################################################################
                                ;
                                ; ** HACK FOR DOS 4.0 REDIR **
                                ;
                                ; The redir requires the following:
                                ;
                                ;   MSVERS  offset D12H
                                ;   YRTAB   offset D14H
                                ;   MONTAB  offset D1CH
                                ;
                                ; WARNING! WARNING!
                                ;
                                ; MARK1 SHOULD NOT BE >= 0D12H. IF SOME VARIABLE IS TO BE ADDED ABOVE DO SO
                                ; WITHOUT VIOLATING THIS AND UPDATE THE FOLL. LINE
                                ;
                                ; CURRENTLY MARK1 = 0D0DH
                                ;
                                ;##########################################################################

                                    ;ORG    0D12h

                                ; DOSDATA:0D12h (MSDOS 6.21, MSDOS.SYS)

                                    ;db     6
                                    ;db     20
```

```
                            ; Offset 0C78h in IBMDOS.COM (MSDOS 3.3, 1987)
                        MSVERSION:                      ; MS-DOS version in hex for $GET_VERSION
                        MSMAJORV: DB       MAJOR_VERSION  ; DOS_MAJOR_VERSION
                        MSMINORV: DB       MINOR_VERSION  ; DOS_MINOR_VERSION

                        ; YRTAB & MONTAB moved from TABLE segment in ms_table.asm
                        ;
                        ;   I_am    YRTAB,8,<200,166,200,165,200,165,200,165>
                        ;   I_am    MONTAB,12,<31,28,31,30,31,30,31,31,30,31,30,31>

                        ; Days in year

                        YRTAB:
                            DB      200,166             ; Leap year
                            DB      200,165
                            DB      200,165
                            DB      200,165

                        ; Days of each month

                        MONTAB:
                            DB       31                   ; January
                        february:
                            DB      28                  ; February--reset each
                                                        ; time year changes
                                DB      31                      ; March
                                DB      30                      ; April
                                DB      31                      ; May
                                DB      30                      ; June
                                DB      31                      ; July
                                DB      31                      ; August
                                DB      30                      ; September
                                DB      31                      ; October
                                DB      30                      ; November
                                DB      31                      ; December

                        ;----------------THE FOLL. BLOCK MOVED FROM TABLE SEG IN MS_TABLE.ASM-------

                        ; SYS init extended table,   DOS 3.3   F.C. 5/29/86

                        SysInitTable:
                            ;dw     SYSINITVAR
                            dw      SYSINITVARS     ; pointer to sysinit var
                                dw      0           ; segment
                                dw  COUNTRY_CDPG    ; pointer to country tabl
                                dw      0           ; segment of pointer

                            ;;;
                            ; 17/01/2024 - Retro DOS v5.0
                            ; PCDOS 7.1 IBMDOS.COM - DOSDATA:0D60h
                            ;
                            times   3 db 0
                        TEMPSEG:
                            dw      0
                        redir_patch:
                            db      0

                        ;%endif

                        ; 17/01/2024
                        ;%if 1

                        ; M021-
                        ;
                        ; DosHasHMA - This flag is set by seg_reinit when the DOS actually
                        ;   takes control of the HMA. When running, this word is a reliable
                        ;   indicator that the DOS is actually using HMA. You can't just use
                        ;   CS, because ROMDOS uses HMA with CS < F000.

                        DosHasHMA:
                            db      0
                        FixExePatch:
                            dw      0               ; M012

                        ;%endif

                        UnknownPatch:
                            dw      0
                            ;;;

                        ; 17/01/2024
                        ;%if 0

                        ; DOS 3.3 F.C. 6/12/86
                        ; FASTOPEN communications area DOS 3.3   F.C. 5/29/86

                        FastTable:                          ; a better name
                        FastOpenTable:
                            dw      2                       ; number of entries
                            dw      FastRet                     ; pointer to ret instr.
                            dw      0                       ; and will be modified by
                            dw      FastRet                     ; FASTxxx when loaded in
                            dw      0

                        ; DOS 3.3 F.C. 6/12/86

                        FastFlg:                            ; flags
                        FastOpenFlg:
                            db      0                       ; don't change the foll: order

                        ;%endif

                        ; FastOpen_Ext_Info is used as a temporary storage for saving dirpos,dirsec
                        ; and clusnum which are filled by DOS 3.nc when calling FastOpen Insert
                        ; or filled by FastOPen when calling FastOpen Lookup

                        FastOpen_Ext_Info: ; label  byte  ;dirpos
                            ;db     SIZE FASTOPEN_EXTENDED_INFO dup(0)
                            ;times  11 db 0
                            times   FEI.size db 0

                        %endif      ; 24/03/2024

                        ; Dir_Info_Buff is a dir entry buffer which is filled by FastOPen
                        ; when calling FastOpen Lookup

                        Dir_Info_Buff:      ; label  byte
                            ;db     SIZE dir_entry dup (0)
                            ;times  32 db 0
00000D7C 00<rep 20h>        times   dir_entry.size db 0

                        Next_Element_Start:
00000D9C 0000               dw      0                   ; save next element start offset
```

```
                                ; 24/03/2024
                                %if 0
                                    ; MSDOS 6.22 MSDOS.SYS - DOSDATA:0D68h

                                Del_ExtCluster:
                                    dw      0                   ; for dos_delete
                                %endif

                                ; The following is a stack and its pointer for interrupt 2F which is used
                                ; by NLSFUNC. There is no significant use of this stack, we are just trying
                                ; not to destroy the INT 21 stack saved for the user.

                                    ; MSDOS 6.22 MSDOS.SYS - DOSDATA:0D6Ah
                                    ; 24/03/2024
                                    ; PCDOS 7.1 IBMDOS.COM - DOSDATA:0D9Eh

                                USER_SP_2F: ; LABEL  WORD
00000D9E [A00D]                     dw      FAKE_STACK_2F

                                Packet_Temp:        ; label  word          ; temporary packet used by readtime
                                DOS_TEMP:   ; label  word          ; temporary word

                                FAKE_STACK_2F:
                                    ; dw    14 dup (0)              ; 12 register temporary storage
00000DA0 0000<rep Eh>               times   14 dw 0

                                ; 24/03/2024 - Retro DOS v5.0
                                ; PCDOS 7.1 IBMDOS.COM
                                %if 0
                                Hash_Temp:  ;label  word           ; temporary word
                                    ;dw     4 dup (0)              ; temporary hash table during config.sys
                                    times   4 dw 0
                                %endif

                                SCAN_FLAG:
00000DBC 00                         db      0                   ; flag to indicate key ALT_Q
                                DATE_FLAG:
00000DBD 0000                       dw      0                   ; flag to update the date

                                ; 24/03/2024
                                %if 0
                                    ; MSDOS 5.0-6.22 MSDOS.SYS - DOSDATA:0D93h

                                FETCHI_TAG: ; label  word          ; OBSOLETE - no longer used
                                    dw      0                   ; formerly part of IBM's piracy protection

                                MSG_EXTERROR:       ; label  DWORD              ; for system message addr
                                    dd      0                       ; for extended error
                                    dd      0                   ; for parser
                                    dd      0                   ; for critical errror
                                    dd      0                   ; for IFS
                                    dd      0                   ; for code reduction

                                SEQ_SECTOR: ; label  DWORD              ; last sector read
                                    dd      -1
                                SC_SECTOR_SIZE:
                                    dw      0                   ; sector size for SC
                                SC_DRIVE:
                                    db      0                   ; drive # for secondary cache

                                ; 17/01/2024
                                ;CurSC_DRIVE:
                                ;   db      -1                  ; current SC drive

                                CurSC_SECTOR:
                                    dd      0                   ; current SC starting sector
                                SC_STATUS:
                                    dw      0                   ; SC status word
                                SC_FLAG:
                                    db      0                   ; SC flag

                                %endif
                                    ; 24/03/2024
                                    ; PCDOS 7.1 IBMDOS.COM - DOSDATA:0DBFh
                                    ; (MSDOS 5.0-6.22 MSDOS.SYS - DOSDATA:0DB8h)
                                AbsDskErr:
00000DBF 0000                       dw      0                   ; Storage for Abs dsk read/write err

                                NO_NAME_ID: ; label byte
00000DC1 4E4F204E414D452020-        db      'NO NAME    '       ; null media id
00000DCA 2020

                                ;hkn; moved from TABLE segment in kstrin.asm

                                KISTR001S:  ; label byte           ; 2/17/KK
00000DCC 00                     LOOKSIZ: DB 0                      ; 0 if byte, NZ if word      2/17/KK
                                KISTR001E:  ; label byte           ; 2/17/KK

                                ; the nul device driver used to be part of the code. However, since the
                                ; header is in the data, and the entry points are only given as an offset,
                                ; the strategy and interrupt entry points must also be in the data now.

                                ; MSDOS 5.0-6.22 MSDOS.SYS - DOSDATA:0DC6h
                                ; 24/03/2024
                                ; PCDOS 7.1 IBMDOS.COM - DOSDATA:0DCDh

                                SNULDEV:
                                ;procedure snuldev,far
                                    ;or     word [es:bx+3],100h
                                    ; 17/12/2022
                                    ;or     byte [es:bx+4],01h
                                    ; 05/01/2024 - Retro DOS v4.2 (*)
                                    ; (Original MSDOS and RetroDOS DATA address compatibility) (*)
                                    ;or     byte [es:bx+SRHEAD.REQSTAT+1],(STDON>>8)
                                    ; 24/03/2024 (PCDOS 7.1 IBMDOS.COM)
00000DCD 26814F030001               or      word [es:bx+SRHEAD.REQSTAT],STDON ; set done bit
                                    ; 05/01/2024 - Retro DOS v5.0
                                    ;or     byte [es:bx+SRHEAD.REQSTAT+1],(STDON>>8)
                                INULDEV:
00000DD3 CB                         retf                            ; must not be a return!
                                ;endproc snuldev

                                ;M044
                                ; Second part of save area for saving last para of Windows memory

                                ; 17/01/2024
                                ;WinoldPatch2:
                                ;   ;db     8 dup (?)       ; M044
                                ;   times   8 db 0

                                    ; 24/03/2024 (PCDOS 7.1 IBMDOS.COM)
00000DD4 00                         db      0

                                UmbSave2:
```

```
53795                                   ;db     5 dup (?)       ; M062
53796 00000DD5 00<rep 5h>               times   5 db 0
53797                                 UmbSaveFlag:
53798 00000DDA 00                       db      0               ; M062
53799
53800                                 ; DOSDATA:0DDBh
53801
53802                                 Mark2:      ; label byte
53803
53804                                 ;IF2
53805                                 ;   IF ((OFFSET MARK2) GT (OFFSET ERR_TABLE_21) )
53806                                 ;           %OUT !DATA CORRUPTION!MARK2 OFFSET TOO BIG. RE-ORGANIZE DATA.
53807                                 ;   ENDIF
53808                                 ;ENDIF
53809
53810                                 ;###########################################################################
53811                                 ;
53812                                 ; ** HACK FOR DOS 4.0 REDIR **
53813                                 ;
53814                                 ; The redir requires the following:
53815                                 ;
53816                                 ;   ERR_TABLE_21   offset DDBH
53817                                 ;   ERR_TABLE_24   offset E5BH
53818                                 ;   ErrMap24       offset EABH
53819                                 ;
53820                                 ; WARNING! WARNING!
53821                                 ;
53822                                 ; MARK2 SHOULD NOT BE >= 0DDBH. IF SOME VARIABLE IS TO BE ADDED ABOVE DO SO
53823                                 ; WITHOUT VIOLATING THIS AND UPDATE THE FOLL. LINE
53824                                 ;
53825                                 ; CURRENTLY MARK2 = 0DD0H
53826                                 ;
53827                                 ;###########################################################################
53828
53829                                     ;ORG    0DDBh
53830
53831                                 ; DOSDATA:0DDBh (MSDOS 6.21, MSDOS.SYS)
53832                                 ; 24/03/2024
53833                                 ; PCDOS 7.1 IBMDOS.COM - DOSDATA:0DDBh
53834
53835                                 ; ---------------------------------------------------------------------------
53836                                 ;
53837                                 ; The following table defines CLASS ACTION and LOCUS info for the INT 21H
53838                                 ; errors. Each entry is 4 bytes long:
53839                                 ;
53840                                 ;       Err#,Class,Action,Locus
53841                                 ;
53842                                 ; A value of 0FFh indicates a call specific value (ie. should already
53843                                 ; be set). AN ERROR CODE NOT IN THE TABLE FALLS THROUGH TO THE CATCH ALL AT
53844                                 ; THE END, IT IS ASSUMES THAT CLASS, ACTION, LOCUS IS ALREADY SET.
53845                                 ;
53846                                 ; ---------------------------------------------------------------------------
53847
53848                                 ;ErrTab  Macro   err,class,action,locus
53849                                 ;ifidn <locus>,<0FFh>
53850                                 ;    DB  error_&err,errCLASS_&class,errACT_&action,0FFh
53851                                 ;ELSE
53852                                 ;    DB  error_&err,errCLASS_&class,errACT_&action,errLOC_&locus
53853                                 ;ENDIF
53854                                 ;ENDM
53855
53856                                 ERR_TABLE_21: ; LABEL   BYTE
53857 00000DDB 010704FF                    DB  error_invalid_function,       errCLASS_Apperr,    errACT_Abort,    0FFh
53858 00000DDF 02080302                    DB  error_file_not_found,         errCLASS_NotFnd,    errACT_User,     errLOC_Disk
53859 00000DE3 03080302                    DB  error_path_not_found,         errCLASS_NotFnd,    errACT_User,     errLOC_Disk
53860 00000DE7 04010401                    DB  error_too_many_open_files,    errCLASS_OutRes,    errACT_Abort,    errLOC_Unk
53861 00000DEB 050303FF                    DB  error_access_denied,          errCLASS_Auth,      errACT_User,     0FFh
53862 00000DEF 06070401                    DB  error_invalid_handle,         errCLASS_Apperr,    errACT_Abort,    errLOC_Unk
53863 00000DF3 07070505                    DB  error_arena_trashed,          errCLASS_Apperr,    errACT_Panic,    errLOC_Mem
53864 00000DF7 08010405                    DB  error_not_enough_memory,      errCLASS_OutRes,    errACT_Abort,    errLOC_Mem
53865 00000DFB 09070405                    DB  error_invalid_block,          errCLASS_Apperr,    errACT_Abort,    errLOC_Mem
53866 00000DFF 0A070405                    DB  error_bad_environment,        errCLASS_Apperr,    errACT_Abort,    errLOC_Mem
53867 00000E03 0B090301                    DB  error_bad_format,             errCLASS_BadFmt,    errACT_User,     errLOC_Unk
53868 00000E07 0C070401                    DB  error_invalid_access,         errCLASS_Apperr,    errACT_Abort,    errLOC_Unk
53869 00000E0B 0D090401                    DB  error_invalid_data,           errCLASS_BadFmt,    errACT_Abort,    errLOC_Unk
53870 00000E0F 0F080302                    DB  error_invalid_drive,          errCLASS_NotFnd,    errACT_User,     errLOC_Disk
53871 00000E13 10030302                    DB  error_current_directory,      errCLASS_Auth,      errACT_User,     errLOC_Disk
53872 00000E17 110D0302                    DB  error_not_same_device,        errCLASS_Unk,       errACT_User,     errLOC_Disk
53873 00000E1B 12080302                    DB  error_no_more_files,          errCLASS_NotFnd,    errACT_User,     errLOC_Disk
53874 00000E1F 500C0302                    DB  error_file_exists,            errCLASS_Already,   errACT_User,     errLOC_Disk
53875 00000E23 200A0202                    DB  error_sharing_violation,      errCLASS_Locked,    errACT_DlyRet,   errLOC_Disk
53876 00000E27 210A0202                    DB  error_lock_violation,         errCLASS_Locked,    errACT_DlyRet,   errLOC_Disk
53877 00000E2B 540104FF                    DB  error_out_of_structures,      errCLASS_OutRes,    errACT_Abort,    0FFh
53878 00000E2F 56030301                    DB  error_invalid_password,       errCLASS_Auth,      errACT_User,     errLOC_Unk
53879 00000E33 52010402                    DB  error_cannot_make,            errCLASS_OutRes,    errACT_Abort,    errLOC_Disk
53880 00000E37 32090303                    DB  error_not_supported,          errCLASS_BadFmt,    errACT_User,     errLOC_Net
53881 00000E3B 550C0303                    DB  error_already_assigned,       errCLASS_Already,   errACT_User,     errLOC_Net
53882 00000E3F 57090301                    DB  error_invalid_parameter,      errCLASS_BadFmt,    errACT_User,     errLOC_Unk
53883 00000E43 530D0401                    DB  error_FAIL_I24,               errCLASS_Unk,       errACT_Abort,    errLOC_Unk
53884 00000E47 24010405                    DB  error_sharing_buffer_exceeded,errCLASS_OutRes,    errACT_Abort,    errLOC_Mem
53885                                      ; MSDOS 6.0
53886 00000E4B 26010401                    DB  error_handle_EOF,             errCLASS_OutRes,    errACT_Abort,    errLOC_Unk ;AN000;
53887 00000E4F 27010401                    DB  error_handle_Disk_Full,       errCLASS_OutRes,    errACT_Abort,    errLOC_Unk ;AN000;
53888 00000E53 5A0D0402                    DB  error_sys_comp_not_loaded,    errCLASS_Unk,       errACT_Abort,    errLOC_Disk ;AN001;
53889 00000E57 FFFFFFFF                    DB  0FFh,                         0FFH,               0FFH,            0FFh
53890
53891                                 ; MSDOS 3.3 (IBMDOS.COM, 1987) - Offset 0D2Ah
53892                                 ;ERR_TABLE_21:     db 1,7,4,0FFh
53893                                 ;           db 2,8,3,2
53894                                 ;           db 3,8,3,2
53895                                 ;           db 4,1,4,1
53896                                 ;           db 5,3,3,0FFh
53897                                 ;           db 6,7,4,1
53898                                 ;           db 7,7,5,5
53899                                 ;           db 8,1,4,5
53900                                 ;           db 9,7,4,5
53901                                 ;           db 0Ah,7,4,5
53902                                 ;           db 0Bh,9,3,1
53903                                 ;           db 0Ch,7,4,1
53904                                 ;           db 0Dh,9,4,1
53905                                 ;           db 0Fh,8,3,2
53906                                 ;           db 10h,3,3,2
53907                                 ;           db 11h,0Dh,3,2
53908                                 ;           db 12h,8,3,2
53909                                 ;           db 50h,0Ch,3,2
53910                                 ;           db 20h,0Ah,2,2
53911                                 ;           db 21h,0Ah,2,2
53912                                 ;           db 54h,1,4,0FFh
53913                                 ;           db 56h,3,3,1
53914                                 ;           db 52h,1,4,2
53915                                 ;           db 32h,9,3,3
53916                                 ;           db 55h,0Ch,3,3
53917                                 ;           db 57h,9,3,1
53918                                 ;           db 53h,0Dh,4,1
```

```
                                ;           db 24h,1,4,5
                                ; MSDOS 6.0 (MSDOS 6.21)
                                ;           db 26h,1,4,1
                                ;           db 27h,1,4,1
                                ;           db 5Ah,0Dh,4,2
                                ; MSDOS 6.0 & MSDOS 3.3
                                ;           db 0FFh,0FFh,0FFh,0FFh

                                ; DOSDATA:0E5Bh (MSDOS 6.21, MSDOS.SYS)

                                ; --------------------------------------------------------------------------
                                ;
                                ; The following table defines CLASS ACTION and LOCUS info for the INT 24H
                                ; errors. Each entry is 4 bytes long:
                                ;
                                ;       Err#,Class,Action,Locus
                                ;
                                ; A Locus value of 0FFh indicates a call specific value (ie. should already
                                ; be set). AN ERROR CODE NOT IN THE TABLE FALLS THROUGH TO THE CATCH ALL AT
                                ; THE END.
                                ;
                                ; --------------------------------------------------------------------------

                                ERR_TABLE_24: ; LABEL   BYTE
00000E5B 130B0702                   DB  error_write_protect,         errCLASS_Media,     errACT_IntRet,   errLOC_Disk
00000E5F 14040501                   DB  error_bad_unit,              errCLASS_Intrn,     errACT_Panic,    errLOC_Unk
00000E63 150507FF                   DB  error_not_ready,             errCLASS_HrdFail,   errACT_IntRet,   0FFh
00000E67 16040501                   DB  error_bad_command,           errCLASS_Intrn,     errACT_Panic,    errLOC_Unk
00000E6B 170B0402                   DB  error_CRC,                   errCLASS_Media,     errACT_Abort,    errLOC_Disk
00000E6F 18040501                   DB  error_bad_length,            errCLASS_Intrn,     errACT_Panic,    errLOC_Unk
00000E73 19050102                   DB  error_seek,                  errCLASS_HrdFail,   errACT_Retry,    errLOC_Disk
00000E77 1A0B0702                   DB  error_not_DOS_disk,          errCLASS_Media,     errACT_IntRet,   errLOC_Disk
00000E7B 1B0B0402                   DB  error_sector_not_found,      errCLASS_Media,     errACT_Abort,    errLOC_Disk
00000E7F 1C020704                   DB  error_out_of_paper,          errCLASS_TempSit,   errACT_IntRet,   errLOC_SerDev
00000E83 1D0504FF                   DB  error_write_fault,           errCLASS_HrdFail,   errACT_Abort,    0FFh
00000E87 1E0504FF                   DB  error_read_fault,            errCLASS_HrdFail,   errACT_Abort,    0FFh
00000E8B 1F0D04FF                   DB  error_gen_failure,           errCLASS_Unk,       errACT_Abort,    0FFh
00000E8F 200A0202                   DB  error_sharing_violation,     errCLASS_Locked,    errACT_DlyRet,   errLOC_Disk
00000E93 210A0202                   DB  error_lock_violation,        errCLASS_Locked,    errACT_DlyRet,   errLOC_Disk
00000E97 220B0702                   DB  error_wrong_disk,            errCLASS_Media,     errACT_IntRet,   errLOC_Disk
00000E9B 32090303                   DB  error_not_supported,         errCLASS_BadFmt,    errACT_User,     errLOC_Net
00000E9F 23070401                   DB  error_FCB_unavailable,       errCLASS_Apperr,    errACT_Abort,    errLOC_Unk
00000EA3 24010405                   DB  error_sharing_buffer_exceeded,errCLASS_OutRes,   errACT_Abort,    errLOC_Mem
00000EA7 FF0D05FF                   DB      0FFh,                        errCLASS_Unk,       errACT_Panic,    0FFh

                                ; MSDOS 3.3 (IBMDOS.COM, 1987) - Offset 0D9Eh
                                ;ERR_TABLE_24:     db 13h,0Bh,7,2
                                ;           db 14h,4,5,1
                                ;           db 15h,5,7,0FFh
                                ;           db 16h,4,5,1
                                ;           db 17h,0Bh,4,2
                                ;           db 18h,4,5,1
                                ;           db 19h,5,1,2
                                ;           db 1Ah,0Bh,7,2
                                ;           db 1Bh,0Bh,4,2
                                ;           db 1Ch,2,7,4
                                ;           db 1Dh,5,4,0FFh
                                ;           db 1Eh,5,4,0FFh
                                ;           db 1Fh,0Dh,4,0FFh
                                ;           db 20h,0Ah,2,2
                                ;           db 21h,0Ah,2,2
                                ;           db 22h,0Bh,7,2
                                ;           db 32h,9,3,3
                                ;           db 23h,7,4,1
                                ;           db 24h,1,4,5
                                ;           db 0FFh,0Dh,5,0FFh

                                ; DOSDATA:0EABh (MSDOS 6.21, MSDOS.SYS)

                                ; --------------------------------------------------------------------------
                                ;
                                ; We need to map old int 24 errors and device driver errors into the new set
                                ; of errors. The following table is indexed by the new errors
                                ;
                                ; --------------------------------------------------------------------------

                                ;Public  ErrMap24
                                ErrMap24: ; Label   BYTE
00000EAB 13                         DB  error_write_protect      ; 0
00000EAC 14                         DB  error_bad_unit           ; 1
00000EAD 15                         DB  error_not_ready          ; 2
00000EAE 16                         DB  error_bad_command ; 3
00000EAF 17                         DB  error_CRC         ; 4
00000EB0 18                         DB  error_bad_length  ; 5
00000EB1 19                         DB  error_seek        ; 6
00000EB2 1A                         DB  error_not_DOS_disk; 7
00000EB3 1B                         DB  error_sector_not_found   ; 8
00000EB4 1C                         DB  error_out_of_paper; 9
00000EB5 1D                         DB  error_write_fault ; A
00000EB6 1E                         DB  error_read_fault  ; B
00000EB7 1F                         DB  error_gen_failure ; C
00000EB8 1F                         DB  error_gen_failure ; D  RESERVED
00000EB9 1F                         DB  error_gen_failure ; E  RESERVED
00000EBA 22                         DB  error_wrong_disk  ; F

                                ;ErrMap24: db 13h, 14h, 15h, 16h, 17h, 18h, 19h, 1Ah
                                ;       db 1Bh, 1Ch, 1Dh, 1Eh, 1Fh, 1Fh, 1Fh, 22h

                                ErrMap24End: ; LABEL   BYTE

                                ; DOSDATA:0EBBh (MSDOS 6.21, MSDOS.SYS)

                                ; 09/03/2024 - Retro DOS v5.0
                                ; PCDOS 7.1 IBMDOS.COM - DOSDATA:0EBBh (SPECIAL_VERSION)

                                ; MSDOS 5.0  MSDOS.SYS - DOSDATA:0EBBh (FIRST_BUFF_ADDR)
                                ; MSDOS 6.22 MSDOS.SYS - DOSDATA:0EBBh (FIRST_BUFF_ADDR)
                                ; Windows ME    IO.SYS - DOSDATA:0EBBh (SPECIAL_VERSION)

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

                                ; 27/04/2019 - Retro DOS v4.0

                                ; 17/01/2024
                                ;FIRST_BUFF_ADDR:
                                ;   dw      0                   ; first buffer address

                                SPECIAL_VERSION:
00000EBB 0000                       dw      0                   ;AN006; used by INT 2F 47H

                                ; 25/03/2024 - Retro DOS v5.0
                                ; PCDOS 7.1 IBMDOS.COM
                                %if 0
                                FAKE_COUNT:
```

```
                                    times 255 db 0              ;AN008; fake version count
                                %endif

                                OLD_FIRSTCLUS:
00000EBD 0000                       dw      0                   ;AN011; save old first cluster for fastopen

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

                                ;smr; moved from TABLE segment in exec.asm

00000EBF 0000                   exec_init_SP: dw 0
00000EC1 0000                   exec_init_SS: dw 0
00000EC3 0000                   exec_init_IP: dw 0
00000EC5 0000                   exec_init_CS: dw 0

                                exec_signature:
00000EC7 0000                       dw      0       ; must contain 4D5A (yay zibo!)
                                exec_len_mod_512:
00000EC9 0000                       dw      0       ; low 9 bits of length
                                exec_pages:
00000ECB 0000                       dw      0       ; number of 512b pages in file
                                exec_rle_count:
00000ECD 0000                       dw      0       ; count of reloc entries
                                exec_par_dir:
00000ECF 0000                       dw      0       ; number of paragraphs before image
                                exec_min_BSS:
00000ED1 0000                       dw      0       ; minimum number of para of BSS
                                exec_max_BSS:
00000ED3 0000                       dw      0       ; max number of para of BSS
                                exec_SS:
00000ED5 0000                       dw      0       ; stack of image
                                exec_SP:
00000ED7 0000                       dw      0       ; SP of image
                                exec_chksum:
00000ED9 0000                       dw      0       ; checksum of file (ignored)
                                exec_IP:
00000EDB 0000                       dw      0       ; IP of entry
                                exec_CS:
00000EDD 0000                       dw      0       ; CS of entry
                                exec_rle_table:
00000EDF 0000                       dw      0       ; byte offset of reloc table

                                exec_header_len    equ $-exec_signature                 ;PBUGBUG

                                ;smr; eom

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

                                ;SR;
                                ; WIN386 instance table for DOS

                                Win386_Info:
                                    ;db     3,0
                                    ; 17/01/2024 - PCDOS 7.1 IBMDOS.COM - DOSDATA:0EE1h
00000EE1 0400                       db      4,0     ; WIN386_SIS version
00000EE3 00000000                   dd      0       ; .Next_Dev_Ptr
                                Win386_Inf_Virt_Dev_Ptr:
00000EE7 00000000                   dd      0       ; .Virt_Dev_File_Ptr
00000EEB 00000000                   dd      0       ; .Reference_Data
                                Instance_Data_Ptr:
00000EEF [F70E]0000                 dw      Instance_Table, 0
                                    ; 17/01/2024 - PCDOS 7.1 IBMDOS.COM
                                    ;;
00000EF3 [3D0F]0000                 dw      Unknown_Table, 0  ; (what is this and what for ?)
                                                              ; (Win386_IIS.size)
                                    ;;

                                Instance_Table:
00000EF7 [2200]00000200             dw      CONTPOS,0,2
00000EFD [3200]00000400             dw      BCON,0,4
00000F03 [F901]00000601             dw      CARPOS,0,106h
00000F09 [0003]00000100             dw      CHARCO,0,1
00000F0F [BF0E]00002200             dw      exec_init_SP,0,34       ; M074
00000F15 [8900]00000100             dw      UMBFLAG,0,1             ; M019
00000F1B [8C00]00000200             dw      UMB_HEAD,0,2            ; M019
                                    ;;;
                                    ; 17/01/2024 - PCDOS 7.1 IBMDOS.COM
00000F21 [8600]C9000100             dw      DOS_FLAG,0C9h,1
00000F27 [B812]C9000100             dw      INDOS_FLAG,0C9h,1 ; (what for a 2nd INDOS flag, windows?)
00000F2D [B912]C9000100             dw      DEVIO_IN_PROGRESS,0C9h,1
                                                              ; "devio call in progress" status flag ptr
                                    ;;;
00000F33 00000000                   dw      0, 0

                                    ;;;
                                    ; 17/01/2024 - PCDOS 7.1 IBMDOS.COM
00000F37 FFFF                       dw      0FFFFh
00000F39 FFFF                       dw      0FFFFh
                                CLOFATENTRY_HW:
00000F3B FFFF                       dw      0FFFFh
                                Unknown_Table:
00000F3D 0000                       dw      0
00000F3F C900                       dw      0C9h
00000F41 [CC00]                     dw      SFTABL          ; DOSDATA:00CCh
00000F43 [F901]                     dw      CARPOS
00000F45 C900                       dw      0C9h
00000F47 [4D11]                     dw      UNKNOWN1        ; ? (points to DOSDATA:114Dh)
00000F49 0000                       dw      0
00000F4B 0000                       dw      0
                                    ;;;

                                ; M001; SR;
                                ; M001; On DOSMGR call ( cx == 0 ), we need to return a table of offsets of
                                ; M001; some DOS variables. Note that the only really important variable in
                                ; M001; this is User_Id. The other variables are needed only to patch stuff
                                ; M001; which does not need to be done in DOS 5.0.

                                ; 29/12/2022
                                ; (MSDOS 6.21 MSDOS.SYS - DOSDATA:1022h)
                                ; 25/03/2024
                                ; PCDOS 7.1 IBMDOS.COM - DOSDATA:0F4Dh

                                Win386_DOSVars:
00000F4D 05                         db      5       ;Major version 5 ; M001
00000F4E 00                         db      0       ;Minor version 0 ; M001
00000F4F [EC05]                     dw      SAVEDS  ; M001
00000F51 [EA05]                     dw      SAVEBX  ; M001
00000F53 [2103]                     dw      INDOS   ; M001
00000F55 [3E03]                     dw      USER_ID ; M001
00000F57 [1503]                     dw      CritPatch ; M001
00000F59 [8C00]                     dw      UMB_HEAD ; M012

                                ;SR;
                                ; Flag to indicate whether WIN386 is running or not
```

```
                                IsWin386:
00000F5B 00                         db      0

                                ; 09/03/2024 (PCDOS 7.1 IBMDOS.COM)

00000F5C 00                         db      0

                                ; 09/03/2024 (PCDOS 7.1 IBMDOS.COM)
                                %if 0

                                ;M018
                                ; This variable contains the path to the VxD device needed for Win386

                                ; 09/01/2024
                                ;VxDpath: db       'c:\wina20.386',0     ;M018

                                ;End WIN386 support

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

                                ;SR;
                                ; These variables have been added for the special lie support for device
                                ;drivers.
                                ;

                                DriverLoad:
                                    db      1       ;initialized to do special handling
                                BiosDataPtr:
                                    dd      0

                                %endif

                                ; 25/03/2024
                                %if 1
                                ; 29/12/2022 - Retro DOS v4.1
                                ;%if 0
                                ; 27/04/2019 - Retro DOS v4.0
                                ; 04/11/2022
                                ; DOSDATA:1044h (MSDOS 6.21 & MSDOS 5.0, MSDOS.SYS)

                                ;--------------------------------------------------------------------------
                                ; Patch for Sidekick
                                ;
                                ; A documented method for finding the offset of the Errormode flag in the
                                ; dos swappable data area if for the app to scan in the dos segment (data)
                                ; for the following sequence of instructions.
                                ;
                                ; Ref: Part C, Article 11, pg 356 of MSDOS Encyclopedia
                                ;
                                ; The Offset of Errormode flag is 0320h
                                ;
                                ;--------------------------------------------------------------------------

00000F5D 36F6062003FF               db      036h, 0F6h, 06h, 020h, 03h, 0FFh ; test ss:[errormode], -1
00000F63 750C                       db      075h, 0Ch                        ; jnz  NearLabel
00000F65 36FF365803                 db      036h, 0FFh, 036h, 058h, 03h      ; push ss:[NearWord]
00000F6A CD28                       db      0CDh, 028h                       ; int  28h

                                ;--------------------------------------------------------------------------
                                ; Patch for PortOfEntry - M036
                                ;
                                ; PortOfEntry by Sector Technology uses an un documented way of determining
                                ; the offset of Errormode flag. The following patch is to support them in
                                ; DOS 5.0. The corresponding code is actually in msdisp.asm
                                ;
                                ;--------------------------------------------------------------------------

00000F6C 803E200300                 db      080h, 03Eh, 020h, 03h, 00h       ; cmp  [errormode], 0
00000F71 7537                       db      075h, 037h                       ; jnz  NearLabel
00000F73 BCA00A                     db      0BCh, 0A0h, 0Ah                         ; mov  sp, dosdata:iostack

                                %endif ; 29/12/2022

                                ; DOSDATA:105Dh (MSDOS 6.21, MSDOS.SYS)
                                ; 25/03/2024
                                ; PCDOS 7.1 IBMDOS.COM - DOSDATA:0F76h

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

                                ;*** New FCB Implementation
                                ; This variable is used as a cache in the new FCB implementation to remember
                                ;the address of a local SFT that can be recycled for a regenerate operation

00000F76 00000000               LocalSFT: dd       0             ; 0 to indicate invalid pointer

                                ;DOSDATA ENDS

                                ;============================================================================
                                ; LMSTUB.ASM (MSDOS 6.0, 1991)
                                ;============================================================================
                                ; 27/04/2019 - Retro DOS 4.0
                                ; 25/03/2024 - Retro DOS 5.0

                                ;DOSDATA  SEGMENT WORD PUBLIC 'DATA'

                                ;--------------------------------------------------------------------------
                                ;   Low Memory Stub for DOS when DOS runs in HMA
                                ;--------------------------------------------------------------------------

                                    ;db     90h

                                    ;EVEN
                                align 2

                                ; DOSDATA:1062h (MSDOS 5.0-6.22, MSDOS.SYS)
                                ; 25/03/2024
                                ; PCDOS 7.1 IBMDOS.COM - DOSDATA:0F7Ah

                                DOSINTTABLE:       ; LABEL DWORD

                                    ;DW     OFFSET DOSCODE:DIVOV          , 0
                                    ;DW     OFFSET DOSCODE:QUIT           , 0
                                    ;DW     OFFSET DOSCODE:COMMAND        , 0
                                    ;DW     OFFSET DOSCODE:ABSDRD         , 0
                                    ;DW     OFFSET DOSCODE:ABSDWRT        , 0
                                    ;DW     OFFSET DOSCODE:Stay_resident  , 0
                                    ;DW     OFFSET DOSCODE:INT2F          , 0
                                    ;DW     OFFSET DOSCODE:CALL_ENTRY     , 0
                                    ;DW     OFFSET DOSCODE:IRETT          , 0

00000F7A [D55F]0000                 dw      DIVOV          , 0  ; DOSINTTABLE+0
00000F7E [C502]0000                 dw      QUIT           , 0  ; DOSINTTABLE+4
```

```
54291 00000F82 [F102]0000                dw      COMMAND         , 0  ; DOSINTTABLE+8
54292 00000F86 [2D05]0000                dw      ABSDRD          , 0  ; DOSINTTABLE+12
54293 00000F8A [DF05]0000                dw      ABSDWRT         , 0  ; DOSINTTABLE+16
54294 00000F8E [3E72]0000                dw      STAY_RESIDENT   , 0  ; DOSINTTABLE+20
54295 00000F92 [3A07]0000                dw      INT2F           , 0  ; DOSINTTABLE+24
54296 00000F96 [CC02]0000                dw      CALL_ENTRY      , 0  ; DOSINTTABLE+28
54297
54298                                   ; 25/03/2024 - Retro DOS v5.0
54299                                   ; PCDOS 7.1 IBMDOS.COM
54300                                   %if 0
54301                                       dw      IRETT           , 0  ; DOSINTTABLE+32
54302                                   %endif
54303
54304 00000F9A 0000                    SS_Save: dw 0               ; save user's stack segment
54305 00000F9C 0000                    SP_Save: dw 0               ; save user's stack offset
54306
54307                                   ;--------------------------------------------------------------------------
54308                                   ;
54309                                   ; LOW MEM STUB:
54310                                   ;
54311                                   ; The low mem stub contains the entry points into DOS for all interrupts
54312                                   ; handled by DOS. This stub is installed if the user specifies that the
54313                                   ; DOS load in HIMEM. Each entry point does this.
54314                                   ;
54315                                   ;   1. if jmp to 8 has been patched out
54316                                   ;      2. if A20 OFF
54317                                   ;         3. Enable A20
54318                                   ;      4. else
54319                                   ;         5. just go to dos entry
54320                                   ;      6. endif
54321                                   ;   7. else
54322                                   ;      8. just go to dos entry
54323                                   ;   9. endif
54324                                   ;
54325                                   ;--------------------------------------------------------------------------
54326
54327                                   ; 27/04/2019 - Retro DOS v4.0
54328
54329                                   ; DOSDATA:108Ah (MSDOS 6.21, MSDOS.SYS)
54330
54331                                   ; 25/03/2024 - Retro DOS v5.0
54332                                   ; PCDOS 7.1 IBMDOS.COM - DOSDATA:0F9Eh
54333
54334                                   ;--------------------------------------------------------------------------
54335                                   ;
54336                                   ; DIVIDE BY 0 handler
54337                                   ;
54338                                   ;--------------------------------------------------------------------------
54339
54340                                   ldivov:
54341                                       ; The following jump, skipping the XMS calls will be patched to
54342                                       ; NOPS by SEG_REINIT if DOS successfully loads high. This jump is
54343                                       ; needed because the stub is installed even before the XMS driver
54344                                       ; is loaded if the user specifies dos=high in the config.sys
54345                                   i0patch:
54346 00000F9E EB03                        jmp     short divov_cont
54347
54348 00000FA0 E8E200                      call    EnsureA20ON             ; we must turn on A20 if OFF
54349                                   divov_cont:
54350 00000FA3 2EFF2E[7A0F]                jmp     far [cs:DOSINTTABLE]  ; jmp to DOS
54351
54352                                   ;--------------------------------------------------------------------------
54353                                   ;
54354                                   ; INT 20h Handler
54355                                   ;
54356                                   ; Here we do not have to set up the stack to return here as the abort call
54357                                   ; will return to the address after the int 21 ah=4b call. This would be the
54358                                   ; common exit point if A20 had been OFF (for TOGGLE DOS) and the A20 line
54359                                   ; will be restored then.
54360                                   ;
54361                                   ;--------------------------------------------------------------------------
54362
54363                                   lquit:
54364                                       ; The following jump, skipping the XMS calls will be patched to
54365                                       ; NOPS by SEG_REINIT if DOS successfully loads high. This jump is
54366                                       ; needed because the stub is installed even before the XMS driver
54367                                       ; is loaded if the user specifies dos=high in the config.sys
54368                                   i20patch:
54369 00000FA8 EB03                        jmp     short quit_cont
54370
54371 00000FAA E8D800                      call    EnsureA20ON             ; we must turn on A20 if OFF
54372                                   quit_cont:
54373 00000FAD 2EFF2E[7E0F]                jmp     far [cs:DOSINTTABLE+4]; jump to DOS
54374
54375                                   ;--------------------------------------------------------------------------
54376                                   ;
54377                                   ; INT 21h Handler
54378                                   ;
54379                                   ;--------------------------------------------------------------------------
54380
54381                                   lcommand:
54382                                       ; The following jump, skipping the XMS calls will be patched to
54383                                       ; NOPS by SEG_REINIT if DOS successfully loads high. This jump is
54384                                       ; needed because the stub is installed even before the XMS driver
54385                                       ; is loaded if the user specifies dos=high in the config.sys
54386                                   i21patch:
54387 00000FB2 EB03                        jmp     short command_cont
54388
54389 00000FB4 E8CE00                      call    EnsureA20ON             ; we must turn on A20 if OFF
54390                                   command_cont:
54391 00000FB7 2EFF2E[820F]                jmp     far [cs:DOSINTTABLE+8]; jmp to DOS
54392
54393                                   ;--------------------------------------------------------------------------
54394                                   ;
54395                                   ; INT 25h
54396                                   ;
54397                                   ;--------------------------------------------------------------------------
54398
54399                                   labsdrd:
54400                                       ; The following jump, skipping the XMS calls will be patched to
54401                                       ; NOPS by SEG_REINIT if DOS successfully loads high. This jump is
54402                                       ; needed because the stub is installed even before the XMS driver
54403                                       ; is loaded if the user specifies dos=high in the config.sys
54404                                   i25patch:
54405 00000FBC EB03                        jmp     short absdrd_cont
54406
54407 00000FBE E8C400                      call    EnsureA20ON             ; we must turn on A20 if OFF
54408                                   absdrd_cont:
54409 00000FC1 2EFF2E[860F]                jmp     far [cs:DOSINTTABLE+12]       ; jmp to DOS
54410
54411                                   ;--------------------------------------------------------------------------
54412                                   ;
54413                                   ; INT 26h
54414                                   ;
```

```
                                ;--------------------------------------------------------------------------
                                
                                labsdwrt:
                                    ; The following jump, skipping the XMS calls will be patched to
                                    ; NOPS by SEG_REINIT if DOS successfully loads high. This jump is
                                    ; needed because the stub is installed even before the XMS driver
                                    ; is loaded if the user specifies dos=high in the config.sys
                                i26patch:
00000FC6 EB03                       jmp     short absdwrt_cont

00000FC8 E8BA00                     call    EnsureA20ON             ; we must turn on A20 if OFF
                                absdwrt_cont:
00000FCB 2EFF2E[8A0F]               jmp     far [cs:DOSINTTABLE+16]       ; jmp to DOS

                                ;--------------------------------------------------------------------------
                                ;
                                ; INT 27h
                                ;
                                ;--------------------------------------------------------------------------

                                lstay_resident:
                                    ; The following jump, skipping the XMS calls will be patched to
                                    ; NOPS by SEG_REINIT if DOS successfully loads high. This jump is
                                    ; needed because the stub is installed even before the XMS driver
                                    ; is loaded if the user specifies dos=high in the config.sys
                                i27patch:
00000FD0 EB03                       jmp     short sr_cont

00000FD2 E8B000                     call    EnsureA20ON             ; we must turn on A20 if OFF
                                sr_cont:
00000FD5 2EFF2E[8E0F]               jmp     far [cs:DOSINTTABLE+20]       ; jmp to DOS

                                ;--------------------------------------------------------------------------
                                ;
                                ; INT 2Fh
                                ;
                                ;--------------------------------------------------------------------------

                                lint2f:
                                    ; The following jump, skipping the XMS calls will be patched to
                                    ; NOPS by SEG_REINIT if DOS successfully loads high. This jump is
                                    ; needed because the stub is installed even before the XMS driver
                                    ; is loaded if the user specifies dos=high in the config.sys
                                i2fpatch:
00000FDA EB03                       jmp     short int2f_cont

00000FDC E8A600                     call    EnsureA20ON             ; we must turn on A20 if OFF
                                int2f_cont:
00000FDF 2EFF2E[920F]               jmp     far [cs:DOSINTTABLE+24]       ; jmp to DOS

                                ;--------------------------------------------------------------------------
                                ;
                                ; CPM entry
                                ;
                                ;--------------------------------------------------------------------------

                                lcall_entry:
                                    ; The following jump, skipping the XMS calls will be patched to
                                    ; NOPS by SEG_REINIT if DOS successfully loads high. This jump is
                                    ; needed because the stub is installed even before the XMS driver
                                    ; is loaded if the user specifies dos=high in the config.sys
                                cpmpatch:
00000FE4 EB03                       jmp     short callentry_cont

00000FE6 E89C00                     call    EnsureA20ON             ; we must turn on A20 if OFF
                                callentry_cont:
00000FE9 2EFF2E[960F]               jmp     far [cs:DOSINTTABLE+28]       ; jmp to DOS

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

                                ; 25/03/2024 - Retro DOS v5.0
                                ; PCDOS 7.1 IBMDOS.COM
                                %if 0

                                lirett:
                                    iret

                                %endif

                                ;--------------------------------------------------------------------------
                                ;
                                ; LowIntXX:
                                ;
                                ; Interrupts from DOS that pass control to a user program must be done from
                                ; low memory, as the user program may change the state of the A20 line or
                                ; they may require that the A20 line be OFF. The following piece of code is
                                ; far call'd from the following places in DOS:
                                ;
                                ;   1. msctrlc.asm where dos issues an int 23h (ctrlc)
                                ;   2. msctrlc.asm where dos issues an int 24h (critical error)
                                ;   3. msctrlc.asm where dos issues an int 28h (idle int)
                                ;
                                ; The int 23 and int 24 handlers may decide to do a far return instead of an
                                ; IRET ane leave the flags on the stack. Therefore we save the return address
                                ; before doing the ints and then do a far junp back into DOS.
                                ;
                                ;--------------------------------------------------------------------------

                                ; 25/03/2024 - Retro DOS v5.0
                                ; PCDOS 7.1 IBMDOS.COM - DOSDATA:0FEEh
                                ; (MSDOS 5.0-6.22 MSDOS.SYS - DOSDATA:10DBh)

00000FEE 00000000               DosRetAddr23:       dd      0
00000FF2 00000000               DosRetAddr24:       dd      0

                                ; 25/03/2024 - Retro DOS v5.0
                                ; PCDOS 7.1 IBMDOS.COM
                                %if 0
                                DosRetAddr28:       dd      0
                                %endif

                                    ; Execute int 23h from low memory
                                LowInt23:
                                                                        ; save the return address that is on
                                                                        ; the stack
00000FF6 2E8F06[EE0F]               pop     word [cs:DosRetAddr23]
00000FFB 2E8F06[F00F]               pop     word [cs:DosRetAddr23+2]

00001000 CD23                       int     23h                     ; ctrl C
                                                                    ; turn on A20 it has been turned OFF
                                                                    ; by int 28/23/24 handler.

00001002 E88000                     call    EnsureA20ON             ; M011: we must turn on A20 if OFF
```

```
00001005 2EFF2E[EE0F]               jmp     far [cs:DosRetAddr23] ; jump back to DOS


                                    ; Execute int 24h from low memory
                                LowInt24:
                                                                  ; save the return address that is on
                                                                  ; the stack
0000100A 2E8F06[F20F]               pop     word [cs:DosRetAddr24]
0000100F 2E8F06[F40F]               pop     word [cs:DosRetAddr24+2]

00001014 CD24                       int     24h                   ; crit error
                                                                  ; turn on A20 it has been turned OFF
                                                                  ; by int 28/23/24 handler.

00001016 E86C00                     call    EnsureA20ON           ; M011: we must turn on A20 if OFF

00001019 2EFF2E[F20F]               jmp     far [cs:DosRetAddr24] ; jump back to DOS


                                    ; Execute int 28h from low memory
                                LowInt28:
0000101E CD28                       int     28h                   ; idle int
                                                                  ; turn on A20 it has been turned OFF
                                                                  ; by int 28/23/24 handler.

00001020 E86200                     call    EnsureA20ON           ; M011: we must turn on A20 if OFF

00001023 CB                         retf

                                ; DOSDATA:1115h (MSDOS 6.21, MSDOS.SYS)

                                ;--------------------------------------------------------------------------
                                ;
                                ; int 21 ah=4b (exec) call will jump to the following label before xferring
                                ; control to the exec'd program. We turn off A20 inorder to allow programs
                                ; that have been packed by the faulty exepack utility to unpack correctly.
                                ; This is so because exepac'd programs rely on address wrap.
                                ;
                                ;--------------------------------------------------------------------------

                                ; 25/03/2024 - Retro DOS v5.0
                                ; PCDOS 7.1 IBMDOS.COM - DOSDATA:1024h

                                disa20_xfer:
00001024 E84800                     call    XMMDisableA20         ; disable A20

                                    ; Look at msproc.asm at label exec_go for understanding the following:

                                    ; DS:SI points to entry point
                                    ; AX:DI points to initial stack
                                    ; DX has PDB pointer
                                    ; BX has initial AX value

00001027 FA                         cli
00001028 2EC606[2103]00             mov     byte [cs:INDOS],0     ; SS Override

                                    ; 25/03/2024 (PCDOS 7.1 IBMDOS.COM)
                                    ;;;
0000102E 36C606[B812]00             mov     byte [ss:INDOS_FLAG],0
                                    ;;;

00001034 8ED0                       mov     SS,AX                 ; set up user's stack
00001036 89FC                       mov     SP,DI                 ; and SP
00001038 FB                         sti

00001039 1E                         push    DS                    ; fake long call to entry
0000103A 56                         push    SI
0000103B 8EC2                       mov     ES,DX                 ; set up proper seg registers
0000103D 8EDA                       mov     DS,DX
0000103F 89D8                       mov     AX,BX                 ; set up proper AX
00001041 CB                         retf

                                ;--------------------------------------------------------------------------
                                ;
                                ; M003:
                                ;
                                ; If an int 21 ah=25 call is made immediately after an exec call, DOS will
                                ; come here, turn A20 OFF restore user stack and registers before returning
                                ; to user. This is done in dos\msdisp.asm. This has been done to support
                                ; programs compiled with MS PASCAL 3.2. See under TAG M003 in DOSSYM.INC for
                                ; more info.
                                ;
                                ; Also at this point DS is DOSDATA. So we can assume DS DOSDATA. Note that
                                ; SS is also DOS stack. It is important that we do the XMS call on DOS's
                                ; stack to avoid additional stack overhead for the user.
                                ;
                                ; --------------------------------------------------------------------------

                                disa20_iret:
00001042 E82A00                     call    XMMDisableA20
00001045 FE0E[2103]                 dec     byte [INDOS]

                                    ; 25/03/2024 (PCDOS 7.1 IBMDOS.COM)
                                    ;;;
00001049 FE0E[B812]                 dec     byte [INDOS_FLAG]
                                    ;;;

0000104D 8E16[8605]                 mov     SS,[USER_SS]          ; restore user stack
00001051 8B26[8405]                 mov     SP,[USER_SP]
00001055 89E5                       mov     BP,SP
                                    ;mov    [BP+user_env.user_AX],AL
00001057 884600                     mov     [bp],al

                                ; 25/03/2024
                                %if 0
                                    mov     AX,[NSP]
                                    mov     [USER_SP],AX
                                    mov     AX,[NSS]
                                    mov     [USER_SS],AX
                                %else
                                    ; 25/03/2024 (PCDOS 7.1 IBMDOS.COM)
                                    ;;;
                                    ;les    ax, dword ptr ds:NSS
0000105A C406[F005]                 les     ax,[NSS]
0000105E 8C06[8405]                 mov     [USER_SP],es
00001062 A3[8605]                   mov     [USER_SS],ax
                                    ;;;
                                %endif
00001065 58                         pop     AX                    ; restore user regs
00001066 5B                         pop     BX
00001067 59                         pop     CX
00001068 5A                         pop     DX
00001069 5E                         pop     SI
```

```
54663 0000106A 5F                       pop     DI
54664 0000106B 5D                       pop     BP
54665 0000106C 1F                       pop     DS
54666 0000106D 07                       pop     ES
54667                                 lirett:           ; 25/03/2024 (PCDOS 7.1 IBMDOS.COM)
54668 0000106E CF                       iret
54669
54670                                 ;****************************************************************************
54671                                 ;***XMMDisableA20 - switch 20th address line
54672                                 ;
54673                                 ;   This routine is used to disable the 20th address line in
54674                                 ;   the system using XMM calls.
54675                                 ;
54676                                 ;   ENTRY   none          ;ds = _DATA
54677                                 ;   EXIT    A20 line disabled
54678                                 ;   USES    NOTHING
54679                                 ;
54680                                 ;****************************************************************************
54681
54682                                 XMMDisableA20:
54683 0000106F 53                       push    bx
54684 00001070 50                       push    ax
54685                                   ;mov    ah,XMM_LOCAL_DISABLE_A20
54686 00001071 B406                     mov     ah,6
54687 00001073 2EFF1E[7B10]             call    far [cs:XMMcontrol]
54688 00001078 58                       pop     ax
54689 00001079 5B                       pop     bx
54690 0000107A C3                       retn
54691
54692                                 ; The entry point in the BIOS XMS driver is defined here.
54693
54694                                 XMMcontrol:
54695 0000107B 00000000                 dd      0
54696
54697                                 ;--------------------------------------------------------------------------
54698                                 ; 24/03/2024 - Retro DOS v5.0
54699                                 ; PCDOS 7.1 IBMDOS.COM - DOSDATA:107Fh
54700                                 ;;;
54701 0000107F 00000000                 dd      0
54702                                 SC_STATUS:
54703 00001083 0000                     dw      0
54704                                 ;;;
54705
54706                                 ;--------------------------------------------------------------------------
54707                                 ;
54708                                 ;***EnsureA20ON - Ensures that A20 is ON
54709                                 ;
54710                                 ;   This routine is used to query the A20 state in
54711                                 ;   the system using XMM calls.
54712                                 ;
54713                                 ;   ENTRY: none
54714                                 ;
54715                                 ;   EXIT : A20 will be ON
54716                                 ;
54717                                 ;   USES : NONE
54718                                 ;
54719                                 ;--------------------------------------------------------------------------
54720
54721                                 ; 19/09/2023
54722                                 ;LowMemory: ; label dword         ; Set equal to 0000:0080
54723                                 ;   dw      00080h
54724                                 ;   dw      00000h
54725                                 ;
54726                                 ;HighMemory:       ; label dword
54727                                 ;   dw      00090h                ; Set equal to FFFF:0090
54728                                 ;   dw      0FFFFh
54729
54730                                     ; 25/03/2024 - Retro DOS v5.0
54731                                     ; PCDOS 7.1 IBMDOS.COM - DOSDATA:1085h
54732                                     ; (MSDOS 5.0-6.22 MSDOS.SYS - DOSDATA:116Fh)
54733
54734                                 EnsureA20ON:
54735 00001085 9C                       pushf
54736 00001086 1E                       push    ds
54737 00001087 06                       push    es
54738 00001088 51                       push    cx
54739 00001089 56                       push    si
54740 0000108A 57                       push    di
54741
54742                                     ; 19/09/2023
54743                                     ;lds    si,[cs:LowMemory]     ; Compare the 4 words at 0000:0080
54744                                     ;les    di,[cs:HighMemory]    ; with the 4 at FFFF:0090
54745
54746                                     ; 25/03/2024
54747                                     ; PCDOS 7.1 IBMDOS.COM
54748                                     ;;;
54749 0000108B 31F6                     xor     si,si
54750 0000108D 8EDE                     mov     ds,si
54751 0000108F 4E                       dec     si
54752 00001090 BF9000                   mov     di,90h  ; 0FFFFh:0090h ; HighMemory
54753 00001093 8EC6                     mov     es,si
54754 00001095 BE8000                   mov     si,80h  ; 0000h:0080h  ; LowMemory
54755                                   ;;;
54756
54757 00001098 B90400                   mov     cx,4
54758 0000109B FC                       cld
54759 0000109C F3A7                     repe    cmpsw
54760
54761 0000109E 7407                     jz      short EA20_OFF
54762                                 EA20_RET:
54763 000010A0 5F                       pop     di
54764 000010A1 5E                       pop     si
54765 000010A2 59                       pop     cx
54766 000010A3 07                       pop     es
54767 000010A4 1F                       pop     ds
54768 000010A5 9D                       popf
54769 000010A6 C3                       retn
54770
54771                                 EA20_OFF:
54772                                     ; We are going to do the XMS call on the DOS's AuxStack.
54773                                     ; NOTE: ints are disabled at this point.
54774
54775 000010A7 53                       push    bx
54776 000010A8 50                       push    ax
54777
54778                                 ; 25/03/2024
54779                                 %if 0
54780                                     mov     ax,ss                 ; save user's stack pointer
54781                                     mov     [cs:SS_Save],ax
54782                                     mov     [cs:SP_Save],sp
54783                                     mov     ax,cs
54784                                     mov     ss,ax
54785                                 %else
54786                                     ; 25/03/2024 (PCDOS 7.1 IBMDOS.COM)
```

```
                                    ;;;
000010A9 8CC8                       mov     ax,cs
000010AB 2E8C16[9A0F]               mov     [cs:SS_Save],ss
000010B0 2E8926[9C0F]               mov     [cs:SP_Save],sp
000010B5 8ED0                       mov     ss,ax
                                    ;;;
                                %endif
000010B7 BC[A007]                   mov     sp,AUXSTACK
                                                            ; ss:sp -> DOSDATA:AuxStack
                                    ;mov    ah,XMM_LOCAL_ENABLE_A20
000010BA B405                       mov     ah,5
000010BC 2EFF1E[7B10]               call    far [cs:XMMcontrol]
000010C1 09C0                       or      ax,ax
000010C3 740E                       jz      short XMMerror          ; AX = 0 fatal error

                                    ;mov    ax,[cs:SS_Save]                 ; restore user stack
                                    ;mov    ss,ax
                                    ;  25/03/2024 (PCDOS 7.1 IBMDOS.COM)
                                    ;;;
000010C5 2E8E16[9A0F]               mov     ss,[cs:SS_Save]
                                    ;;;
000010CA 2E8B26[9C0F]               mov     sp,[cs:SP_Save]

000010CF 58                         pop     ax
000010D0 5B                         pop     bx

000010D1 EBCD                       jmp     short EA20_RET

                                XMMerror:                           ; M006 - Start
000010D3 B40F                       mov     ah,0Fh                  ; get video mode
000010D5 CD10                       int     10h
000010D7 3C07                       cmp     al,7                    ; Q: are we an MDA
000010D9 7405                       je      short XMMcont           ; Y: do not change mode
                                    ;xor    ah,ah ; 0               ; set video mode
                                    ;mov    al,02h                  ; 80 X 25 text
                                    ; 25/03/2024 (PCDOS 7.1 IBMDOS.COM)
                                    ;;;
000010DB B80200                     mov     ax,2
                                    ;;;
000010DE CD10                       int     10h
                                XMMcont:
                                    ;mov    ah,05h                  ; set display page
                                    ;xor    al,al                   ; page 0
                                    ; 25/03/2024 (PCDOS 7.1 IBMDOS.COM)
                                    ;;;
000010E0 B80005                     mov     ax,500h
                                    ;;;
000010E3 CD10                       int     10h

000010E5 BE[1312]                   mov     si,XMMERRMSG
000010E8 0E                         push    cs
000010E9 1F                         pop     ds
000010EA FC                         cld                             ; clear direction flag
                                XMMprnt:
000010EB AC                         lodsb
000010EC 3C24                       cmp     al,'$'                  ; indicates end of XMMERRMSG
000010EE 7409                       jz      short XMMStall          ; function 0Eh
000010F0 B40E                       mov     ah,0Eh
000010F2 BB0700                     mov     bx,7
000010F5 CD10                       int     10h
000010F7 EBF2                       jmp     short XMMprnt

                                XMMStall:
000010F9 FB                         sti                             ; allow the user to warm boot
000010FA EBFD                       jmp     short XMMStall          ; M006 - End

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

                                ; 27/04/2019 - Retro DOS v4.0

                                ; retrodos4.s ; offset 0Ch in BIOS segment (0070h)
                                ALTAH       equ 0Ch

                                ;This has been put in for WIN386 2.XX support. The format of the instance
                                ;table was different for this. Segments will be patched in at init time.

                                    ; 25/03/2024
                                    ; PCDOS 7.1 IBMDOS.COM - DOSDATA:10FCh
                                    ; (MSDOS 5.0-6.22 MSDOS.SYS - DOSDATA:11E7h)

                                OldInstanceJunk:
000010FC 7000                       dw      70h     ;segment of BIOS
000010FE 0000                       dw      0       ;indicate stacks in SYSINIT area
00001100 0600                       dw      6       ;6 instance items

                                    ;dw     0,offset dosdata:contpos, 2
                                    ;dw     0,offset dosdata:bcon, 4
                                    ;dw     0,offset dosdata:carpos,106h
                                    ;dw     0,offset dosdata:charco, 1
                                    ;dw     0,offset dosdata:exec_init_sp, 34               ;M032
                                    ;dw     070h,offset BData:altah, 1     ; altah byte in bios

00001102 0000[2200]0200             dw      0,CONTPOS,2
00001108 0000[3200]0400             dw      0,BCON,4
0000110E 0000[F901]0601             dw      0,CARPOS,106h
00001114 0000[0003]0100             dw      0,CHARCO,1
0000111A 0000[BF0E]2200             dw      0,exec_init_SP,34
00001120 70000C000100               dw      70h,ALTAH,1     ; altah byte in bios

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

                                ; 24/03/2024
                                ; 17/01/2024
                                %if 0

                                ; M021-
                                ;
                                ; DosHasHMA - This flag is set by seg_reinit when the DOS actually
                                ;   takes control of the HMA. When running, this word is a reliable
                                ;   indicator that the DOS is actually using HMA. You can't just use
                                ;   CS, because ROMDOS uses HMA with CS < F000.

                                DosHasHMA:
                                    db      0
                                FixExePatch:
                                    dw      0               ; M012

                                ; 21/03/2024 - Retro DOS v5.0
                                ;;28/12/2022 - Retro DOS v4.1
                                RationalPatchPtr:
                                    dw      0               ; M012

                                %endif
```

```
                                ; End M021
                                
                                ;--------------------------------------------------------------------------
                                
                                ; 21/03/2024 - Retro DOS v5.0
                                %if 1
                                ; 28/12/2022 - Retro DOS v4.1
                                ;%if 0
                                
                                ; M020 Begin
                                
                                RatBugCode: ; proc  far
00001126 51                         push    cx
00001127 8B0E1000                   mov     cx,[10h]
                                rbc_loop:
                                    ;loop   $
0000112B E2FE                       loop    rbc_loop
0000112D 59                         pop     cx
0000112E CB                         retf
                                
                                ; M020 End
                                
                                %endif
                                
                                ;--------------------------------------------------------------------------
                                
                                UmbSave1:
                                    ;db     11 dup (?)      ; M023
0000112F 00<rep Bh>                 times   11 db 0
                                
                                ;--------------------------------------------------------------------------
                                ; 16/01/2024 - Retrodos v5.0
                                ; PCDOS 7.1 IBMDOS.COM - DOSDATA: 113Ah
                                
                                OLD_FIRSTCLUS_HW:
0000113A 0000                       dw      0
                                
                                ; 17/01/2024
                                %if 1
                                
                                ; DOS 3.3 F.C. 6/12/86
                                ; FASTOPEN communications area DOS 3.3   F.C. 5/29/86
                                
                                FastTable:                          ; a better name
                                FastOpenTable:
0000113C 0200                       dw      2                         ; number of entries
0000113E [8218]                     dw      FastRet                       ; pointer to ret instr.
00001140 0000                       dw      0                         ; and will be modified by
00001142 [8218]                     dw      FastRet                       ; FASTxxx when loaded in
00001144 0000                       dw      0
                                
                                ; DOS 3.3 F.C. 6/12/86
                                
                                FastFlg:                            ; flags
                                FastOpenFlg:
00001146 00                         db      0                       ; don't change the foll: order
                                
                                %endif
                                
                                ; 24/03/2024 - Retrto DOS v5.0
                                ; PCDOS 7.1 IBMDOS.COM - DOSDATA:1147h
                                %if 1
                                
                                FastOpen_Ext_Info:
                                    ;db     6 dup(0)
00001147 00<rep 6h>                 times   6 db 0
                                UNKNOWN1:
0000114D 0000                       dw      0
                                    ;db     5 dup(0)
0000114F 00<rep 5h>                 times   5 db 0
                                PATHNAMELEN:
00001154 4300                       dw      67
                                
                                    ;db     31 dup(0)
00001156 00<rep 1Fh>                times   31 db 0
                                %endif
                                
                                ; 17/01/2024
                                ; PCDOS 7.1 IBMDOS.COM - DOSDATA:1175h
                                CurSC_DRIVE:
00001175 FF                         db      -1  ; 0FFh      ; current SC drive
                                
                                ;M044
                                ; Second part of save area for saving last para of Windows memory
                                
                                ; 17/01/2024
                                WinoldPatch2:
                                    ;db     8 dup (?)       ; M044
00001176 00<rep 8h>                 times   8 db 0
                                
                               FIRST_BUFF_ADDR:
0000117E 0000                       dw      0
                                
                                ;--------------------------------------------------------------------------
                                ; 17/01/2024 - Retrodos v5.0
                                ; PCDOS 7.1 IBMDOS.COM
                                ; DOSDATA:1180h
                                
                                %if 1
                                
                                ;==========================================================================
                                ; WPATCH.INC (MSDOS 6.0, 1991)  ;;; Windows 3.1 patches ;;;
                                ;==========================================================================
                                ; 27/04/2019 - Retro DOS 4.0
                                
                                ; DOSDATA:12CFh (MSDOS 6.21, MSDOS.SYS)
                                
                                ; Retro DOS v5.0
                                ; PCDOS 7.1 IBMDOS.COM - DOSDATA:1180h
                                
                                ; 05/11/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
                                ; DOSDATA:12CFh (MSDOS 5.0, MSDOS.SYS)
                                
                                ; first and second DOS patches
                                ;   Non-console device read/write (system calls 3Fh and 40h)
                                ;
                                ; Code in disk.asm, 2 locations, one for read, one for write
                                ;   DVRDLP:
                                ;   DVWRTLP:
                                ;
                                ;
                                ; 036h      lds     si,SS:[????]                        ; ThisSFT
                                ;   lds     si,si+7                             ; sf_devptr
                                ; 0E8h      call    ????            <- "simulate" int28 event ; DSKSTATCHK
```

```
55035
55036 00001180 36C536                DOSP1_ID:   db      036h,0C5h,036h
55037 00001183 3605C57407E8          DOSP1_THISSFT:      db      036h,005h,0C5h,074h,007h,0E8h
55038                                DOSP1_ID_LEN        equ     $-DOSP1_ID
55039
55040 00001189 9090                            db      90h, 90h
55041
55042 0000118B 36C536                DOSP12_ID:  db      036h,0C5h,036h
55043 0000118E 3605C57407E8          DOSP12_THISSFT:     db      036h,005h,0C5h,074h,007h,0E8h
55044                                DOSP12_ID_LEN       equ     $-DOSP1_ID
55045
55046                                ; DOSDATA:12E3h
55047
55048                                ; Third/Fourth DOS patch - System call 3Fh (Read) from console
55049                                ;
55050                                ; Code in disk.asm, 1 location
55051                                ;   GETBUF:
55052                                ;
55053                                ; 051h      push    cx      <- begin special int28 mode
55054                                ;   push    es
55055                                ;   push    di
55056                                ;   mov     dx,???? ; offset dosgroup:CONBUF
55057                                ;   call    ????    ; $STD_CON_STRING_INPUT
55058                                ;   pop     di
55059                                ;   pop     es
55060                                ; 059h      pop     cx      <- end special int28 mode
55061
55062 00001194 510657BA              DOSP3_ID:   db      051h,006h,057h,0BAh
55063 00001198 2902E8                DOSP3_CONBUF:       db      029h,002h,0E8h
55064                                DOSP3_ID_LEN        equ     $-DOSP3_ID
55065 0000119B 9AE35F07                        db      09Ah,0E3h,05Fh,007h   ; ???? , pop di, pop es
55066 0000119F 59                    DOSP4_ID:   db      059h                  ; pop cx
55067                                DOSP4_ID_OFF        equ     (DOSP4_ID - DOSP3_ID)
55068
55069                                ; DOSDATA:12EFh
55070
55071                                ; Fifth DOS patch - System call 40h (Write) to console
55072                                ;
55073                                ; Code in disk.asm, 1 location
55074                                ;
55075                                ;           push    cx
55076                                ;       WRCONLP: lodsb
55077                                ;           cmp     al,1Ah
55078                                ;           jz      ????
55079                                ;           call    ????    <- "simulate" int28 event
55080                                ;           loop    WRCONLP
55081                                ;       CONEOF:     pop     ax
55082
55083 000011A0 51                    DOSP5_ID:   db      051h                  ; push cx
55084 000011A1 AC3C1A7405                      db      0ACh,03Ch,01Ah,074h,005h
55085 000011A6 E8                              db      0E8h                  ; call
55086                                DOSP5_ID_LEN        equ     $-DOSP5_ID
55087
55088                                ; DOSDATA:12F6h
55089
55090                                ; Seventh DOS patch - System call entry, patch USER_ID with VMid for share
55091                                ;
55092                                ; Code in disp.asm, 1 location
55093                                ;
55094                                ;
55095                                ;   mov [SaveDS],ds
55096                                ;   mov [SaveBX],bx
55097                                ;   mov bx,cs
55098                                ;   mov ds,bx
55099                                ;   inc [indos]
55100                                ;   xor ax,ax
55101                                ;   mov [USER_ID],AX       <- Patch to set USER_ID to VMID
55102
55103 000011A7 2E8C1E                DOSP7_ID:   db      02Eh,08Ch,01Eh
55104 000011AA 7E05                  DOSP7_SAVEDS:       db      07Eh,05h              ; mov [SaveDS],ds
55105 000011AC 2E891E                          db      02Eh,089h,01Eh
55106 000011AF 7C05                  DOSP7_SAVEBX:       db      07Ch,05h              ; mov [SaveBX],bx
55107 000011B1 8CCB                            db      08Ch,0CBh             ; mov bx,cs
55108 000011B3 8EDB                            db      08Eh,0DBh             ; mov ds,bx
55109 000011B5 FE06                            db      0FEh,006h
55110 000011B7 CF02                  DOSP7_INDOS:        db      0CFh,002h             ; inc [indos]
55111 000011B9 33C0                            db      033h,0C0h             ; xor ax,ax
55112                                DOSP7_ID_LEN        equ     $-DOSP7_ID
55113
55114                                ; DOSDATA:130Ah
55115
55116                                ; Eighth DOS patch - OWNER check in handle calls. For share, need to NOP test
55117                                ;
55118                                ; Code in handle.asm, 1 location in routine CheckOwner
55119                                ;
55120                                ;
55121                                ;
55122                                ;   push    ax
55123                                ;   mov     ax,ss:[USER_ID]     <- patch to XOR AX,AX to set zero
55124                                ;   cmp     ax,es:[di.sf_UID]   <- NOP
55125                                ;   pop     ax
55126                                ;   jz      ????
55127
55128 000011BB 50                    DOSP8_ID:   db      050h                  ; push ax
55129 000011BC 36A1                            db      036h,0A1h
55130 000011BE EA02                  DOSP8_USER_ID:      db      0EAh,002h             ; mov  ax,ss:[USER_ID]
55131 000011C0 263B45                          db      026h,03Bh,045h        ; cmp  ax,es:[di+2F]
55132                                DOSP8_ID_LEN        equ     $-DOSP8_ID
55133 000011C3 2F58                            db      02Fh,058h             ; pop  ax
55134
55135                                ; DOSDATA:1314h
55136
55137                                ; 10th, 11th, 12th DOS patch - System call 3Fh (Read) in raw mode
55138                                ;
55139                                ;   Take RAW read to STDIN SFT and turn it into a polling loop doing
55140                                ;   a yeild when a character is not ready to be read.
55141                                ;
55142                                ; Code in disk.asm, 3 locations
55143                                ;
55144                                ;   DVRDRAW:
55145                                ;       PUSH    ES
55146                                ;       POP     DS
55147                                ;   ReadRawRetry:                       <- Patch 10
55148                                ;       MOV     BX,DI
55149                                ;       XOR     AX,AX                   <- Reenter #2
55150                                ;       MOV     DX,AX
55151                                ;       call    SETREAD
55152                                ;       PUSH    DS                      <- Reenter #1
55153                                ;       LDS     SI,[THISSFT]
55154                                ;       call    DEVIOCALL
55155                                ;       MOV     DX,DI
55156                                ;       MOV     AH,86H
55157                                ;       MOV     DI,[DEVCALL.REQSTAT]
55158                                ;       TEST    DI,STERR
```

```
                                ;       JZ      CRDROK
                                ;       call    CHARHARD
                                ;       MOV     DI,DX
                                ;       OR      AL,AL
                                ;       JZ      CRDROK
                                ;       CMP     AL,3
                                ;       JZ      CRDFERR
                                ;       POP     DS
                                ;       JMP     ReadRawRetry
                                ;
                                ;   CRDFERR:
                                ;       POP     DI                      <- Patch 11
                                ;   DEVIOFERR:
                                ;       LES     DI,[THISSFT]
                                ;       jmp     SET_ACC_ERR_DS
                                ;
                                ;   CRDROK:
                                ;       POP     DI                      <- Patch 12
                                ;       MOV     DI,DX
                                ;       ADD     DI,[CALLSCNT]
                                ;       JMP     SHORT ENDRDDEVJ3

000011C5 061F                   DOSP10_ID:          db      006H,01FH
                                DOSP10_LOC_OFFSET   equ     $-DOSP10_ID
000011C7 8BDF                   DOSP10_LOC:         db      08BH,0DFH
                                DOSP10_REENT2_OFFSET        equ     $-DOSP10_LOC
000011C9 33C08BD0E8                                 db      033H,0C0H,08BH,0D0H,0E8H
                                DOSP10_ID_LEN               equ     $-DOSP10_ID
000011CE DF0E                                       db      0DFH,00EH
                                DOSP10_REENT1_OFFSET        equ     $-DOSP10_LOC
000011D0 1E36C5363605E8AF0E                         db      01EH,036H,0C5H,036H,036H,005H,0E8H,0AFH,00EH
000011D9 8BD7B486368B3E                             db      08BH,0D7H,0B4H,086H,036H,08BH,03EH
                                DOSP10_PACKVAL_OFFSET       equ     $-DOSP10_ID
000011E0 0903                                       db      009H,003H
000011E2 F7C700807419E84717                         db      0F7H,0C7H,000H,080H,074H,019H,0E8H,047H,017H
000011EB 8BFA0AC074103C0374-                        db      08BH,0FAH,00AH,0C0H,074H,010H,03CH,003H,074H,003H
000011F4 03
000011F5 1FEBCF                                     db      01FH,0EBH,0CFH
                                DOSP11_LOC_OFFSET   equ     $-DOSP10_ID
000011F8 5F                                         db      05FH
                                DOSP11_REENT_OFFSET         equ     $-DOSP10_LOC
000011F9 36C43E3605E9A104                           db      036H,0C4H,03EH,036H,005H,0E9H,0A1H,004H

                                DOSP12_LOC_OFFSET   equ     $-DOSP10_ID
00001201 5F8BFA                                     db      05FH,08BH,0FAH
                                ; DOSDATA:1353h

                                ; 13th DOS patch - Actually a SYSINIT patch. Patches the stack fault code
                                ;          which prints the fatal stack fault error on DOS >= 3.20.
                                ;
                                ;       Sets focus to current VM so user can see fatal message.
                                ;
                                ;
                                ;   l0: lodsb               <- Setfocus here
                                ;       cmp al, '$'
                                ;       je l1
                                ;       mov bl, 7
                                ;       mov ah, 0Eh
                                ;       int 10h
                                ;       jmp l0
                                ;   l1: jmp $

00001204 AC                     DOSP13_ID:  db      0ACh                ; l0: lodsb
00001205 3C24                               db      03Ch,024h           ;     cmp al, '$'
00001207 7408                               db      074h,008h           ;     je l1
00001209 B307                               db      0B3h,007h           ;     mov bl, 7
0000120B B40E                               db      0B4h,00Eh           ;     mov ah, 0Eh
0000120D CD10                               db      0CDh,010h           ;     int 10h
0000120F EBF3                               db      0EBh,0F3h           ;     jmp l0
00001211 EBFE                               db      0EBh,0FEh           ; l1: jmp $
                                DOSP13_ID_LEN       equ     $-DOSP13_ID

                                ; 05/11/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
                                ; DOSDATA:1362h (MSDOS 5.0 MSDOS.SYS)

                                ; 05/01/2024
                                %endif      ; 05/11/2022

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

                                ; DOSDATA:122Ah

                                Mark3:      ; label byte

                                ;IF2
                                ;   IF ((OFFSET MARK3) GT (OFFSET COUNTRY_CDPG) )
                                ;       %OUT !DATA CORRUPTION!MARK3 OFFSET TOO BIG. RE-ORGANIZE DATA.
                                ;   ENDIF
                                ;ENDIF

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

                                ; 27/04/2019 - Retro DOS v4.0
                                ; 05/01/2024 - Retro DOS v5.0

                                ;include msdos.cl2               ; XMMERRMSG

                                ; DOSDATA:12B8h (MSDOS 6.22, MSDOS.SYS) ; 17/01/2024
                                ; DOSDATA:1213h (PCDOS 7.1, IBMDOS.COM) ; 05/01/2024

                                XMMERRMSG:
00001213 0D0A                       db      0Dh,0Ah
00001215 413230204861726477-        db      'A20 Hardware Error',0Dh,0Ah,'$'
0000121E 617265204572726F72-
00001227 0D0A24

                                ;--------------------------------------------------------------------------
                                ; 17/01/2024 - Note: COUNTRY_CDPG must be at DOSDATA:122Ah (to 12B8h) addr
                                ; It is fixed at 122Ah in PCDOS 7.1 IBMDOS.COM and MSDOS 5.0-6.22 MSDOS.SYS
                                ;--------------------------------------------------------------------------

                                ;##########################################################################
                                ;
                                ; ** HACK FOR DOS 4.0 REDIR **
                                ;
                                ; The dos 4.X redir requires that country_cdpg is at offset 0122ah. Any new
                                ; data variable that is to be added to DOSDATA must go in between Mark3
                                ; COUNTRY_CDPG if it can.
                                ;
                                ; MARK3 SHOULD NOT BE > 122AH
                                ;
                                ; As of 9/6/90, this area is FULL!
                                ;
                                ;##########################################################################
```

```
                                    ;ORG    0122Ah
                                ; DOSDATA:122Ah (MSDOS 6.21, MSDOS.SYS)
                                ; 09/01/2024 - Retro DOS v5.0
                                ; PCDOS 7.1 IBMDOS.COM - DOSDATA:122Ah
                                ; 17/01/2024
                                ; MSDOS 5.0  MSDOS.SYS - DOSDATA:122Ah
                                ; MSDOS 6.22 MSDOS.SYS - DOSDATA:122Ah
                                ; 09/03/2024
                                ; Windows ME    IO.SYS - DOSDATA:122Ah
                                ; The following table is used for DOS 3.3
                                ;DOS country and code page information is defined here for DOS 3.3.
                                ;The initial value for ccDosCountry is 1 (USA).
                                ;The initial value for ccDosCodepage is 850.
                                ; country and code page information
                                ; ---------------------------------------------------------------------------
0000122A 0000000000000000       COUNTRY_CDPG:      ; label  byte
                                    db   0,0,0,0,0,0,0,0          ; reserved words
00001232 5C434F554E5452592E-        db   '\COUNTRY.SYS',0         ; path name of country.sys
0000123B 53595300
                                    ;db   51 dup (?)
0000123F 00<rep 33h>                times 51 db 0
                                ; ------------------------------------------------<MSKK01>-------------------
                                ;ifdef     DBCS
                                ;  ifdef JAPAN
                                ;   dw   932                     ; system code page id (JAPAN)
                                ;  endif
                                ;  ifdef TAIWAN
                                ;   dw   938                     ; system code page id (TAIWAN)
                                ;  endif
                                ;  ifdef KOREA
                                ;   dw   934                     ; system code page id (KOREA IBM)
                                ;  endif
                                ;else
00001272 B501                       dw   437                     ; system code page id
                                ;endif
                                ; ------------------------------------------------<MSKK01>-------------------
00001274 0600                       dw   6                       ; number of entries
                                COUNTRY_CDPG_76: ; COUNTRY_CDPG + 76
00001276 02                         db   SetUcase  ; 2           ; Ucase type
00001277 [FA0A]                     dw   UCASE_TAB               ;pointer to upper case table
00001279 0000                       dw   0                       ; segment of poiter
0000127B 04                         db   SetUcaseFile  ; 4       ; Ucase file char type
0000127C [7C0B]                     dw   FILE_UCASE_TAB          ;pointer to file upper case table
0000127E 0000                       dw   0                       ; segment of poiter
00001280 05                         db   SetFileList ; 5         ; valid file chars type
00001281 [280D]                     dw   FILE_CHAR_TAB           ;pointer to valid file char tab
00001283 0000                       dw   0                       ; segment of poiter
00001285 06                         db   SetCollate       ; 6          ; collate type
00001286 [FE0B]                     dw   COLLATE_TAB             ;pointer to collate table
00001288 0000                       dw   0                       ; segment of poiter
0000128A 07                         db   SetDBCS   ; 7           ;AN000; DBCS Ev                       2/12/KK
0000128B [000D]                     dw   DBCS_TAB                ;AN000;pointer to DBCS Ev table      2/12/KK
0000128D 0000                       dw   0                       ;AN000; segment of pointer   2/12/KK
0000128F 01                         db   SetCountryInfo  ; 1     ; country info type
00001290 2600                       dw   NEW_COUNTRY_SIZE        ; extended country info size
                                ; ------------------------------------------------<MSKK01>-------------------
                                ;ifdef     DBCS
                                ;   ......
                                ;else
                                    ; 09/01/2024 - Retro DOS v5.0
                                    ; PCDOS 7.1 IBMDOS.COM - DOSDATA:1292h
                                _COUNTRY_ID:
00001292 0100                       dw   1                       ; USA country id
00001294 B501                       dw   437                     ; USA system code page id
                                COUNTRY_CDPG_108: ; COUNTRY_CDPG + 108
00001296 0000                       dw   0                       ; date format
00001298 2400000000                 db   '$',0,0,0,0             ; currency symbol
0000129D 2C00                       db   ',',0                   ; thousand separator
0000129F 2E00                       db   '.',0                   ; decimal separator
000012A1 2D00                       db   '-',0                   ; date separator
000012A3 3A00                       db   ':',0                   ; time separator
000012A5 00                         db   0                       ; currency format flag
000012A6 02                         db   2                       ; # of digits in currency
000012A7 00                         db   0                       ; time format
000012A8 [6B0D]                     dw   MAP_CASE                ; mono case routine entry point
000012AA 0000                       dw   0                       ; segment of entry point
000012AC 2C00                       db   ',',0                   ; data list separator
000012AE 00000000000000000000-      dw   0,0,0,0,0               ; reserved
000012B7 00
                                ;endif
                                ; ------------------------------------------------<MSKK01>-------------------
                                ;---------------------------------------------------------------------------
                                ; 01/01/2024 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMDOS.COM)
                                ; PCDOS 7.1 IBMDOS.COM - DOSDATA:12B8h
                                INDOS_FLAG:
000012B8 00                         db 0           ; duplicated INDOS flag, what for ?
                                                   ; (PCDOS 7.1 kernel CODE always updates it together
                                                   ; with 'INDOS' flag !?)
                                ; 09/01/2024
                                DEVIO_IN_PROGRESS:
000012B9 00                         db 0
                                ; 09/01/2024
                                ; PCDOS 7.1 IBMDOS.COM - DOSDATA:12BAh
                                ; ----------
                                ; INTERNATIONALIZATION INFORMATION
                                ;    for Get/Set Extended Country Info functions
000012BA 454E5500               _ENU:       db 'ENU',0
000012BE 55534100               _USA:       db 'USA',0
000012C2 5553                   _US:db 'US'
000012C4 0100                       dw 1
000012C6 0200                       dw 2
000012C8 0000                       dw 0
000012CA 414D00                 _AM:db 'AM',0
000012CD 504D00                 _PM:db 'PM',0
                                _MMDDYY:
000012D0 4D2F642F7979202020-        db 'M/d/yy     dddd,MMMMdd,yyyy         '
000012D9 2020646464642C4D4D-
000012E2 4D4D64642C79797979-
000012EB 2020202020202020
000012F4 00                         db 0
000012F5 00                         db 0
000012F6 0000                       dw 0
```

```
                                    ; 09/01/2024
                                    ; PCDOS 7.1 IBMDOS.COM - DOSDATA:12F8h

                                    VxDpath:
000012F8 633A5C77696E613230-            db 'c:\wina20.386',0
00001301 2E33383600
00001306 0000                           dw 0

                                    ; 09/01/2024
                                    ; PCDOS 7.1 IBMDOS.COM - DOSDATA:1308h

                                    drive_flags:
00001308 00<rep 1Ah>                    times 26 db 0

                                    DriverLoad:
00001322 01                             db 1
                                    BiosDataPtr:        ; 08/03/2024
00001323 0000<rep 2h>                   times 2 dw 0
00001327 00<rep 5h>                     times 5 db 0
0000132C 0400                           dw 4
0000132E [B812]                         dw INDOS_FLAG
00001330 [0813]                         dw drive_flags
00001332 [3613]                         dw NLS_YES      ; "YN"
00001334 [3A13]                         dw unknown_zero_dd
                                    NLS_YES:
00001336 59                             db 'Y'
00001337 4E                         NLS_NO:     db 'N'
                                    NLS_yes2:
00001338 79                             db 'y'
                                    NLS_no2:
00001339 6E                             db 'n'

                                    unknown_zero_dd:
0000133A 00000000                       dd 0

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

                                    ; 27/04/2019 - Retro DOS v4.0

                                    ;include msdos.cl2                  ; XMMERRMSG

                                    ; DOSDATA:12B8h (MSDOS 6.21, MSDOS.SYS)

                                    ;XMMERRMSG:
                                    ;   db      0Dh,0Ah
                                    ;   db      'A20 Hardware Error',0Dh,0Ah,'$'

                                    ; DOSDATA ends

                                    ; 05/11/2022
                                    ;---------------------------------------------------------------------------
                                    ; End of MSDOS 5.0 MSDOS.SYS /// Retro DOS v4.0 (2022) - 05/11/2022
                                    ;---------------------------------------------------------------------------

                                    ; 28/12/2022 - Retro DOS v4.1
                                    ; (windows 3.1 and Rational Extender patches are removed/disabled)
                                    ; (Windows 3.1 does not use the patches below if DOS version is MSDOS 5.0)
                                    ;---------------------------------------------------------------------------
                                    %if 0

                                    ;---------------------------------------------------------------------------
                                    ; 05/12/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)

                                    ;===========================================================================
                                    ; WPATCH.INC (MSDOS 6.0, 1991)  ;;; Windows 3.1 patches ;;;
                                    ;===========================================================================
                                    ; 27/04/2019 - Retro DOS 4.0

                                    ;DOSDATA Segment

                                    ; DOSDATA:12CFh (MSDOS 6.21, MSDOS.SYS)

                                    ; Retro DOS v5.0
                                    ; PCDOS 7.1 IBMDOS.COM - DOSDATA:1180h

                                    ; 05/11/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
                                    ; DOSDATA:12CFh (MSDOS 5.0, MSDOS.SYS)

                                    ; first and second DOS patches
                                    ;   Non-console device read/write (system calls 3Fh and 40h)
                                    ;
                                    ; Code in disk.asm, 2 locations, one for read, one for write
                                    ;   DVRDLP:
                                    ;   DVWRTLP:
                                    ;
                                    ;
                                    ; 036h     lds     si,SS:[????]                        ; ThisSFT
                                    ;   lds     si,si+7                                 ; sf_devptr
                                    ; 0E8h     call    ????            <- "simulate" int28 event ; DSKSTATCHK

                                    DOSP1_ID:   db      036h,0C5h,036h
                                    DOSP1_THISSFT:      db      036h,005h,0C5h,074h,007h,0E8h
                                    DOSP1_ID_LEN        equ     $-DOSP1_ID

                                                db      90h, 90h

                                    DOSP12_ID:  db      036h,0C5h,036h
                                    DOSP12_THISSFT:     db      036h,005h,0C5h,074h,007h,0E8h
                                    DOSP12_ID_LEN       equ     $-DOSP1_ID

                                    ; DOSDATA:12E3h

                                    ; Third/Fourth DOS patch - System call 3Fh (Read) from console
                                    ;
                                    ; Code in disk.asm, 1 location
                                    ;   GETBUF:
                                    ;
                                    ; 051h     push    cx      <- begin special int28 mode
                                    ;   push    es
                                    ;   push    di
                                    ;   mov     dx,???? ; offset dosgroup:CONBUF
                                    ;   call    ????    ; $STD_CON_STRING_INPUT
                                    ;   pop     di
                                    ;   pop     es
                                    ; 059h     pop     cx      <- end special int28 mode

                                    DOSP3_ID:   db      051h,006h,057h,0BAh
                                    DOSP3_CONBUF:       db      029h,002h,0E8h
                                    DOSP3_ID_LEN        equ     $-DOSP3_ID
                                                db      09Ah,0E3h,05Fh,007h   ; ???? , pop di, pop es
                                    DOSP4_ID:   db      059h                  ; pop cx
                                    DOSP4_ID_OFF        equ     (DOSP4_ID - DOSP3_ID)

                                    ; DOSDATA:12EFh
```

```
55522
55523                           ; Fifth DOS patch - System call 40h (Write) to console
55524                           ;
55525                           ; Code in disk.asm, 1 location
55526                           ;
55527                           ;          push   cx
55528                           ;      WRCONLP: lodsb
55529                           ;          cmp    al,1Ah
55530                           ;          jz     ????
55531                           ;          call   ????   <- "simulate" int28 event
55532                           ;          loop   WRCONLP
55533                           ;      CONEOF:    pop    ax
55534
55535                           DOSP5_ID:   db     051h                 ; push cx
55536                                       db     0ACh,03Ch,01Ah,074h,005h
55537                                       db     0E8h                 ; call
55538                           DOSP5_ID_LEN        equ    $-DOSP5_ID
55539
55540                           ; DOSDATA:12F6h
55541
55542                           ; Seventh DOS patch - System call entry, patch USER_ID with VMid for share
55543                           ;
55544                           ; Code in disp.asm, 1 location
55545                           ;
55546                           ;
55547                           ;   mov [SaveDS],ds
55548                           ;   mov [SaveBX],bx
55549                           ;   mov bx,cs
55550                           ;   mov ds,bx
55551                           ;   inc [indos]
55552                           ;   xor ax,ax
55553                           ;   mov [USER_ID],AX      <- Patch to set USER_ID to VMID
55554
55555                           DOSP7_ID:   db     02Eh,08Ch,01Eh
55556                           DOSP7_SAVEDS:       db     07Eh,05h              ; mov [SaveDS],ds
55557                                       db     02Eh,089h,01Eh
55558                           DOSP7_SAVEBX:       db     07Ch,05h              ; mov [SaveBX],bx
55559                                       db     08Ch,0CBh             ; mov bx,cs
55560                                       db     08Eh,0DBh             ; mov ds,bx
55561                                       db     0FEh,006h
55562                           DOSP7_INDOS:        db     0CFh,002h             ; inc [indos]
55563                                       db     033h,0C0h             ; xor ax,ax
55564                           DOSP7_ID_LEN        equ    $-DOSP7_ID
55565
55566                           ; DOSDATA:130Ah
55567
55568                           ; Eighth DOS patch - OWNER check in handle calls. For share, need to NOP test
55569                           ;
55570                           ; Code in handle.asm, 1 location in routine CheckOwner
55571                           ;
55572                           ;
55573                           ;
55574                           ;   push   ax
55575                           ;   mov    ax,ss:[USER_ID]     <- patch to XOR AX,AX to set zero
55576                           ;   cmp    ax,es:[di.sf_UID]   <- NOP
55577                           ;   pop    ax
55578                           ;   jz     ????
55579
55580                           DOSP8_ID:   db     050h                 ; push ax
55581                                       db     036h,0A1h
55582                           DOSP8_USER_ID:      db     0EAh,002h             ; mov  ax,ss:[USER_ID]
55583                                       db     026h,03Bh,045h        ; cmp  ax,es:[di+2F]
55584                           DOSP8_ID_LEN        equ    $-DOSP8_ID
55585                                       db     02Fh,058h             ; pop  ax
55586
55587                           ; DOSDATA:1314h
55588
55589                           ; 10th, 11th, 12th DOS patch - System call 3Fh (Read) in raw mode
55590                           ;
55591                           ;   Take RAW read to STDIN SFT and turn it into a polling loop doing
55592                           ;   a yeild when a character is not ready to be read.
55593                           ;
55594                           ; Code in disk.asm, 3 locations
55595                           ;
55596                           ;   DVRDRAW:
55597                           ;       PUSH    ES
55598                           ;       POP     DS
55599                           ;   ReadRawRetry:                        <- Patch 10
55600                           ;       MOV     BX,DI
55601                           ;       XOR     AX,AX                   <- Reenter #2
55602                           ;       MOV     DX,AX
55603                           ;       call    SETREAD
55604                           ;       PUSH    DS                      <- Reenter #1
55605                           ;       LDS     SI,[THISSFT]
55606                           ;       call    DEVIOCALL
55607                           ;       MOV     DX,DI
55608                           ;       MOV     AH,86H
55609                           ;       MOV     DI,[DEVCALL.REQSTAT]
55610                           ;       TEST    DI,STERR
55611                           ;       JZ     CRDROK
55612                           ;       call    CHARHARD
55613                           ;       MOV     DI,DX
55614                           ;       OR     AL,AL
55615                           ;       JZ     CRDROK
55616                           ;       CMP     AL,3
55617                           ;       JZ     CRDFERR
55618                           ;       POP     DS
55619                           ;       JMP     ReadRawRetry
55620                           ;
55621                           ;   CRDFERR:
55622                           ;       POP     DI                      <- Patch 11
55623                           ;   DEVIOFERR:
55624                           ;       LES     DI,[THISSFT]
55625                           ;       jmp     SET_ACC_ERR_DS
55626                           ;
55627                           ;   CRDROK:
55628                           ;       POP     DI                      <- Patch 12
55629                           ;       MOV     DI,DX
55630                           ;       ADD     DI,[CALLSCNT]
55631                           ;       JMP     SHORT ENDRDDEVJ3
55632
55633                           DOSP10_ID:          db     006H,01FH
55634                           DOSP10_LOC_OFFSET   equ    $-DOSP10_ID
55635                           DOSP10_LOC:         db     08BH,0DFH
55636                           DOSP10_REENT2_OFFSET        equ    $-DOSP10_LOC
55637                                               db     033H,0C0H,08BH,0D0H,0E8H
55638                           DOSP10_ID_LEN               equ    $-DOSP10_ID
55639                                               db     0DFH,00EH
55640                           DOSP10_REENT1_OFFSET        equ    $-DOSP10_LOC
55641                                               db     01EH,036H,0C5H,036H,036H,005H,0E8H,0AFH,00EH
55642                                               db     08BH,0D7H,0B4H,086H,036H,08BH,03EH
55643                           DOSP10_PACKVAL_OFFSET       equ    $-DOSP10_ID
55644                                               db     009H,003H
55645                                               db     0F7H,0C7H,000H,080H,074H,019H,0E8H,047H,017H
```

```
                                db      08BH,0FAH,00AH,0C0H,074H,010H,03CH,003H,074H,003H
                                db      01FH,0EBH,0CFH
DOSP11_LOC_OFFSET       equ     $-DOSP10_ID
                                db      05FH
DOSP11_REENT_OFFSET             equ     $-DOSP10_LOC
                                db      036H,0C4H,03EH,036H,005H,0E9H,0A1H,004H

DOSP12_LOC_OFFSET       equ     $-DOSP10_ID
                                db      05FH,08BH,0FAH
; DOSDATA:1353h

; 13th DOS patch - Actually a SYSINIT patch. Patches the stack fault code
;          which prints the fatal stack fault error on DOS >= 3.20.
;
;       Sets focus to current VM so user can see fatal message.
;
;
;   l0: lodsb               <- Setfocus here
;       cmp al, '$'
;       je l1
;       mov bl, 7
;       mov ah, 0Eh
;       int 10h
;       jmp l0
;   l1: jmp $

DOSP13_ID:  db      0ACh                    ; l0: lodsb
            db      03Ch,024h               ;     cmp al, '$'
            db      074h,008h               ;     je l1
            db      0B3h,007h               ;     mov bl, 7
            db      0B4h,00Eh               ;     mov ah, 0Eh
            db      0CDh,010h               ;     int 10h
            db      0EBh,0F3h               ;     jmp l0
            db      0EBh,0FEh               ; l1: jmp $
DOSP13_ID_LEN       equ     $-DOSP13_ID

; 05/11/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
; DOSDATA:1362h (MSDOS 5.0 MSDOS.SYS)

; 06/12/2022
;DOSDATASIZE        equ  $ - DOSDATASTART ; 4962 bytes (1362h)

; DOSDATA ends

; 05/01/2024
%endif      ; 05/11/2022

;==============================================================================
; MPATCH.ASM (MSDOS 6.0, 1993)
;==============================================================================
; 27/04/2019 - Retro DOS 4.0

;mpatch.asm -- holds data patch location for callouts
; -- allocate cluster in rom.asm
;
; This area is pointed to by OffsetMagicPatch[609h] in fixed DOS data.
; Currently, this location is used only by magicdrv.sys's patch to
; cluster allocation, however it can be expanded to be used by other
; patches. This is important since we have an easy-access pointer to
; this location in OffsetMagicPatch. Magicdrv.sys is guaranteed to
; only patch out a far call/retf, so any space after that could be
; used as a patch by using OffsetMagicPatch+6. See rom.asm on how
; to call out here.
;
; Currently, we allocate only the minimum space required for the 6
; byte magicdrv patch, so if you change the dos data, you may want
; to reserve space here if your new data will be position dependent
; and would prohibit growing of this table.
;
;history    -       created 8-7-92 by scottq
;           -       added Rational386PatchPtr 2-1-93 by jimmat
;
;Exported Functions
;==================
;MagicPatch     -       callout patched by magidrv.sys for cluster allocations

; DosData Segment

; DOSDATA:1362h (MSDOS 6.21, MSDOS.SYS)

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

; Rational386PatchPtr points to either a RET instruction (80286 or less) or
; a routine to fix buggy versions of the Rational DOS Extender (80386 or
; greater). Added to this file because it needed to be somewhere and is
; 'patch' related.

Rational386PatchPtr:
0000133E 0000       dw      0       ; points to patch routine or RET instr.
; ---------------------------------------------------------------------------

; 25/03/2024
; PCDOS 7.1 IBMDOS.COM - DOSDATA:1340h
; (Windows ME IO.SYS - DOSDATA:133Eh)

MagicPatch:
;MagicPatch proc far
00001340 CB             retf            ;default is to just return to allocate
00001341 90             nop             ;however, this code will be patched
00001342 90             nop             ;by magicdrv.sys to
00001343 90             nop             ; call far ?:?
00001344 90             nop             ; retf or perhaps just jmp far
00001345 90             nop             ;retf/nop take one byte, so we need six instructions
                                        ;for 6 byte patch
;MagicPatch endp

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

;DosData Ends

; MSDOS 5.0-6.22 MSDOS.SYS - DOSDATA:136Ah
;
; 25/03/2024 - Retro DOS v5.0
; PCDOS 7.1 IBMDOS.COM - DOSDATA:1346h

;; Windows ME IO.SYS
;; db 12 dup(0)

; (Windows ME IO.SYS - DOSDATA:1344h)

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

;DOSDATALAST SEGMENT
```

```
55770                                   ; 29/04/2019 - Retro DOS v4.0
55771                                   
55772                                   ;--------------------------------------------------------------------------
55773                                   ; 25/05/2019 - Retro DOS v4.0 Modification (paragraph alignment)
55774                                   
55775                                   ;db 0,1,12,64,19,0 ; ! Magic numbers !
55776                                   
55777                                   ;align 16
55778                                   
55779                                   ; !!! DOSDATA:1370h ; Retro DOS v4.0 only!
55780                                   
55781                                   ;--------------------------------------------------------------------------
55782                                   
55783                                   ; 05/01/2024
55784                                   ;%endif    ; 05/11/2022
55785                                   
55786                                   ; 05/12/2022
55787                                   ;MSDAT001E: ; label byte
55788                                   
55789                                   ; 22/03/2024 - Retro DOS v5.0 (Modified PCDOS 7.1 IBMDOS.COM)
55790                                   ; 05/12/2022 - Retro DOS v4.0 (Modified MSDOS 5.0 MSDOS.SYS)
55791                                   DOSDATAEND equ $
55792                                   DOSDATASIZE equ DOSDATAEND - DOSDATASTART ; = 4962 for MSDOS 5.0 MSDOS.SYS
55793                                                             ; = 4970 for MSDOS 6.22 MSDOS.SYS
55794                                   MSDAT001E equ DOSDATAEND - DOSDATASTART ; = 4934 for PCDOS 7.1 IBMDOS.COM
55795                                                           ; (= 4944 for Windows ME IO.SYS)
55796                                   ;DOSDATALAST ENDS
55797                                   
55798                                   ; Retro DOS v4.0 by Erdogan Tan (Redevelopment of MSDOS 5.0 KERNEL via NASM)
55799                                   ; DECEMBER 2022, ISTANBUL - TURKIYE.
55800                                   ;==========================================================================
55801                                   ;   END
55802                                   ;==========================================================================
55803                                   ; Retro DOS v4.0 by Erdogan Tan (Redevelopment of MSDOS 6.21 KERNEL via NASM)
55804                                   ; ---------------------------
55805                                   ; MAY 2019, ISTANBUL - TURKIYE.
55806                                   ;==========================================================================
55807                                   ; 25/03/2024 - RETRO DOS v5.0 KERNEL code is READY! (Start: 01/01/2024)
```