```
                        Title D-DOS Simple Disk Operating System
                        Subttl Version 1.1




                              .Comment  *

                        Program D-DOS
                        =============

                        A simple DOS for use with Nascom 1 or 2 fitte
                        with  the  Henelec  FDC  card and software, t
                        allow  simple  disk  read,  write  and  forma
                        operations under NAS-SYS control.

                        Four commands are provided:
                        R <aaaa> <nn> <tt> <ss> <dd>
                               Read data starting at  address  <aaaa
                               for  <nn>  sectors,  starting  at <tt
                               track, <ss> sector from drive <dd>.
                        W <ssss> <eeee-1> <tt> <ss> <dd>
                               Write data starting at address  <ssss
                               to  address <eeee-1> to disk, startin
                               <tt> track, <ss> sector on drive <dd>
                        F      Format disk.
                        N      Return to NAS-SYS.




                        D. R. Hunt        Version 1.1        15.11.8
                        Copyright (C)     D. R. Hunt 1980.
```

```
0000'                         .Z80
0000'                         ASEG

                              ; NAS-SYS monitor calls
0028                          PRS     EQU 28H       ; Print following string
0030                          ROUT    EQU 30H       ; Print character in A
005B                          MRET    EQU 5BH       ; Return to NAS-SYS
0060                          ARGS    EQU 60H       ; Load ARGS to registers
0063                          INLIN   EQU 63H       ; Get an input line
0066                          TBCD3   EQU 66H       ; Print contents of HL
0068                          B2HEX   EQU 68H       ; Print contents of A
0069                          SPACE   EQU 69H       ; Print a space
006A                          CRLF    EQU 6AH       ; Print a new line
006B                          ERRM    EQU 6BH       ; Print Error message
0079                          RLIN    EQU 79H       ; Analyse an input line
007B                          BLINK   EQU 7BH       ; Output blinking cursor

                              ; NAS-SYS variables
000D                          CR      EQU 0DH       ; Code for CRLF
0C0B                          ARGN    EQU 0C0BH     ; NAS-SYS input args

                              .Comment  *
                              FDC set up parameters. The first 10H bytes  of
                              the  FDC  software  consists  of  a  table  of
                              parameters which give the configuration of the
                              system.                                         *
B000                          RSTART  EQU 0B000H    ; Start of FDC software
B000     c3, 00, B4           STJMP   EQU RSTART    ; Jump to control
B003     0B, 0C               WKSPC   EQU RSTART+3H ; Workspace location
B005     00, 10               ISTACK  EQU RSTART+5H ; Initial stack location
B007     00, 10               BOOTST  EQU RSTART+7H ; Bootstrap location
B009     00, 10               FMTBUF  EQU RSTART+9H ; Format buffer location
B00B     03,                  DRIVES  EQU RSTART+0BH ; Max. drives in system
B00C     23,                  TRACKS  EQU RSTART+0CH ; Max. tracks in system
B00D     05,                  NTRY    EQU RSTART+0DH ; No. of retries allowed
B00E     F0                   TDEL    EQU RSTART+0EH ; Time delay constant
B00F     01                   DBLS    EQU RSTART+0FH ; Sides flag

                              .Comment  *
                              This is followed by a jump  table  of  entries
                              into the FDC software.                          *
B010     c3, 70, B2           $READ   EQU RSTART+10H ; Read a sector
B013     c3, A5, B2           $WRITE  EQU RSTART+13H ; Write a sector
B016     c3, 31, B0           $INIT   EQU RSTART+16H ; Initialize FDC
B019     c3, DA, B2           $FORMAT EQU RSTART+19H ; Internal format
B01C     c3, A1, B3           $WRBOOT EQU RSTART+1CH ; Warm boot
B01F     c3, 73, B1           $SKTRK  EQU RSTART+1FH ; Seek specified track
B022     c3, 04, B2           $RDENTR EQU RSTART÷22H ; Read entire track
B025     c3, 39, B2           $WENTR  EQU RSTART+25H ; Write entire track
B028     c3, 24, B1           $DRSEL  EQU RSTART+28H ; Drive select
B02B     c3, 07, B1           $LDDRS  EQU RSTART+2BH ; Test side select
```

```
         B02E   C3, bA, B0      $LDCMD  EQU RSTART+2EH  ; Send a cmd. to FC

                                .Comment  *
                                D-DOS  workspaces  are  calculated  from
                                contents  of  WKSPC  in  the FDC software.
                                locations are as follows:
                                WKSPC   = 0C0BH          Origin of workspace
                                TADDR   = WKSPC+11       Address to read/wri
                                UNIT    = WKSPC+13       Current drive numbe
                                SCTR    = WKSPC+14       Current sector numb
                                TRK     = WKSPC+15       Current track numbe
                                NREC    = WKSPC+16       Current No. of sect
                                *

                                ; FDC variables
         0012                   MAXSCT  EQU 18           ; Maximum sectors
         005B                   STEPIN  EQU 5BH          ; Step in command
         00F4                   WRTRK   EQU 0F4H         ; Write track comm
         0005                   CPORT   EQU 5            ; Control port

                                ; Macro expression used to evaluate SCAL
                                SCAL    MACRO X
           0000                         RST 18H
           FD60                         DEFB X
                                        ENDM

                                        ORG 100H
                                        .PHASE 0B400H    ; Start of D-DOS

                                .Comment  *
                                Table of jumps to locations  within  D-DOS
                                allow entry to the READ and WRITE routines
                                use when called from within other programs.
                                $START  Starts D-DOS in the normal mode.
          [illegible] 0  |      $INITD  Initializes the FDC, homes the head
                                        and selects the default drive.
                                $SAVE   Write from the data supplied by A
         10D6                     ARG1 (0C0CH)   =  Start address
                                  ARG2 (0C0EH)   =  End address - 1
                                  ARG3 (0C10H)   =  Track number  (0 - 45H)
                                  ARG4 (0C12H)   =  Sector number (1 - 12H)
                                  ARG5 (0C14H)   =  Drive number  (0 - 2H)
                                $READ   Read from data supplied by ARGS
                                  ARG1           =  Start address
                                  ARG2           =  Number of sectors to loa
                                  ARG3           =  Track number
                                  ARG4           =  Sector number
                                  ARG5           =  Drive number

         B400     C3 B40C       $START: JP START    | Move to B400 & delete
         B403     C3 B016       $INITD: JP $INIT    |  JP B400
         B406     C3 B51C       $SAVE:  JP SAVE     |
         B409     C3 B492       $LOAD:  JP LOAD     |
                          ; Initialize NAS-SYS and disks
B40C    ED 7B B005        START:  LD SP,(ISTACK)
B410    EF                        RST PRS
B411    0D                        DEFB CR
B412    44 2D 44 4F               DEFM "D-DOS Vers 1.1"
B416    53 20 56 65
B41A    72 73 20 31
B41E    2E 31
B420    0D 00                     DEFB CR,0
B422    CD B016                   CALL $INIT       ; Start disk system

                          ; Wait for an input
B425    CD B6AA           GET:    CALL CLRARG      ; Clear args to 0
                                  SCAL INLIN       ; Get an input line
B428    DF                        RST 18H
B429    63                        DEFB INLIN
B42A    1A                        LD A,(DE)        ; Get the first char
B42B    FE 20                     CP " "           ; Is a blank or letter ?
B42D    28 F6                     JR Z,GET         ; A blank, try again
B42F    FE 4E                     CP "N"           ; Is it an N ?
B431    20 05                     JR NZ,TRYW       ; No, jump to TRYW
                          ; Yes, so clear down the args
B433    CD B6AA                   CALL CLRARG
                                  SCAL MRET        ; Now  return to NAS-SYS
B436    DF                        RST 18H
B437    5B                        DEFB MRET
B438    FE 57             TRYW:   CP "W"           ; Is it a W ?
B43A    28 78                     JR Z,DWR         ; Yes, jump to DWR
B43C    FE 52                     CP "R"           ; Is it an R ?
B43E    28 09                     JR Z,DRD         ; Yes, jump to DRD
B440    FE 46                     CP "F"           ; Is it an F ?
B442    CA B573                   JP Z,FORMAT      ; Yes, jump to FORMAT

B445                      ERRTN:  SCAL ERRM        ; Not command, so error
B445    DF                        RST 18H
B446    6B                        DEFB ERRM
B447    18 DC                     JR GET           ; Go back to start

                          ; Read command
B449    13                DRD:    INC DE           ; See if args ready
                                  SCAL RLIN        ; Analyse input args
B44A    DF                        RST 18H
B44B    79                        DEFB RLIN
B44C    38 0A                     JR C,DRDPT1      ; Wrong, so prompt
B44E    3A 0C0B                   LD A,(ARGN)      ; Test for 5 args
B451    FE 05                     CP 5
B453    28 2A                     JR Z,DRD1        ; Ok, so verify
B455    B7                        OR A             ; Test for no args
B456    28 08                     JR Z,DRDPT2      ; No args, so prompt

                          ;In error, or no args, put up prompts
B458    EF                DRDPT1: RST PRS          ; Put up error message
```

```
        B459    45 72 72 6F              DEFM "Error "
        B45D    72 20
        B45F    00                       DEFB 0
        B460    EF              DRDPT2:  RST PRS          ; Put up prompt
        B461    28 54 6F 20              DEFM "(To Sctrs "
        B465    53 63 74 72
        B469    73 20
        B46B    00                       DEFB 0
        B46C    CD B69E                  CALL TSDMSG      ; Print Trk & Sctr msg
                                         SCAL CRLF        ; CR to next line
        B46F    DF                       RST 18H
        B470    6A                       DEFB CRLF

                                ; Now get the input lines after prompts
                                         SCAL INLIN
        B471    DF                       RST 18H
        B472    63                       DEFB INLIN
                                         SCAL RLIN
        B473    DF                       RST 18H
        B474    79                       DEFB RLIN
        B475    38 6A                    JR C,DWR1        ; On error jump to DW

                                ; Now validate these inputs
        B477    21 0C0B                  LD HL,ARGN       ; Point to ARGN
        B47A    3E 05                    LD A,5           ; Set number of args
        B47C    BE                       CP (HL)          ; Test for equality
        B47D    20 62                    JR NZ,DWR1       ; Idiot boobed again
                                                          ; Tell him to try aga

                                ; Test that tracks and sectors are ok
                                ; Re-entry point if args are ok
        B47F    21 0C10         DRD1:    LD HL,ARGN+5
        B482    CD B65B                  CALL DTEST1
        B485    38 5A                    JR C,DWR1        ; On error jump to DW
                                ; Test drive number ok
        B487    CD B676                  CALL DTEST3
        B48A    38 55                    JR C,DWR1        ; On error jump to DW
                                ; All ready, so call LOAD
        B48C    CD B492                  CALL LOAD
        B48F    C3 B425                  JP GET
---------------------------------------------------------------------------
                                ; All args are valid so save them
        B492    CD B6B5         LOAD:    CALL WSPC        ; Find work space
        B495    23                       INC HL           ; Point to NRECS
        B496    23                       INC HL
        B497    23                       INC HL
        B498    23                       INC HL
        B499    23                       INC HL
        B49A    3A 0C0E                  LD A,(ARGN+3)    ; Get number of recs
        B49D    77                       LD (HL),A        ; Save in (NRECS)
        B49E    CD B67F                  CALL DATSV1
        B4A1    CD B68A                  CALL DATSV2
```

```
                        ; Load data from the disk, and test the Carry
                        ; flag for a good load
B4A4    CD B5B3                 CALL LOADER
B4A7    D8                      RET C           ; Bad load, start again
                        ; Tell 'em its ok
B4A8    EF                      RST PRS
B4A9    43 6F 6D 70             DEFM "Complete"   [illegible]
B4AD    6C 65 74 65
B4B1    0D 00                   DEFB CR,0
B4B3    C9                      RET             ; Go back to start

                        ; Write command
B4B4    13              DWR:    INC DE          ; See if args are ready
                                SCAL RLIN       ; Analyse input line
B4B5    DF                      RST 18H
B4B6    79                      DEFB RLIN
B4B7    38 0A                   JR C,DWRPT1     ; Wrong, so prompt
B4B9    3A 0C0B                 LD A,(ARGN)     ; Test for 5 args
B4BC    FE 05                   CP 5
B4BE    28 42                   JR Z,DWR3       ; Ok so jump to DWR3
B4C0    B7                      OR A            ; Test for no args
B4C1    28 08                   JR Z,DWRPT2     ; No args, so prompt

                        ; On error, or no args, put out prompt
34C3    EF              DWRPT1: RST PRS         ; Put up error message
34C4    45 72 72 6F             DEFM "Error "
34C8    72 20
34CA    00                      DEFB 0
34CB    EF              DWRFT2: RST PRS         ; Put up promt string
34CC    28 46 72 6F             DEFM "(From To "
34D0    6D 20 54 6F
34D4    20
 4D5    00                      DEFB 0
 4D6    CD B69E                 CALL TSDMSG     ; Print Trk & Sctr msg
                                SCAL CRLF       ; CR to next line
 4D9    DF                      RST 18H
34DA    6A                      DEFB CRLF

                        ; Get new input line
                                SCAL INLIN
34DB    DF                      RST 18H
34DC    63                      DEFB INLIN
                                SCAL RLIN
34DD    DF                      RST 18H
34DE    79                      DEFB RLIN
34DF    30 19                   JR NC,DWR2      ; Ok so jump to DWR2

                        ; Idiot boobed on second input !! Start again.
34E1    EF              DWR1:   RST PRS
34E2    45 72 72 6F             DEFM "Error. Start again."
34E6    72 2E 20 53
34EA    74 61 72 74
34EE    20 61 67 61
```

```
B4F2    69 6E 2E
B4F5    0D 00                    DEFB CR,0
B4F7    C3 B425                  JP GET              ; Go back to start

                        ; Validate the number of args input
B4FA    21 0C0B         DWR2:    LD HL,ARGN          ; Point to ARGN
B4FD    3E 05                    LD A, 5
B4FF    BE                       CP (HL)             ; Test for equality
B500    38 DF                    JR C,DWR1           ; Wrong, start again

                        ; Test that ARG2 > ARG1
B502                    DWR3:    SCAL ARGS
B502    DF                       RST 18H
B503    60                       DEFB ARGS
B504    EB                       EX DE,HL
B505    ED 52                    SBC HL,DE
B507    38 D8                    JR C,DWR1           ; Wrong, start again

                        ; Test that track and sector are valid
B509    21 0C10                  LD HL,ARGN+5
B50C    CD B65B                  CALL DTEST1
B50F    38 D0                    JR C,DWR1           ; Wrong, start again
                        ; Test that drive number is valid
B511    CD B676                  CALL DTEST3
B514    38 CB                    JR C,DWR1           ; Wrong, start again
                        ; All ready so call SAVE
B516    CD B51C                  CALL SAVE
B519    C3 B425                  JP GET
------------------------------------------------------------------------
                        ; All are valid, so save
B51C    CD B67F         SAVE:    CALL DATSV1
B51F    CD B68A                  CALL DATSV2

                        ; Calculate the number of sectors to be sav
                                 SCAL ARGS           ; Get start and end
B522    DF                       RST 18H
B523    60                       DEFB ARGS
B524    EB                       EX DE,HL
B525    37                       SCF                 ; Make answer one l
B526    ED 52                    SBC HL,DE           ; Gives the length
                        ; Now reduce by modulo 128 to count the sec
                        ; Sector count in DE
B528    11 0000                  LD DE,0
B52B    01 0080                  LD BC,128
B52E    ED 42           MDLOOP:  SBC HL,BC           ; Reduce HL by 128
B530    13                       INC DE              ; Count 1 in DE
B531    30 FB                    JR NC,MDLOOP        ; Not negative, go
                        ; Save sectors count in NREC
B533    CD B6B5                  CALL WSPC           ; Find workspace
B536    23                       INC HL              ; Point to NREC
B537    23                       INC HL           after SBC  HL DE
B538    23                       INC HL
B539    23                       INC HL
```

```
B53A    23                     INC HL
B53B    73                     LD (HL),E        ; Save in NREC
                        ; Save the number of sectors for later
B53C    D5                     PUSH DE

                        ; Put the data on the disk, test Carry flag for
                        ; a good write.
B53D    CD B5BD                CALL SAVER
B540    D1                     POP DE           ; Get sectors count back
B541    D8                     RET C            ; Error, start again ?

                        ; Put out the number of sectors saved
                        ; and start address of the next file.
B542    EF                     RST PRS
B543    53 65 63 74            DEFM "Sectors saved "
B547    6F 72 73 20
B54B    73 61 76 65
B5 F    64 20
B  1    00                     DEFB 0
B552    7B                     LD A,E           ; Put sectors in A
                               SCAL B2HEX       ; Print contents of A
B553    DF                     RST 18H
B554    68                     DEFB B2HEX
                               SCAL CRLF        ; Print a CR
B555    DF                     RST 18H
B556    6A                     DEFB CRLF
B557    CD B624                CALL SECINC      ; Inc. to next sector
B55A    EF                     RST PRS
B55B    4E 65 78 74            DEFM "Next file at ("
B55F    20 66 69 6C
B563    65 20 61 74
B567    20 28
B569    00                     DEFB 0
B56A    CD B69E                CALL TSDMSG      ; Print Trk & Sctr msg
                               SCAL SPACE
 55D    DF                     RST 18H
B. E    69                     DEFB SPACE
B56F    CD B603                CALL PUTDAT      ; Print track & sector
B572    C9                     RET              ; Go back to start

                        ; Format command
B573    EF              FORMAT: RST PRS         ; Print warning message
B574    46 6F 72 6D            DEFM "Format wipes disk. Ok ? "
B578    61 74 20 77
B57C    69 70 65 73
B580    20 64 69 73
B584    6B 2E 20 4F
B588    6B 20 3F 20
B58C    00                     DEFB 0
                        ; Get an input and print it
                               SCAL BLINK
B58D    DF                     RST 18H
B58E    7B                     DEFB BLINK
```

```
B58F    F7                      RST ROUT
                        ; Save the input, print a CR, restore the in
B590    F5                      PUSH AF
                                SCAL CRLF
B591    DF                      RST 18H
B592    6A                      DEFB CRLF
B593    F1                      POP AF
                        ; Test it for "Yes"
B594    FE 59                   CP "Y"
B596    C2 B425                 JP NZ,GET          ; Not Y, back to sta

                        ; Ok to format, so call format .
B599    CD B6D2                 CALL FORM

                        ; Test for format error (no error bits in A)
B59C    B7                      OR A
B59D    CA B425                 JP Z,GET           ; If Ok, back to st r
B5A0    EF                      RST PRS            ; Print fail messa
B5A1    46 6F 72 6D             DEFM "Format error."
B5A5    61 74 20 65
B5A9    72 72 6F 72
B5AD    2E
B5AE    0D 00                   DEFB CR,0
B5B0    C3 B425                 JP GET             ; Go back to start
-----------------------------------------------------------------------
                        ; Loader reads the number of 128 byte sectors
                        ; set up in NREC to the address TADDR
B5B3    CD B5C7         LOADER: CALL READ          ; Read a sector
B5B6    C0                      RET NZ             ; Return on error
B5B7    CD B624                 CALL SECINC        ; Go for next sector
B5BA    C8                      RET Z              ; Return if end
B5BB    18 F6                   JR LOADER          ; More so round agai

                        ; Saver loads the number of 128 byte sectors
                        ; set up in NREC starting at address TADDR
B5BD    CD B61A         SAVER:  CALL WRITE         ; Write a sector
B5C0    C0                      RET NZ             ; Return on error
B5C1    CD B624                 CALL SECINC        ; Go for next sector
B5C4    C8                      RET Z              ; Return if end
B5C5    18 F6                   JR SAVER           ; More so round agai

                        ; Read a 128 byte sector to address TADDR
                        ; Test for errors and report
B5C7    CD B64A         READ:   CALL SETR          ; Set regs to data
B5CA    CD B010         READER  CALL $READ         ; read the sector
                        ; Test for errors
B5CD    B7                      OR A
B5CE    C8                      RET Z              ; Return if none

                        ; Exit for read write errors and report
B5CF    CD B5D6         ENDRW:  CALL ERMESG        ; Print error messa
B5D2    AF                      XOR A              ; Clear any flags
B5D3    3C                      INC A              ; Set A to error fl
```

```
 B5D4     37                        SCF              ; Set C flag for error
 B5D5     C9                        RET

                          ; Put out error message, error number, track
                          ; and sector details
 B5D6     F5              ERMESG:   PUSH AF          ; Save error code
 B5D7     EF                        RST PRS          ; Print error message
 B5D8     44 69 73 6B               DEFM "Disk error "
 B5DC     20 65 72 72
 B5E0     6F 72 20
 B5E3     00                        DEFB 0
                          ; Put out error number
 B5E4     F1                        POP AF           ; Get error code back
                                    SCAL B2HEX       ; Print contents of A
 B5E5     DF                        RST 18H
 B5E6     68                        DEFB B2HEX
 B5E7     EF                        RST PRS          ; Print location message
 B5E8     20 61 74 20               DEFM " at (Buff "
 B5EC     28 42 75 66
 B5F0     66 20
 B5F2     00                        DEFB 0
 B5F3     CD B69E                   CALL TSDMSG      ; Print Trk & Sctr msg
                                    SCAL SPACE
 B5F6     DF                        RST 18H
 B5F7     69                        DEFB SPACE
                          ; Put out address, track, sector and drive Nos.
 B5F8     CD B6B5                   CALL WSPC        ; Find workspace
 B5FB     5E                        LD E,(HL)        ; Load DE with TADDR
 B5FC     23                        INC HL
 B5FD     56                        LD D,(HL)
 B5FE     EB                        EX DE,HL         ; Swap with HL
                                    SCAL TBCD3       ; Print contents of HL
 B5FF     DF                        RST 18H
 B600     66                        DEFB TBCD3
                                    SCAL SPACE
 B601     DF                        RST 18H
 B602     69                        DEFB SPACE
 B603     CD B6B5         PUTDAT:   CALL WSPC        ; Find workspace
 B606     23                        INC HL           ; Point to TRK
 B607     23                        INC HL
 B608     23                        INC HL
 B609     23                        INC HL
 B60A     06 02                     LD B,2           ; Set to loop twice
 B60C     7E              ERRLP:    LD A,(HL)        ; Get the data
                                    SCAL B2HEX       ; Print contents of A
 B60D     DF                        RST 18H
 B60E     68                        DEFB B2HEX
                                    SCAL SPACE
 B60F     DF                        RST 18H
 B610     69                        DEFB SPACE
 B611     2B                        DEC HL           ; Point to next
 B612     10 F8                     DJNZ ERRLP       ; More ? Round again
                          ; Put out last of data with CR
```

```
        B614    7E                      LD A,(HL)           ; Get the data
                                        SCAL B2HEX          ; Print contents of A
        B615    DF                      RST 18H
        B616    68                      DEFB B2HEX
                                        SCAL CRLF           ; Print a CR
        B617    DF                      RST 18H
        B618    6A                      DEFB CRLF
        B619    C9                      RET

                                ; Write a sector
        B61A    CD B64A         WRITE:  CALL SETR           ; Set the regs to dat
        B61D    CD B013                 CALL $WRITE         ; Write the sector
        B620    B7                      OR A                ; Test for errors
        B621    C8                      RET Z               ; None so return
                                ; On error go and report details
        B622    18 AB                   JR ENDRW

                                ; Increment routine to get the track and sect
        B624    CD B6B5         SECINC: CALL WSPC           ; Find workspace
        B627    5E                      LD E,(HL)
        B628    23                      INC HL
        B629    56                      LD D,(HL)
        B62A    E5                      PUSH HL
        B62B    EB                      EX DE,HL            ; Put (TADDR) into HL
        B62C    11 0080                 LD DE,128
        B62F    19                      ADD HL,DE           ; Point to next 128
        B630    EB                      EX DE,HL            ; Put HL into (TADDR
        B631    E1                      POP HL
        B632    72                      LD (HL),D
        B633    2B                      DEC HL
        B634    73                      LD (HL),E
        B635    23                      INC HL              ; Point to NREC
        B636    23                      INC HL
        B637    23                      INC HL
        B638    23                      INC HL
        B639    23                      INC HL
        B63A    35                      DEC (HL)            ; Reduce by one
        B63B    C8                      RET Z               ; If zero, finish
                                ; More to do, so inc the sector number
        B63C    2B                      DEC HL              ; Point to SCTR
        B63D    2B                      DEC HL
        B63E    34                      INC (HL)            ; Increment by one
                                ; Test for overflow
        B63F    3E 12                   LD A,MAXSCT
        B641    3C                      INC A
        B642    BE                      CP (HL)             ; Test for equality
        B643    C0                      RET NZ              ; Not equal, carry
                                ; Too many sectors, so inc. track, and set
                                ; sector to 1
        B644    36 01                   LD (HL),1           ; Set sector to 1
        B646    23                      INC HL
        B647    34                      INC (HL)            ; increment track
        B648    B7                      OR A                ; Clear any error f
```

```
B649    C9                  RET

                            ; This routine sets the regs to the details of
                            ; the sector to be read or written
                            ;       HL = Address to start at
                            ;       D  = Track number
                            ;       E  = Sector number
                            ;       C  = Drive number
                            ;       B  = Drive side if applicable
                            ;       A  = Returned error code (if any)
B64A    CD B6B5             SETR:   CALL WSPC       ; Find workspace
B64D    5E                  SETREG  LD E,(HL)       ; Get (TADDR) into DE
B64E    23                          INC HL
B64F    56                          LD D,(HL)
B650    D5                          PUSH DE         ; Save (TADDR)
B651    23                          INC HL
B652    7E                          LD A,(HL)       ; Get (UNIT)
B653    17                          RLA             ; Correct side
B654    4F                          LD C,A          ; Save in C
B655    23                          INC HL
B656    5E                          LD E,(HL)       ; Get (SCTR) into E
B657    23                          INC HL
B658    56                          LD D,(HL)       ; Get (TRK) into D
B659    E1                          POP HL          ; Get (TADDR) into HL
B65A    C9                          RET

                            ; Test if track and sector valid
B65B    3A B00C             DTEST1: LD A,(TRACKS)   ; Get max. tracks/side
B65E    5F                          LD E,A
B65F    3A B00F                     LD A,(DBLS)     ; Test for D/S
B662    B7                          OR A
B663    7B                          LD A,E
B664    28 01                       JR Z,DTESTA     ; Not D/S, skip
B666    83                          ADD A,E         ; D/S, double tracks
B667    3D                  DTESTA: DEC A
B668    BE                          CP (HL)         ; Test for < max.
B669    D8                          RET C           ; Return on track fail
B66A    23                          INC HL          ; Point to sectors i/p
B66B    23                          INC HL
B66C    AF                          XOR A           ; Clear A
B66D    BE                          CP (HL)         ; Test for sector 0
B66E    28 04                       JR Z,DTEST2     ; On fail jump to DTEST:
B670    3E 12                       LD A,MAXSCT     ; Test for > max sector:
B672    BE                          CP (HL)
B673    C9                          RET             ; C flag set if failed
B674    37                  DTEST2: SCF             ; Set C for error
B675    C9                          RET

                            ; Test that drive number is valid
B676    23                  DTEST3: INC HL          ; Point to drive i/p
B677    23                          INC HL
B678    3A B00B                     LD A,(DRIVES)   ; Get max drives
B67B    87                          ADD A,A          ← ?? why
```

called from PRD [illegible]
i.e. [illegible] actual drive [illegible]

```
B67C    3D                      DEC A
B67D    BE                      CP (HL)            ; Test for < max.
B67E    C9                      RET                ; C flag set if fail

                        ; Routines to copy ARGS to FDC workspace
B67F    CD B6B5         DATSV1: CALL WSPC          ; Find workspace
B682    ED 5B 0C0C              LD DE,(ARGN+1)     ; Get ARG1
B686    73                      LD (HL),E          ; Put in (TADDR)
B687    23                      INC HL              TRANSFER ARG1 - ARG
B688    72                      LD (HL),D
B689    C9                      RET
B68A    CD B6B5         DATSV2: CALL WSPC          ; Find workspace
B68D    23                      INC HL             ; Point to UNIT
B68E    23                      INC HL                 ARG7
B68F    3A 0C14                 LD A,(ARGN+9)      ; Get drive number
B692    77                      LD (HL),A
B693    23                      INC HL             ; Point to SCTR
B694    3A 0C12                 LD A,(ARGN+7)      ; Get sector numbe
B697    77                      LD (HL),A
B698    23                      INC HL             ; Point to TRK
B699    3A 0C10                 LD A,(ARGN+5)      ; Get track number
B69C    77                      LD (HL),A
B69D    C9                      RET

                        ; This is the Track, Sectors and Drive messag
B69E    EF              TSDMSG: RST PRS            ; Print the message
B69F    54 72 20 53             DEFM "Tr Sc Dr)"
B6A3    63 20 44 72
B6A7    29
B6A8    00                      DEFB 0
B6A9    C9                      RET

                        ; Routine to clear ARGS to 0
B6AA    2A B003         CLRARG: LD HL,(WKSPC)
B6AD    06 15                   LD B,21
B6AF    36 00           CLRLP:  LD (HL),0
B6B1    23                      INC HL
B6B2    10 FB                   DJNZ CLRLP
B6B4    C9                      RET

                        ; Routine to calculate the workspace
B6B5    D5              WSPC:   PUSH DE
B6B6    2A B003                 LD HL,(WKSPC)
B6B9    11 000B                 LD DE,11
B6BC    19                      ADD HL,DE
B6BD    D1                      POP DE
B6BE    C9                      RET

                        ; Format routine using a skewtable
B6BF    01 07 0D        SKWTAB: DEFB 01H,07H,0DH
B6C2    02 08 0E                DEFB 02H,08H,0EH
B6C5    03 09 0F                DEFB 03H,09H,0FH
B6C8    04 0A 10                DEFB 04H,0AH,10H
```

```
 B6CB     05 0B 11               DEFB 05H,0BH,11H
 B6CE     06 0C 12               DEFB 06H,0CH,12H
 B6D1     00                     DEFB 0             ; Table delimiter

 B6D2     CD B016       FORM:    CALL $INIT         ; Home head and init.
 B6D5     16 00                  LD D,0             ; Init, track counter
 B6D7     DD 21 B6BF    TRACKL:  LD IX,SKWTAB       ; Point to table
 B6DB     DD 5E 00               LD E,(IX)          ; Get a sector number

                        ; Set up an image of a track in RAM
 B6DE     2A B009                LD HL,(FMTBUF)     ; Get start of image

                        ; Set 14 bytes to FF
 B6E1     06 0E                  LD B,14
 B6E3     36 FF         FORM1:   LD (HL),0FFH
 B6E5     23                     INC HL
 B6E6     10 FB                  DJNZ FORM1
                        ; Set 6 bytes to 0 for sync
 B  8     06 06         SECTL:   LD B,6
 B6EA     36 00         FORM1A:  LD (HL),0
 B6EC     23                     INC HL
 B6ED     10 FB                  DJNZ FORM1A
                        ; Set ID, track and sector addresses
 B6EF     36 FE                  LD (HL),0FEH       ; ID address mark
 B6F1     23                     INC HL
 B6F2     72                     LD (HL),D          ; Track address
 B6F3     23                     INC HL
 B6F4     36 00                  LD (HL),0          ; Gap
 B6F6     23                     INC HL
 B6F7     73                     LD (HL),E          ; Sector address
 B6F8     23                     INC HL
 B6F9     36 00                  LD (HL),0          ; Gap
 B6FB     23                     INC HL
 B6FC     36 F7                  LD (HL),0F7H       ; ID field CRC
 B6FE     23                     INC HL
                        ; Set 11 bytes to FF
 B  F     06 0B                  LD B,11
 B701     36 ~~00~~ FF     FORM4:   LD (HL),~~0~~ -1
 B703     23                     INC HL
 B704     10 FB                  DJNZ FORM4
                        ; Set 6 bytes to 0 for sync
 B706     06 06                  LD B,6
 B708     36 00         FORM4A:  LD (HL),0
 B70A     23                     INC HL
 B70B     10 FB                  DJNZ FORM4A
                        ; Set data address mark
 B70D     36 FB                  LD (HL),0FBH       ; Data address mark
 B70F     23                     INC HL
                        ; Set 128 bytes to E5 for data field
 B710     06 80                  LD B,128
 B712     36 E5         FORM5:   LD (HL),0E5H
 B714     23                     INC HL
 B715     10 FB                  DJNZ FORM5
```

```
                                ; Set data field CRC
  B717      36 F7                       LD (HL),0F7H
  B719      23                          INC HL
                                ; Set 11 bytes to FF
  B71A      06 0B                       LD B,11
  B71C      36 FF               FORM6:  LD (HL),0FFH
  B71E      23                          INC HL
  B71F      10 FB                       DJNZ FORM6
                                ; Test for last sector (end of track image)
  B721      DD 23                       INC IX          ; Point to next sec
  B723      DD 5E 00                    LD E,(IX)       ; Put sector No. in
  B726      7B                          LD A,E          ; Copy to A
  B727      B7                          OR A            ; Test for 0
  B728      20 BE                       JR NZ,SECTL     ; If not 0, round ag
                                ; Fill end of track gap with FF
  B72A      06 FF                       LD B,255
  B72C      36 FF               FORM7:  LD (HL),0FFH
  B72E      23                          INC HL
  B72F      10 FB                       DJNZ FORM7

                                ; Write the memory image to an entire track
  B731      3A B00F                     LD A,(DBLS)
  B734      F5                  SDLP:   PUSH AF
                                ; Side select
  B735      CD B02B                     CALL $LDDRS
  B738      2A B009                     LD HL,(FMTBUF)
  B73B      3E F4                       LD A,WRTRK      ; WRITE TRACK command
  B73D      CD B025                     CALL $WENTR
  B740      B7                          OR A            ; Test for write err
  B741      C0                          RET NZ          ; Return if error
                                ; Test that a minimum of C08H byte written
  B742      E5                          PUSH HL
  B743      2A B009                     LD HL,(FMTBUF)
  B746      01 0C08                     LD BC,0C08H
  B749      09                          ADD HL,BC
  B74A      4D                          LD C,L
  B74B      44                          LD B,H
  B74C      E1                          POP HL
  B74D      ED 42                       SBC HL,BC
  B74F      38 1B                       JR C,FERR       ; Return if error
                                ; Test if double sided
  B751      F1                          POP AF
  B752      B7                          OR A
  B753      28 03                       JR Z,SDNXT
  B755      3D                          DEC A
  B756      18 DC                       JR SDLP

                                ; Test if this is the last track
  B758      14                  SDNXT:  INC D           ; Inc to next track
  B759      3A B00C                     LD A,(TRACKS)
  B75C      92                          SUB D
  B75D      C8                          RET Z          ; Return if all done
                                ; More to do, so step head in one
```

```
B75E     3E 5B                  LD A,STEPIN
B760     CD B02E                CALL $LDCMD      ; Send command to FDC
                        ; Wait until head settled
B763     DB 05          FORM8:  IN A,(CPORT)     ; Get FDC INTRQ
B765     E6 40                  AND 40H          ; Test it
B767     28 FA                  JR Z,FORM8
B769     C3 B6D7                JP TRACKL        ; Go do the next track

B76C     F1             FERR:   POP AF
                        ; Format error code
B76D     3E FF                  LD A,0FFH
B76F     C9                     RET

                                .DEPHASE

                                END              ; END OF LISTING
```

```
D-DOS Simple Disk Operating System      MACRO-80 3.35          P
Version 1.1

Macros:
SCAL

Symbols:
$DRSEL  B028     $FORMA  B019     $INIT   B016     $INITD  B403
$LDCMD  B02E     $LDDRS  B02B     $LOAD   B409     $RDENT  B022
$READ   B010     $SAVE   B406     $SKTRK  B01F     $START  B400
$WENTR  B025     $WRBOO  B01C     $WRITE  B013     ARGN    0C0B
ARGS    0060     B2HEX   0068     BLINK   007B     BOOTST  B007
CLRARG  B6AA     CLRLP   B6AF     CPORT   0005     CR      000D
CRLF    006A     DATSV1  B67F     DATSV2  B68A     DBLS    B00F
DRD     B449     DRD1    B47F     DRDPT1  B458     DRDPT2  B460
DRIVES  B00B     DTEST1  B65B     DTEST2  B674     DTEST3  B676
DTESTA  B667     DWR     B4B4     DWR1    B4E1     DWR2    B4FA
DWR3    B502     DWRPT1  B4C3     DWRPT2  B4CB     ENDRW   B5CF
ERMESG  B5D6     ERRLP   B60C     ERRM    006B     ERRTN   B449
FERR    B76C     FMTBUF  B009     FORM    B6D2     FORM1   B6E1
FORM1A  B6EA     FORM4   B701     FORM4A  B708     FORM5   B71
FORM6   B71C     FORM7   B72C     FORM8   B763     FORMAT  B5
GET     B425     INLIN   0063     ISTACK  B005     LOAD    B49
LOADER  B5B3     MAXSCT  0012     MDLOOP  B52E     MRET    005B
NTRY    B00D     PRS     0028     PUTDAT  B603     READ    B5C7
RLIN    0079     ROUT    0030     RSTART  B000     SAVE    B51
SAVER   B5BD     SDLP    B734     SDNXT   B758     SECINC  B624
SECTL   B6E8     SETR    B64A     SKWTAB  B6BF     SPACE   0069
START   B40C     STEPIN  005B     STJMP   B000     TBCD3   006
TDEL    B00E     TRACKL  B6D7     TRACKS  B00C     TRYW    B43
TSDMSG  B69E     WKSPC   B003     WRITE   B61A     WRTRK   00F4
WSPC    B6B5


No  Fatal error(s)
```