$99

TRSDOS/LS - DOS 6.x

"THE SOURCE"

```
LD      A,(BUFFER$+1)   ;P/u buffer hi-order addr
LD      D,A
LD      BC,13           ;Move name/ext into dest
LDIR
LD      D,(IY+9)        ;P/u dir cyl of dest
POP     BC              ;Rcvr DEC of source
PUSH    BC
LD      A,B             ;Calc dir sector for
AND     1FH             ;  source SYS module
ADD     A,2
LD      E,A
LD      HL,(BUFFER$)    ;P/u buffer ptr for dest
CALL    WRSYS           ;Write the dir to dest
LD      A,18            ;Init "Dir write error
JP      NZ,EXIT3        ;  and quit on bad write

The HIT entries were transferred prior

POP     BC              ;Rcvr DEC of source
PUSH    BC
LD      A,B             ;Test for SYSØ
CP      2
JP      NZ,DOFILØ       ;Bypass if not SYSØ
CALL    PMTSRC          ;Prompt source
IF      @MOD4
LD      B,16            ;Init to xfer BOOT track
LD      DE,Ø            ;Init track Ø, sector Ø
ENDIF
IF      @MOD2
LD      DE,(PROTSEC)    ;Get sysinfo sector
LD      A,D
OR      A
LD      B,5
```

LOGICAL SYSTEMS INC.

Volume 1

The SYSTEM

"THE SOURCE"

CONTENTS

LOGICAL SYSTEMS INC.

8970 North 55th Street
P.O. Box 23956
Milwaukee, WI 53223

Introduction to Volume One

This is volume one of three in the set of commented source code listings for LS-DOS/TRSDOS 6.2, as assembled for the TRS-8Ø Model 4/4P computer. This volume contains the resident part of the operating system as well as the pieces loaded as overlays. The resident portion of the DOS is contained in the files BOOT/SYS (Lowcore) and SYSØ/SYS (Sysres). The overlays are files SYS1 through SYS13, excluding the three Library modules (SYS6-SYS8).

Each file will be preceded by a brief description of its functions. Cross reference listings for the two resident files will be found immediately following the assembly listings. The overlay files are not cross referenced, but will have a symbol table listing following each assembly listing.

Two EQUate files are generated by the assembly and cross reference of the resident portion of the DOS. These two files are then used by the overlays to reference the main DOS areas. The files, LDOS6Ø/EQU and SYSØ/EQU, will be found in Appendix A in the rear of this volume. The standard system debugger, SYS5, also generates an EQUate file for use by SYS9, the extended debugger. This file is listed at the end of Appendix A.

This book should by no means be considered a tutorial on assembly language or on the workings of the LS-DOS/TRSDOS operating system. It is only the commented source code used to assemble the previously mentioned operating systems. It can be used for reference purposes and to view examples of interfacing outside drivers, filters and other programs to the DOS and to each other. It is not meant to replace the normal technical reference manual available from the computer manufacturer.

This product is sold on an as-is basis, and is totally unsupported by Logical Systems, Inc. No questions regarding any aspect of the source code will be answered by LSI customer or technical support. Support for LS-DOS users is provided through their OEM dealer. Support for TRSDOS 6.x is provided by Tandy Corporation. Comments or suggestions may be sent to Logical Systems, Inc. in care of the Source Code Technical Editor, but correspondence concerning these comments will not be made.

LOWCORE/ASM - Low memory assignments and BIOS

This is the source code that assembles the file BOOT/SYS as a core image (no load information) file. The Lowcore file contains the bootstrap loader, certain low memory storage locations, the I/O drivers, the DCBs and DCTs, memory bank handling, byte I/O routines, and other miscellaneous code. The file normally occupies track Ø on a system disk, and is read into memory by the computer's boot ROM. At that point, execution is transferred to the Lowcore boot loader and Sysres is loaded.

The Lowcore source code is divided into several subsections. In order of appearance, they are:

DCB area - Default DCBs for the standard I/O devices.
BOOT - Boot loader for Sysres and CRTC initialization.
SYSINFO & DCT - The stack area, input buffer, and default DCTs.
IODVR - Byte I/O handling for standard and non-standard devices.
MULDIV - Math routines used by the system.
CLOCKS - Heartbeat processing, vidram and bank switching.
KIDVR - The keyboard driver and type ahead processing.
DODVR - The video driver and VDCTL SVC handler.
PRDVR - The printer driver.
FDCDVR - The floppy disk driver.

A cross reference of all non-local labels follows the assembly listing.

NOTES:

```
                00100 ;LOWCORE/ASM - Low Memory Assignments
0000            00110         TITLE   <LOWCORE - LS-DOS 6.2>
0000            00120 @MOD2   EQU     00              ;Set MOD2 false
FFFF            00130 @MOD4   EQU     -1              ;Set MOD4 true
                00140 ;
                00150 ;       LDOS 6.x Low Core RAM storage assignments
                00160 ;       Copyright (C) 1982 by Logical Systems, Inc.
                00170 ;
                00180 ;       Define switches for international or domestic
                00190 ;
0000            00200 @GERMAN EQU     0
0000            00210 @FRENCH EQU     0
                00220         IF      @GERMAN.AND.@FRENCH
                00230         ERR     'Can''t do both French and German'
                00240         ENDIF
                00250         IF      @GERMAN.OR.@FRENCH
                00260 @INTL   EQU     -1
                00270 @USA    EQU     00
                00280 @HZ50   EQU     -1
                00290         ELSE
0000            00300 @INTL   EQU     00
FFFF            00310 @USA    EQU     -1
0000            00320 @HZ50   EQU     00
                00330         ENDIF
                00340 ;
0000            00350 START$  EQU     0
                00360 ;
                00370 ;       These EQUs are detailed in SYSRES
                00380 ;
000E            00390 FDDINT$ EQU     0EH
001B            00400 PDRV$   EQU     1BH
002B            00410 TIMSL$  EQU     2BH
002C            00420 TIMER$  EQU     2CH
002D            00430 TIME$   EQU     TIMER$+1
0033            00440 DATE$   EQU     33H
003E            00450 INTVC$  EQU     3EH
006A            00460 FLGTAB$ EQU     6AH
006C            00470 CFLAG$  EQU     FLGTAB$+'C'-'A'
006D            00480 DFLAG$  EQU     FLGTAB$+'D'-'A'
0072            00490 IFLAG$  EQU     FLGTAB$+'I'-'A'
0074            00500 KFLAG$  EQU     FLGTAB$+'K'-'A'
0076            00510 MODOUT$ EQU     FLGTAB$+'M'-'A'
0077            00520 NFLAG$  EQU     FLGTAB$+'N'-'A'
0078            00530 OPREG$  EQU     FLGTAB$+'O'-'A'
007B            00540 RFLAG$  EQU     FLGTAB$+'R'-'A'
007C            00550 SFLAG$  EQU     FLGTAB$+'S'-'A'
007F            00560 VFLAG$  EQU     FLGTAB$+'V'-'A'
0089            00570 @KITSK  EQU     FLGTAB$+31
                00580 ;
0200            00590         ORG     200H+START$
                00600 ;
                00610 ;       Page 2 - Device Control Blocks
                00620 ;
0200 00         00630 BUR$    DB      00H             ;Bank use RAM
0201 FE         00640 BAR$    DB      0FEH            ;Bank available RAM
0202 14         00650 LBANK$  DB      20              ;Dir cyl & logical bank
0203 01         00660 JCLCB$  DB      1,0,0           ;Mini-DCB for JCL gets
     00 00
0206 F40F       00670 DVRHI$  DW      DVREND$         ;Start of low I/O zone
0208 05         00680 KIDCB$  DB      5               ;Permit CTL, GET
0209 F008       00690         DW      KIDVR
```

```
020B 00          00700          DB      0,0,0,'KI'
     00 00 4B 49
0210 07          00710 DODCB$   DB      7               ;Permit CTL, PUT, GET
0211 880B        00720          DW      DODVR
0213 00          00730          DB      0,0,0,'DO'
     00 00 44 4F
0218 06          00740 PRDCB$   DB      6               ;Permit CTL, PUT
0219 010E        00750          DW      PRDVR
021B 00          00760          DB      0,0,0,'PR'
     00 00 50 52
0220 15          00770 SIDCB$   DB      15H             ;Routed to *KI
0221 0802        00780          DW      KIDCB$
0223 0D          00790          DB      0DH,0,0,'SI'
     00 00 53 49
0228 17          00800 SODCB$   DB      17H             ;Routed to *DO
0229 1002        00810          DW      DODCB$
022B 0F          00820          DB      0FH,0,0,'SO'
     00 00 53 4F
0230 0A          00830 JLDCB$   DB      0AH,0,0,0AH,0,0,'JL'
     00 00 0A 00 00 4A 4C
0238             00840 S1DCB$   EQU     $               ;1st spare DCB
0031             00850 DCBKL$   EQU     JLDCB$&0FFH+1   ;Non-killable DCB's
                 00860 ;
                 00870 ;        Now load the BOOT loader - part in this page
                 00880 ;
0238             00890 *GET     BOOT4:3
                 00010 ;BOOT4/ASM - LS-DOS 6.2
0238             00020          SUBTTL  '<Bootstrap Loader>'
```

Bootstrap Loader

```
                    00040 *MOD
                    00050 ;
0040                00060 KEYIN   EQU     40H
0066                00070 NMIVECT EQU     66H
021B                00080 DSPLY   EQU     21BH
1200                00090 BUFFER  EQU     1200H
43F6                00100 BOOTBUF EQU     43FFH-9
                    00110 ;
                    00120 ;       Boot loader routine read in by ROM, along with
                    00130 ;        the lowcore I/O drivers.
                    00140 ;       This section loads in SYSRES
                    00150 ;
0238 FD217004       00160 LBOOT   LD      IY,DCT$         ;Set IY for FDCDVR use
023C FD7E09         00170         LD      A,(IY+9)        ;Directory track is
023F FD7705         00180         LD      (IY+5),A        ;  the current track
0242 3E04           00190         LD      A,4
0244 327B00         00200         LD      (FLGTAB$+'R'-'A'),A     ;Set retries
0247 3EC9           00210         LD      A,0C9H
0249 320E00         00220         LD      (FDDINT$),A     ;Return for disk driver
024C 3E12           00230         LD      A,18            ;5" sectors/track, dden
024E FDCB046E       00240         BIT     5,(IY+4)        ;Dbl sided?
0252 2801           00250         JR      Z,NOTDBL
0254 87             00260         ADD     A,A             ;Adjust to 36 sect/cyl
0255 32A502         00270 NOTDBL  LD      (SECTRK),A
                    00280 ;
                    00290 ;       Set up for a fragmented file
                    00300 ;
0258 D9             00310         EXX
0259 0E06           00320         LD      C,6             ;Sectors/gran
025B CDB102         00330         CALL    GETEXT          ;Pick up extent 1
025E D9             00340         EXX
                    00350 ;
025F CD6802         00360         CALL    LOAD            ;Read in sysres
0262 3EFB           00370         LD      A,0FBH          ;EI instruction
0264 32950F         00380         LD      (DISKEI),A      ;  stuffed into FDCDVR
0267 E9             00390         JP      (HL)            ;Continue system init
                    00400 ;
0268 CD9702         00410 LOAD    CALL    RDBYTE          ;Get type code
026B 3D             00420         DEC     A
026C 200C           00430         JR      NZ,LOAD2        ;Bypass if not type 1
026E CD8802         00440         CALL    GETADR          ;Get blk len & load adr
0271 CD9702         00450 LOAD1   CALL    RDBYTE          ;Start reading the block
0274 77             00460         LD      (HL),A          ;Stuff into memory
0275 23             00470         INC     HL              ;Bump memory pointer
0276 10F9           00480         DJNZ    LOAD1           ;Loop for entire block
0278 18EE           00490         JR      LOAD            ;Restart the process
                    00500 ;
027A 3D             00510 LOAD2   DEC     A               ;Test if type 2 (traadr)
027B 280B           00520         JR      Z,GETADR        ;Ah, go if transfer addr
027D CD9702         00530         CALL    RDBYTE          ;Assume comment,
0280 47             00540         LD      B,A             ;  get comment length
0281 CD9702         00550 LOAD3   CALL    RDBYTE          ;  & ignore it
0284 10FB           00560         DJNZ    LOAD3
0286 18E0           00570         JR      LOAD            ;Continue to read
                    00580 ;
                    00590 ;       got the transfer address type code
                    00600 ;
0288 CD9702         00610 GETADR  CALL    RDBYTE          ;Get block length
028B 47             00620         LD      B,A
028C CD9702         00630         CALL    RDBYTE          ;Get lo-order load addr
```

Bootstrap Loader

```
028F 6F       00640          LD      L,A
0290 05       00650          DEC     B               ;Adj length for this byte
0291 CD9702   00660          CALL    RDBYTE          ;Get hi-order load addr
0294 67       00670          LD      H,A
0295 05       00680          DEC     B               ;Adj length for this byte
0296 C9       00690          RET
              00700 ;
              00710 ;        routine to read a byte
              00720 ;
0297 D9       00730 RDBYTE   EXX                     ;Switch memory/buf ptrs
0298 2C       00740          INC     L               ;Bump buf pointer
0299 2013     00750          JR      NZ,RDB2         ;Bypass disk i/o if more
029B C5       00760          PUSH    BC
029C 0609     00770          LD      B,9             ;Read sector function #
029E CD7004   00780          CALL    DCT$            ;Get another sector
02A1 C1       00790          POP     BC
02A2 1C       00800          INC     E               ;Bump sector counter
02A3 7B       00810          LD      A,E
02A4 D600     00820          SUB     $-$             ;Is this the last sector
02A5          00830 SECTRK   EQU     $-1
02A6 2002     00840          JR      NZ,RDB1         ;  on the cylinder?
02A8 5F       00850          LD      E,A             ;Yes, restart at 0
02A9 14       00860          INC     D               ;  & bump the cylinder up
02AA 05       00870 RDB1     DEC     B               ;Dec sectors this extent
02AB CCB102   00880          CALL    Z,GETEXT        ;Get next extent if 0
02AE 7E       00890 RDB2     LD      A,(HL)          ;P/u a byte
02AF D9       00900          EXX                     ;Exc mem/buf pointers
02B0 C9       00910          RET
              00920 ;
              00930 ;        Load DE track,sector, B sectors this extent
              00940 ;
              00950 GETEXT
02B1 DD23     00960          INC     IX              ;Index directory entry
02B3 DD23     00970          INC     IX              ;Pt at grans this ext.
02B5 DD7E00   00980          LD      A,(IX)
02B8 F5       00990          PUSH    AF              ;Save for later
02B9 07       01000          RLCA
02BA 07       01010          RLCA                    ;Normalize start gran
02BB 07       01020          RLCA
02BC E607     01030          AND     7
02BE CDCE02   01040          CALL    MULTCA          ;Start gran * grans/sec
02C1 5F       01050          LD      E,A             ;This is start sector
02C2 F1       01060          POP     AF
02C3 E61F     01070          AND     000011111B      ;Get total grans
02C5 3C       01080          INC     A               ;  this extent
02C6 CDCE02   01090          CALL    MULTCA          ;  * sect/gran
02C9 47       01100          LD      B,A             ;Sectors this extent
02CA DD56FF   01110          LD      D,(IX-1)        ;Cyl this extent
02CD C9       01120          RET
              01130 ;
              01140 ;        Short multiply C * A
              01150 ;
02CE C5       01160 MULTCA   PUSH    BC              ;Save sect/gran in C
02CF 57       01170          LD      D,A
02D0 AF       01180          XOR     A
02D1 0608     01190          LD      B,8
02D3 87       01200 MLTCA    ADD     A,A
02D4 CB21     01210          SLA     C
02D6 3001     01220          JR      NC,MLTCA1
```

Bootstrap Loader

```
Ø2D8 82       Ø123Ø          ADD     A,D
Ø2D9 1ØF8     Ø124Ø MLTCA1   DJNZ    MLTCA
Ø2DB C1       Ø125Ø          POP     BC
Ø2DC C9       Ø126Ø          RET
              Ø127Ø ;
              Ø128Ø ;        Initialize the CRTC
              Ø129Ø ;
              Ø13ØØ INITCRTC
Ø2DD Ø188ØF   Ø131Ø          LD      BC,15<8!88H     ;Count, CRTC address reg
Ø2EØ 21FDØ2   Ø132Ø          LD      HL,CRTCTAB
Ø2E3 7E       Ø133Ø $A1      LD      A,(HL)
Ø2E4 ED41     Ø134Ø                                  ;Pass reg # to CRTC
Ø2E6 D389     Ø135Ø                                  ;Pass value to CRTC reg
Ø2E8 2B       Ø136Ø          DEC     HL              ;Backup to next value
Ø2E9 Ø5       Ø137Ø          DEC     B               ;To next lower reg
Ø2EA F2E3Ø2   Ø138Ø          JP      P,$A1
Ø2ED C9       Ø139Ø          RET
Ø2EE 63       Ø14ØØ          DB      99              ;Horiz total MD
Ø2EF 5Ø       Ø141Ø          DB      8Ø              ;Horiz displayed MD
Ø2FØ 56       Ø142Ø          DB      86              ;Horiz sync position MD
Ø2F1 Ø8       Ø143Ø          DB      8               ;Horiz sync width
Ø2F2 18       Ø144Ø          DB      24              ;Vertical total
Ø2F3 ØØ       Ø145Ø          DB      Ø               ;Vertical total adjust
Ø2F4 18       Ø146Ø          DB      24              ;Vertical displayed
Ø2F5 18       Ø147Ø          DB      24              ;Vertical sync position
Ø2F6 ØØ       Ø148Ø          DB      Ø               ;Interlace mode
Ø2F7 Ø9       Ø149Ø          DB      9               ;Maximum scan line addr
Ø2F8 65       Ø15ØØ          DB      65H             ;Cursor start
Ø2F9 Ø9       Ø151Ø          DB      9               ;Cursor end
Ø2FA ØØ       Ø152Ø          DB      Ø               ;Start address (H)
Ø2FB ØØ       Ø153Ø          DB      Ø               ;Start address (L)
Ø2FC ØØ       Ø154Ø          DB      Ø               ;Cursor (H)
Ø2FD ØØ       Ø155Ø CRTCTAB  DB      Ø               ;Cursor (L)
Ø2FE ØØ       Ø156Ø          DC      -$&ØFFH,Ø
     ØØ
              Ø157Ø ;
              Ø158Ø ;        System BOOT entry point, loaded by ROM
              Ø159Ø ;
Ø3ØØ          Ø16ØØ CORE$    DEFL    $
43ØØ          Ø161Ø          ORG     43ØØH
43ØØ ØØ       Ø162Ø BOOT     NOP
43Ø1 FE14     Ø163Ø          CP      14H             ;Directory track location
43Ø2          Ø164Ø DIRTRK   EQU     $-1
43Ø3 F3       Ø165Ø          DI
43Ø4 3E86     Ø166Ø                                  ;Bring up the RAM
43Ø6 D384     Ø167Ø
43Ø8 3278ØØ   Ø168Ø          LD      (OPREG$),A
43ØB 21ØØF8   Ø169Ø          LD      HL,CRTBGN$      ;Clear video RAM
43ØE 11Ø1F8   Ø17ØØ          LD      DE,CRTBGN$+1
4311 Ø17FØ7   Ø171Ø          LD      BC,CRTSIZE-1
4314 362Ø     Ø172Ø          LD      (HL),' '
4316 EDBØ     Ø173Ø          LDIR
4318 21CD43   Ø174Ø          LD      HL,NMIRET       ;Set NMI vector
431B 2267ØØ   Ø175Ø          LD      (NMIVECT+1),HL
431E 3EC3     Ø176Ø          LD      A,ØC3H
432Ø 3266ØØ   Ø177Ø          LD      (NMIVECT),A
4323 3EC9     Ø178Ø          LD      A,ØC9H          ;Stuff return for ints
4325 3238ØØ   Ø179Ø          LD      (38H),A
              Ø18ØØ ;
```

Bootstrap Loader

```
                    01810 ;       Read the first 16 sectors of track 0
                    01820 ;
4328 210002         01830         LD      HL,START$+200H  ;Pt to page 2
432B 55             01840         LD      D,L             ;Init to track 0, sec 0
432C 5D             01850         LD      E,L
432D CD7743         01860 RDBOOT  CALL    RDSEQ           ;Read a sector
4330 24             01870         INC     H               ;Bump to next page
4331 1C             01880         INC     E               ;Bump to next
4332 3E10           01890         LD      A,16
4334 BB             01900         CP      E               ;Loop if more
4335 20F6           01910         JR      NZ,RDBOOT
4337 CDDD02         01920         CALL    INITCRTC        ;Initialize the CRTC
                    01930 ;
                    01940 ;       Now set up to load SYSRES
                    01950 ;
433A 3A0243         01960         LD      A,(DIRTRK)      ;P/u dir cyl
433D 327904         01970         LD      (DCT$+9),A      ;Update DCT to show DIR
4340 57             01980         LD      D,A             ;Set starting track and
4341 1E00           01990         LD      E,0             ;  init to read the GAT
4343 CD7443         02000         CALL    RDSECT          ;  into BUFFER
4346 3ACD12         02010         LD      A,(BUFFER+0CDH) ;Update DCT$ to show
4349 E620           02020         AND     20H             ;  the # of sides
434B 217404         02030         LD      HL,DCT$+4
434E B6             02040         OR      (HL)
434F 77             02050         LD      (HL),A
4350 1E04           02060         LD      E,4             ;Pt to SYS0 dir sector
4352 CD7443         02070         CALL    RDSECT          ;Read the SYS0 dir sec
4355 3A0012         02080         LD      A,(BUFFER)      ;Test if system disk
4358 E610           02090         AND     10H
435A 282D           02100         JR      Z,NOTSYS        ;Go if not
435C 211D12         02110         LD      HL,BUFFER+21+8  ;SYS0 extent info
435F 11F643         02120         LD      DE,BOOTBUF      ;Use 43FF-8
4362 010800         02130         LD      BC,8
4365 EDB8           02140         LDDR                    ;Store 1st four extents
4367 D5             02150         PUSH    DE              ;Pt IX to 1 byte
4368 DDE1           02160         POP     IX              ;  before extent info
436A D9             02170         EXX
436B 21FF12         02180         LD      HL,BUFFER+255   ;Init to buffer end
436E D9             02190         EXX
436F C33802         02200         JP      LBOOT           ;Load SYSRES
4372 00             02210         DB      0,0             ;Padding for posn
     00
                    02220 ;
                    02230 ;       routine to read a sector
                    02240 ;
4374 210012         02250 RDSECT  LD      HL,BUFFER       ;Set buffer
4377 0605           02260 RDSEQ   LD      B,5             ;Init retry counter
4379 C5             02270 RDS1    PUSH    BC              ;Save counter
437A E5             02280         PUSH    HL              ;Save for retries
437B CD9643         02290         CALL    READ            ;Attempt read
437E E1             02300         POP     HL
437F C1             02310         POP     BC
4380 E61C           02320         AND     1CH             ;Mask status
4382 C8             02330         RET     Z               ;Return if no error
4383 10F4           02340         DJNZ    RDS1            ;Loop for retry
4385 21E043         02350 GOTERR  LD      HL,DISKERR      ;"Disk error"
4388 DD             02360         DB      0DDH            ;Hide next instruction
4389 21EA43         02370 NOTSYS  LD      HL,NOSYS        ;"No system"
438C 010A00         02380         LD      BC,ERRLEN
```

Bootstrap Loader

```
438F 1193FB   Ø239Ø          LD      DE,8Ø*11+CRTBGN$+35     ;Middle of screen
4392 EDBØ     Ø24ØØ          LDIR
4394 18FE     Ø241Ø HALTS    JR      HALTS           ;Wait for RESET
              Ø242Ø ;
4396 Ø1F481   Ø243Ø READ                             ;Set DDEN, DS1, d.s. port
4399 ED41     Ø244Ø                                  ;Select it
439B ØD       Ø245Ø          DEC     C               ;Point C to data reg
439C 3E18     Ø246Ø          LD      A,18H           ;Seek command (6 ms)
439D          Ø247Ø BOOTST$  EQU     $-1             ;Set for boot step rate
              Ø248Ø          IFNE    BOOTST$,439DH
              Ø249Ø          ERR     'Boot step out of position'
              Ø25ØØ          ENDIF
439E ED51     Ø251Ø                                  ;Set desired track
43AØ CDD943   Ø252Ø          CALL    FDCMD           ;Pass command & delay
43A3 DBFØ     Ø253Ø SEEK1    IN      A,(ØFØH)        ;Get status
43A5 CB47     Ø254Ø          BIT     Ø,A             ;Busy?
43A7 2ØFA     Ø255Ø          JR      NZ,SEEK1
43A9 7B       Ø256Ø          LD      A,E             ;Set sector register
43AA D3F2     Ø257Ø
43AC 3E81     Ø258Ø          LD      A,81H           ;Set DDEN & DS1
43AE D3F4     Ø259Ø
43BØ D5       Ø26ØØ          PUSH    DE
43B1 11Ø2C1   Ø261Ø          LD      DE,81H!4ØH<8!2  ;D=DS1 + DDEN + WSGEN
              Ø262Ø                                  ;E=Mask to see DRQ
43B4 3E8Ø     Ø263Ø          LD      A,8ØH           ;FDC READ command
43B6 CDD943   Ø264Ø          CALL    FDCMD           ;Pass to ctrlr & set B=Ø
43B9 3ECØ     Ø265Ø          LD      A,ØCØH          ;Enable INTRQ & timeout
43BB D3E4     Ø266Ø
43BD DBFØ     Ø267Ø READLP1                          ;Grab status
43BF A3       Ø268Ø          AND     E               ;Test bit 1
43CØ 28FB     Ø269Ø          JR      Z,READLP1
43C2 EDA2     Ø27ØØ          INI
43C4 7A       Ø271Ø          LD      A,D             ;Set DDEN & DS1 & WSGEN
43C5 D3F4     Ø272Ø READLP2                          ;Continue to select
43C7 EDA2     Ø273Ø          INI                     ;  while inputting
43C9 2ØFA     Ø274Ø          JR      NZ,READLP2
43CB 18FE     Ø275Ø          JR      $               ;Wait for NMI
43CD D1       Ø276Ø NMIRET   POP     DE              ;Pop interrupt ret
43CE D1       Ø277Ø          POP     DE              ;Restore DE
43CF AF       Ø278Ø          XOR     A               ;Disable INTRQ & timeout
43DØ D3E4     Ø279Ø
43D2 3E81     Ø28ØØ          LD      A,81H           ;Reselect drive
43D4 D3F4     Ø281Ø
43D6 DBFØ     Ø282Ø                                  ;Get status
43D8 C9       Ø283Ø          RET
43D9 D3FØ     Ø284Ø FDCMD                            ;Give cmd to ctrlr
43DB Ø618     Ø285Ø          LD      B,24            ;Time delay
43DD 1ØFE     Ø286Ø          DJNZ    $
43DF C9       Ø287Ø          RET
43EØ 44       Ø288Ø DISKERR  DB      'Disk error'
     69 73 6B 2Ø 65 72 72 6F
     72
43EA 4E       Ø289Ø NOSYS    DB      'No system '
     6F 2Ø 73 79 73 74 65 6D
     2Ø
ØØØA          Ø29ØØ ERRLEN   EQU     $-NOSYS         ;Length of error msg
43F4 ØØ       Ø291Ø          DC      -$&ØFFH,Ø
     ØØ ØØ ØØ ØØ ØØ ØØ ØØ ØØ
     ØØ ØØ ØØ
```

Bootstrap Loader

```
0400            02920           ORG     CORE$+256
                00900 ;
0400            00910           SUBTTL  '<SYSinfo Section>'
```

SYSinfo Section

```
                    00930 ;
                    00940 ;       Page 3 - System stack and Sysinfo section
                    00950 ;
0380                00960 STACK$  EQU     $-128           ;Start stack 128 bytes low
0382                00970 PAUSE@  EQU     STACK$+2        ;Where pause will be
                    00980 ;
                    00990 ;       Page 4 - Miscellaneous stuff
                    01000 ;
0400 62             01010         DB      62H             ;Operating system version
0401 C9             01020 ZERO$   DB      0C9H            ;Config on BOOT, yes = 0
0401                01030 MAXDAY$ EQU     $-1             ;Max days per month
0402 1F             01040         DB      31,28,31,30,31,30,31,31,30,31,30,31
     1C 1F 1E 1F 1E 1F 1F 1E
     1F 1E 1F
0002                01050 HIGH$   DS      2               ;Highest available memory
0410 4C             01060 PAKNAM$ DB      'LS-DOS62Level-xx'
     53 2D 44 4F 53 36 32 4C
     65 76 65 6C 2D 78 78
                    01070 ;
                    01080 ;       Command line input buffer & AUTO buffer area
                    01090 ;
0420 0D             01100 INBUF$  DB      0DH             ;Input buffer - 80 bytes
0421 00             01110         DC      79,0
     00 00 00 00 00 00 00 00
     00 00 00 00 00 00 00 00
     00 00 00 00 00 00 00 00
     00 00 00 00 00 00 00 00
     00 00 00 00 00 00 00 00
     00 00 00 00 00 00 00 00
     00 00 00 00 00 00 00 00
     00 00 00 00 00 00 00 00
     00 00 00 00 00 00
                    01120 ;
                    01130 ;       System drive code tables
                    01140 ;
0470                01150 DCT$    EQU     $               ;System drive code tables
0470 C33D0E         01160         JP      FDCDVR          ;Floppy drive 0
0473 44             01170         DB      44H,0C1H,0,27H,17,3-1<5+6-1,20
     C1 00 27 11 45 14
047A C33D0E         01180         JP      FDCDVR          ;Floppy drive 1
047D 44             01190         DB      44H,42H,-1,27H,17,3-1<5+6-1,20
     42 FF 27 11 45 14
0484 C9             01200         RET                     ;Disable drive #2
0485 3D0E           01210         DW      FDCDVR
0487 44             01220         DB      44H,44H,-1,27H,17,3-1<5+6-1,20
     44 FF 27 11 45 14
048E C9             01230         RET                     ;Disable drive #3
048F 3D0E           01240         DW      FDCDVR
0491 44             01250         DB      44H,48H,-1,27H,17,3-1<5+6-1,20
     48 FF 27 11 45 14
0498 C9             01260         RET                     ;Logical drive 4
0499 2A0F           01270         DW      FDCRET
049B 00             01280         DB      0,0,0,27H,0,0,0
     00 00 27 00 00 00
04A2 C9             01290         RET                     ;Logical drive 5
04A3 2A0F           01300         DW      FDCRET
04A5 00             01310         DB      0,0,0,27H,0,0,0
     00 00 27 00 00 00
04AC C9             01320         RET                     ;Logical drive 6
```

SYSinfo Section

```
04AD 2A0F      01330          DW      FDCRET
04AF 00        01340          DB      0,0,0,27H,0,0,0
     00 00 27 00 00 00
04B6 C9        01350          RET                     ;Logical drive 7
04B7 2A0F      01360          DW      FDCRET
04B9 00        01370          DB      0,0,0,27H,0,0,0
     00 00 27 00 00 00
               01380 ;
               01390 ;        SYSINFO - miscellaneous information
               01400 ;
04C0 FF        01410 DSKTYP$  DB      -1              ;0 = DATA, <> 0 = SYS
04C1 00        01420          DB      0               ;Reserved
04C2 00        01430 DTPMT$   DB      0               ;Date prompt at boot
04C3 FF        01440 TMPMT$   DB      -1              ;Time prompt at boot
04C4 00        01450 RSTOR$   DB      0               ;Suppress restores on BOOT
0002           01460          DS      2               ;Reserved
04C7 53        01470 DAYTBL$  DB      'SunMonTueWedThuFriSat'
     75 6E 4D 6F 6E 54 75 65
     57 65 64 54 68 75 46 72
     69 53 61 74
04DC 4A        01480 MONTBL$  DB      'JanFebMarAprMayJunJulAugSepOctNovDec'
     61 6E 46 65 62 4D 61 72
     41 70 72 4D 61 79 4A 75
     6E 4A 75 6C 41 75 67 53
     65 70 4F 63 74 4E 6F 76
     44 65 63
               01490 ;
               01500 ;        End of low core assignments
               01510 ;
0500           01520 *GET     IODVR:3                 ;I/O driver, KEYIN, etc.
               02940 ;IODVR/ASM - LS-DOS 6.2
0500           02950          SUBTTL  '<Device I/O handling>
```

Device I/O handling

```
                    Ø297Ø ;
ØØ1C                Ø298Ø HOME    EQU     1CH
ØØ1F                Ø299Ø CLRFRM  EQU     1FH
                    Ø3ØØØ ;
                    Ø3Ø1Ø ;       Log out routine - display & log
                    Ø3Ø2Ø ;
Ø5ØØ CD2DØ5         Ø3Ø3Ø @LOGOT  CALL    @DSPLY
                    Ø3Ø4Ø ;
                    Ø3Ø5Ø ;       Job log loger routine
                    Ø3Ø6Ø ;
Ø5Ø3 3A3ØØ2         Ø3Ø7Ø @LOGER  LD      A,(JLDCB$)      ;If NIL, don't do
Ø5Ø6 EEØ8           Ø3Ø8Ø         XOR     8               ;  anything
Ø5Ø8 E6Ø8           Ø3Ø9Ø         AND     8
Ø5ØA C8             Ø31ØØ         RET     Z
Ø5ØB E5             Ø311Ø         PUSH    HL              ;Save pointer to command
Ø5ØC 211DØ5         Ø312Ø         LD      HL,LOGBUF       ;Get time string into buf
Ø5ØF E5             Ø313Ø         PUSH    HL
Ø51Ø CD8DØ7         Ø314Ø         CALL    @TIME
Ø513 E1             Ø315Ø         POP     HL
Ø514 113ØØ2         Ø316Ø         LD      DE,JLDCB$       ;Log the time
Ø517 CD3ØØ5         Ø317Ø         CALL    @MSG
Ø51A E1             Ø318Ø         POP     HL              ;Log the command
Ø51B 1813           Ø319Ø         JR      @MSG
Ø51D 68             Ø32ØØ LOGBUF  DB      'hh:mm:ss  ',3
     68 3A 6D 6D 3A 73 73 2Ø
     2Ø Ø3
                    Ø321Ø ;
                    Ø322Ø ;       Line print routine
                    Ø323Ø ;
Ø528 1118Ø2         Ø324Ø @PRINT  LD      DE,PRDCB$       ;Printer DCB
Ø52B 18Ø3           Ø325Ø         JR      @MSG
                    Ø326Ø ;
                    Ø327Ø ;       Line display routine
                    Ø328Ø ;
Ø52D 111ØØ2         Ø329Ø @DSPLY  LD      DE,DODCB$       ;Video DCB
                    Ø33ØØ ;
                    Ø331Ø ;       Device message routine
                    Ø332Ø ;
                    Ø333Ø *MOD
Ø53Ø E5             Ø334Ø @MSG    PUSH    HL              ;Save pointer to message
Ø531 7E             Ø335Ø $B1     LD      A,(HL)          ;P/u a message character
Ø532 FEØ3           Ø336Ø         CP      3               ;Exit on ETX
Ø534 28ØD           Ø337Ø         JR      Z,$B3
Ø536 FEØD           Ø338Ø         CP      CR              ;Exit & put on ENTER
Ø538 28Ø6           Ø339Ø         JR      Z,$B2
Ø53A C445Ø6         Ø34ØØ         CALL    NZ,@PUT         ;Else put the char
Ø53D 23             Ø341Ø         INC     HL              ;  & loop on no error
Ø53E 28F1           Ø342Ø         JR      Z,$B1           ;  else fall thru & exit
Ø54Ø CC45Ø6         Ø343Ø $B2     CALL    Z,@PUT
Ø543 E1             Ø344Ø $B3     POP     HL
Ø544 C9             Ø345Ø         RET
                    Ø346Ø ;
                    Ø347Ø ;       Clear screen routine
                    Ø348Ø ;
Ø545 3E1C           Ø349Ø @CLS    LD      A,HOME          ;Cursor home to Ø,Ø
Ø547 CD4DØ5         Ø35ØØ         CALL    DSPBYT
Ø54A CØ             Ø351Ø         RET     NZ              ;Return on error
Ø54B 3E1F           Ø352Ø         LD      A,CLRFRM        ;Clear to end of frame
Ø54D D5             Ø353Ø DSPBYT  PUSH    DE
Ø54E CD42Ø6         Ø354Ø         CALL    @DSP
```

Device I/O handling

```
Ø551 D1          Ø3550           POP     DE
Ø552 C9          Ø3560           RET
                 Ø3570 ;
                 Ø3580 ;         Check and Clear <BREAK> bit SVC
                 Ø3590 ;
                 Ø3600 @CKBRKC
Ø553 E5          Ø3610           PUSH    HL              ;Save registers
Ø554 217400      Ø3620           LD      HL,KFLAG$       ;Point to KFLAG$
Ø557 CB46        Ø3630           BIT     Ø,(HL)          ;Check break bit
Ø559 281A        Ø3640           JR      Z,NOBRK         ;  and ret if none
Ø55B F5          Ø3650           PUSH    AF              ;Save flags
Ø55C C5          Ø3660           PUSH    BC
Ø55D D5          Ø3670           PUSH    DE
Ø55E CB86        Ø3680 BRKTEST   RES     Ø,(HL)          ;Reset the break bit
Ø56Ø Ø1ØØØB      Ø3690           LD      BC,ØBØØH        ;Wait more than 1/3Ø
Ø563 CD82Ø3      Ø3700           CALL    PAUSE@          ;  of a second
Ø566 CB46        Ø3710           BIT     Ø,(HL)          ;Test the bit again
Ø568 2ØF4        Ø3720           JR      NZ,BRKTEST      ;Loop until gone
Ø56A 11Ø8Ø2      Ø3730           LD      DE,KIDCB$       ;Point at keyboard &
Ø56D 3EØ3        Ø3740           LD      A,Ø3            ;  clear buffer with
Ø56F CD23Ø6      Ø3750           CALL    @CTL            ;  control 3 call
Ø572 D1          Ø3760           POP     DE
Ø573 C1          Ø3770           POP     BC              ;Recover registers
Ø574 F1          Ø3780           POP     AF              ;Recover FLAGS
Ø575 E1          Ø3790 NOBRK     POP     HL
Ø576 C9          Ø3800           RET
                 Ø3810 ;
                 Ø3820 ;         Keyboard line input routine
                 Ø3830 ;
                 Ø3840 *MOD
                 Ø3850 ;
                 Ø3860 ;         Backspace to beginning of line
                 Ø3870 ;
Ø577 CDDBØ5      Ø3880 $C4       CALL    $C6             ;Backspace
Ø57A 2B          Ø3890           DEC     HL              ;Get the char prior
Ø57B 7E          Ø3900           LD      A,(HL)          ;  to the current
Ø57C 23          Ø3910           INC     HL
Ø57D FEØA        Ø3920           CP      ØAH             ;Return if line feed
Ø57F C8          Ø3930           RET     Z
Ø58Ø 78          Ø3940 $C5       LD      A,B             ;Check for empty buffer
Ø581 B9          Ø3950           CP      C
Ø582 2ØF3        Ø3960           JR      NZ,$C4          ;Loop if not
Ø584 C9          Ø3970           RET                     ;  else return
Ø585 E5          Ø3980 @KEYIN    PUSH    HL              ;Save buffer pointer
Ø586 48          Ø3990           LD      C,B             ;Set C = buffer size
Ø587 1128Ø6      Ø4ØØØ $C1       LD      DE,@KEY         ;Init for standard input
Ø58A 3A7CØØ      Ø4Ø1Ø           LD      A,(SFLAG$)      ;If JCL is active,
Ø58D E62Ø        Ø4Ø2Ø           AND     2ØH             ;  then use the JCL input
Ø58F 28Ø2        Ø4Ø3Ø           JR      Z,$CØ           ;Must loop here in case
Ø591 1E3Ø        Ø4Ø4Ø           LD      E,@JCL&ØFFH     ;  JCL exits with //STOP
Ø593 ED5398Ø5    Ø4Ø5Ø $CØ       LD      ($C1A+1),DE
Ø597 CDØØØØ      Ø4Ø6Ø $C1A      CALL    $-$             ;Get a key
Ø59A 2Ø3D        Ø4Ø7Ø           JR      NZ,$C3B         ;Back on error
Ø59C FE8Ø        Ø4Ø8Ø           CP      8ØH             ;Break?
Ø59E 2875        Ø4Ø9Ø           JR      Z,$C1Ø
Ø5AØ FE2Ø        Ø41ØØ           CP      2ØH             ;Go if not a control
Ø5A2 3Ø2Ø        Ø411Ø           JR      NC,$C2
Ø5A4 FEØD        Ø412Ø           CP      ØDH             ;Carriage return?
Ø5A6 286E        Ø413Ø           JR      Z,$C11
```

Device I/O handling

```
Ø5A8 FE1F    Ø414Ø         CP      1FH             ;Clear?
Ø5AA 2825    Ø415Ø         JR      Z,$C3
Ø5AC 1187Ø5  Ø416Ø         LD      DE,$C1          ;Set return address
Ø5AF D5      Ø417Ø         PUSH    DE
Ø5BØ FEØ8    Ø418Ø         CP      Ø8H             ;Backspace?
Ø5B2 2827    Ø419Ø         JR      Z,$C6
Ø5B4 FE18    Ø42ØØ         CP      18H             ;Backspace to BOL?
Ø5B6 28C8    Ø421Ø         JR      Z,$C5
Ø5B8 FEØ9    Ø422Ø         CP      Ø9H             ;Tab?
Ø5BA 283C    Ø423Ø         JR      Z,$C8
Ø5BC FE12    Ø424Ø         CP      'R'&1FH         ;CTL-R?
Ø5BE 282A    Ø425Ø         JR      Z,$C7
Ø5CØ FEØA    Ø426Ø         CP      ØAH             ;Line feed?
Ø5C2 CØ      Ø427Ø         RET     NZ              ;Ret if none above
Ø5C3 D1      Ø428Ø         POP     DE              ;Pop the return
Ø5C4 77      Ø429Ø $C2     LD      (HL),A          ;Stuff the char
Ø5C5 78      Ø43ØØ         LD      A,B             ;Check on buffer full
Ø5C6 B7      Ø431Ø         OR      A
Ø5C7 28BE    Ø432Ø         JR      Z,$C1           ;Loop if so
Ø5C9 7E      Ø433Ø         LD      A,(HL)          ;  else get char
Ø5CA 23      Ø434Ø         INC     HL              ;  & bump pointer
Ø5CB Ø5      Ø435Ø         DEC     B               ;Count down
Ø5CC CD42Ø6  Ø436Ø         CALL    @DSP            ;Display entry
Ø5CF 18Ø6    Ø437Ø         JR      $C3A            ;  then loop
             Ø438Ø ;
             Ø439Ø ;       Clear the screen invoked
             Ø44ØØ ;
Ø5D1 CD45Ø5  Ø441Ø $C3     CALL    @CLS
Ø5D4 41      Ø442Ø         LD      B,C             ;Reset to start of
Ø5D5 E1      Ø443Ø         POP     HL              ;  line & start of
Ø5D6 E5      Ø444Ø         PUSH    HL              ;  buffer
Ø5D7 28AE    Ø445Ø $C3A    JR      Z,$C1
Ø5D9 183B    Ø446Ø $C3B    JR      $C11
             Ø447Ø ;
             Ø448Ø ;       Backspace key entry
             Ø449Ø ;
Ø5DB 78      Ø45ØØ $C6     LD      A,B             ;If buffer is empty,
Ø5DC B9      Ø451Ø         CP      C               ;  return
Ø5DD C8      Ø452Ø         RET     Z
Ø5DE 2B      Ø453Ø         DEC     HL              ;  else do the backspace
Ø5DF 7E      Ø454Ø         LD      A,(HL)
Ø5EØ FEØA    Ø455Ø         CP      ØAH             ;Last char a linefeed?
Ø5E2 23      Ø456Ø         INC     HL
Ø5E3 C8      Ø457Ø         RET     Z               ;Return if so
Ø5E4 2B      Ø458Ø         DEC     HL
Ø5E5 Ø4      Ø459Ø         INC     B               ;Add back one char
Ø5E6 3EØ8    Ø46ØØ         LD      A,8             ;Backspace the cursor
Ø5E8 1858    Ø461Ø         JR      @DSP
             Ø462Ø ;
             Ø463Ø ;       Test if repeat last command
             Ø464Ø ;
Ø5EA 3A6CØØ  Ø465Ø $C7     LD      A,(CFLAG$)      ;Test if SYS1 KEYIN bit
Ø5ED E6Ø4    Ø466Ø         AND     4               ;  is set (bit 2)
Ø5EF C8      Ø467Ø         RET     Z               ;Ignore CTL if not
Ø5FØ 78      Ø468Ø         LD      A,B             ;If not at 1st position,
Ø5F1 B9      Ø469Ø         CP      C               ;  don't permit it
Ø5F2 CØ      Ø47ØØ         RET     NZ
Ø5F3 E1      Ø471Ø         POP     HL              ;Pop return to KEY
Ø5F4 E1      Ø472Ø         POP     HL              ;Point to command buffer
```

Device I/O handling

```
Ø5F5 C32DØ5    Ø473Ø          JP      @DSPLY          ;Display the old command
               Ø474Ø ;
               Ø475Ø ;        Tab entered
               Ø476Ø ;
Ø5F8 E5        Ø477Ø $C8      PUSH    HL              ;Get pos on line
Ø5F9 CDF1ØD    Ø478Ø          CALL    ADDR_2_ROWCOL   ;Get row,col in HL
Ø5FC 7D        Ø479Ø          LD      A,L             ;Xfer column to A
Ø5FD E1        Ø48ØØ          POP     HL
Ø5FE E6Ø7      Ø481Ø          AND     7
Ø6ØØ ED44      Ø482Ø          NEG                     ;Negate and add tab
Ø6Ø2 C6Ø8      Ø483Ø          ADD     A,8
Ø6Ø4 5F        Ø484Ø          LD      E,A             ;Reg E has tab length
Ø6Ø5 78        Ø485Ø $C9      LD      A,B             ;Check on buffer full
Ø6Ø6 B7        Ø486Ø          OR      A
Ø6Ø7 C8        Ø487Ø          RET     Z
Ø6Ø8 3E2Ø      Ø488Ø          LD      A,' '           ;Put spaces until
Ø6ØA 77        Ø489Ø          LD      (HL),A          ;  tab expanded
Ø6ØB 23        Ø49ØØ          INC     HL
Ø6ØC CD4DØ5    Ø491Ø          CALL    DSPBYT
Ø6ØF CØ        Ø492Ø          RET     NZ
Ø61Ø Ø5        Ø493Ø          DEC     B               ;Dec buffer remaining
Ø611 1D        Ø494Ø          DEC     E               ;Dec tab count
Ø612 C8        Ø495Ø          RET     Z
Ø613 18FØ      Ø496Ø          JR      $C9
               Ø497Ø ;
               Ø498Ø ;        Exit KEYIN routine
               Ø499Ø ;
Ø615 37        Ø5ØØØ $C1Ø     SCF                     ;BREAK exit with CF
Ø616 F5        Ø5Ø1Ø $C11     PUSH    AF              ;Save flag
Ø617 3EØD      Ø5Ø2Ø          LD      A,ØDH           ;Stuff CR at end
Ø619 77        Ø5Ø3Ø          LD      (HL),A
Ø61A CD42Ø6    Ø5Ø4Ø          CALL    @DSP            ;  & display it
Ø61D 79        Ø5Ø5Ø          LD      A,C             ;Calculate # of chars
Ø61E 9Ø        Ø5Ø6Ø          SUB     B               ;  entered
Ø61F 47        Ø5Ø7Ø          LD      B,A
Ø62Ø F1        Ø5Ø8Ø          POP     AF              ;Rcvr flag
Ø621 E1        Ø5Ø9Ø          POP     HL              ;Restore buffer ptr
Ø622 C9        Ø51ØØ          RET
               Ø511Ø ;
               Ø512Ø ;        Byte I/O device handler
               Ø513Ø ;                C => character if PUT or CTL
               Ø514Ø ;                DE => Device control block
               Ø515Ø ;
               Ø516Ø *MOD
Ø623 C5        Ø517Ø @CTL     PUSH    BC
Ø624 Ø6Ø4      Ø518Ø          LD      B,4             ;Bit 2, CTL
Ø626 182Ø      Ø519Ø          JR      IOBGN
Ø628 CD35Ø6    Ø52ØØ @KEY     CALL    @KBD            ;Scan the keyboard
Ø62B C8        Ø521Ø          RET     Z               ;Ret if key available
Ø62C B7        Ø522Ø          OR      A               ;Return if error
Ø62D 28F9      Ø523Ø          JR      Z,@KEY
Ø62F C9        Ø524Ø          RET
Ø63Ø 11Ø3Ø2    Ø525Ø @JCL     LD      DE,JCLCB$       ;JCL file FCB
Ø633 18Ø3      Ø526Ø          JR      @GET
Ø635 11Ø8Ø2    Ø527Ø @KBD     LD      DE,KIDCB$       ;Keyboard DCB
Ø638 C5        Ø528Ø @GET     PUSH    BC
Ø639 Ø6Ø1      Ø529Ø          LD      B,1             ;Bit Ø, GET
Ø63B 18ØB      Ø53ØØ          JR      IOBGN
Ø63D 1118Ø2    Ø531Ø @PRT     LD      DE,PRDCB$       ;Printer DCB
```

Device I/O handling

```
Ø64Ø 18Ø3     Ø532Ø          JR      @PUT
Ø642 111ØØ2   Ø533Ø @DSP     LD      DE,DODCB$       ;Video DCB
Ø645 C5       Ø534Ø @PUT     PUSH    BC
Ø646 Ø6Ø2     Ø535Ø          LD      B,2             ;Bit 1, PUT
Ø648 DDE5     Ø536Ø IOBGN    PUSH    IX              ;Save the registers
Ø64A E5       Ø537Ø          PUSH    HL
Ø64B D5       Ø538Ø          PUSH    DE              ;Xfer DCB to IX
Ø64C DDE1     Ø539Ø          POP     IX
Ø64E D5       Ø54ØØ          PUSH    DE
Ø64F 4F       Ø541Ø          LD      C,A             ;Xfer the I/O char
Ø65Ø 218ØØ6   Ø542Ø          LD      HL,@RSTREG      ;Restore register routine
Ø653 3AØ2Ø2   Ø543Ø          LD      A,(LBANK$)      ;If bank Ø is not
Ø656 B7       Ø544Ø          OR      A               ;  resident, need to
Ø657 28ØE     Ø545Ø          JR      Z,$DØ           ;  get it resident!
              Ø546Ø ;
              Ø547Ø ;        Some other bank is resident - invoke bank Ø
              Ø548Ø ;
Ø659 C5       Ø549Ø          PUSH    BC              ;Save reg again
Ø65A AF       Ø55ØØ          XOR     A               ;Prepare for bank-Ø
Ø65B 47       Ø551Ø          LD      B,A
Ø65C 4F       Ø552Ø          LD      C,A
Ø65D CD77Ø8   Ø553Ø          CALL    @BANK           ;Invoke bank-Ø
Ø66Ø 6Ø       Ø554Ø          LD      H,B             ;Get old bank data
Ø661 69       Ø555Ø          LD      L,C             ;  into reg HL
Ø662 C1       Ø556Ø          POP     BC              ;Rcvr BC
Ø663 E5       Ø557Ø          PUSH    HL              ;Bank data to stack
Ø664 2179Ø6   Ø558Ø          LD      HL,RSTBNK       ;Set return address
Ø667 E5       Ø559Ø $DØ      PUSH    HL              ;  to restore registers
Ø668 1A       Ø56ØØ          LD      A,(DE)          ;P/u DCB type byte
Ø669 B7       Ø561Ø          OR      A
Ø66A C8       Ø562Ø          RET     Z               ;Back if nothing
Ø66B FEØ8     Ø563Ø          CP      8               ;Ck on GET/PUT/CTL
Ø66D 3Ø1A     Ø564Ø          JR      NC,@CHNIO       ;Branch if special
Ø66F DD6EØ1   Ø565Ø          LD      L,(IX+1)        ;  else p/u the vector
Ø672 DD66Ø2   Ø566Ø          LD      H,(IX+2)
Ø675 78       Ø567Ø $D1      LD      A,B             ;Xfer I/O code
Ø676 FEØ2     Ø568Ø          CP      2               ;Set flags state
Ø678 E9       Ø569Ø          JP      (HL)
Ø679 C1       Ø57ØØ RSTBNK   POP     BC              ;Get old bank data
Ø67A F5       Ø571Ø          PUSH    AF              ;Can't affect AF
Ø67B 79       Ø572Ø          LD      A,C             ;Request to A
Ø67C CD77Ø8   Ø573Ø          CALL    @BANK           ;Bring back original bank
Ø67F F1       Ø574Ø          POP     AF
Ø68Ø D1       Ø575Ø @RSTREG  POP     DE              ;Restore regs
Ø681 E1       Ø576Ø          POP     HL
Ø682 DDE1     Ø577Ø          POP     IX
Ø684 C1       Ø578Ø          POP     BC
Ø685 C9       Ø579Ø          RET
              Ø58ØØ ;
Ø686 E5       Ø581Ø $D2      PUSH    HL
Ø687 DDE1     Ø582Ø          POP     IX
Ø689 DD6EØ1   Ø583Ø @CHNIO   LD      L,(IX+1)        ;P/u vector address
Ø68C DD66Ø2   Ø584Ø          LD      H,(IX+2)
Ø68F DD7EØØ   Ø585Ø $D3      LD      A,(IX+Ø)        ;P/u the DCB type
Ø692 B7       Ø586Ø          OR      A               ;File Control Block?
Ø693 FAØØ13   Ø587Ø          JP      M,@BYTEIO
Ø696 CB5F     Ø588Ø          BIT     3,A             ;Test NIL bit 2nd
Ø698 2Ø24     Ø589Ø          JR      NZ,$D5
Ø69A CB67     Ø59ØØ          BIT     4,A             ;Routed?
```

Device I/O handling

```
Ø69C 2ØE8     Ø591Ø          JR      NZ,$D2          ;Go get routed DCB
Ø69E CB6F     Ø592Ø          BIT     5,A             ;If not linked, then
Ø6AØ 28D3     Ø593Ø          JR      Z,$D1           ;  must be filtered
Ø6A2 E5       Ø594Ø          PUSH    HL              ;Point to the link DCB
Ø6A3 DDE1     Ø595Ø          POP     IX
Ø6A5 DD7ØØ3   Ø596Ø          LD      (IX+3),B        ;Save the direction
Ø6A8 DDE5     Ø597Ø          PUSH    IX
Ø6AA CD89Ø6   Ø598Ø          CALL    @CHNIO          ;I/O to 1st device
Ø6AD DDE1     Ø599Ø          POP     IX
Ø6AF DD46Ø3   Ø6ØØØ          LD      B,(IX+3)        ;P/u the direction
Ø6B2 2ØØC     Ø6Ø1Ø          JR      NZ,$D6          ;Go on NZ flag
              Ø6Ø2Ø ;
              Ø6Ø3Ø ;        Z-flag on return - check input/output
              Ø6Ø4Ø ;
Ø6B4 CB4Ø     Ø6Ø5Ø          BIT     Ø,B             ;If input & got char,
Ø6B6 DD6EØ4   Ø6Ø6Ø $D4      LD      L,(IX+4)        ;  p/u the linked DCB
Ø6B9 DD66Ø5   Ø6Ø7Ø          LD      H,(IX+5)
Ø6BC 28C8     Ø6Ø8Ø          JR      Z,$D2
Ø6BE BF       Ø6Ø9Ø $D5      CP      A
Ø6BF C9       Ø61ØØ          RET
              Ø611Ø ;
              Ø612Ø ;        1st link got NZ condition - if input, get link
              Ø613Ø ;
Ø6CØ CB4Ø     Ø614Ø $D6      BIT     Ø,B             ;Was it input/output?
Ø6C2 28Ø3     Ø615Ø          JR      Z,$D7           ;Output is error
Ø6C4 B7       Ø616Ø          OR      A               ;If A=Ø, then no input
Ø6C5 28EF     Ø617Ø          JR      Z,$D4
Ø6C7 B7       Ø618Ø $D7      OR      A
Ø6C8 C9       Ø619Ø          RET
Ø6C9          Ø153Ø *GET     MULDIV:3                ;16-bit MULT & DIV
              Ø62ØØ ;MULDIV/ASM - 16 x 8 multiplication & division
Ø6C9          Ø621Ø          SUBTTL  '<16 X 8 multiply/divide>'
```

16 X 8 multiply/divide

```
                    Ø623Ø *MOD
                    Ø624Ø ;
                    Ø625Ø ;       Multiply HL by A - SVC 91
                    Ø626Ø ;       HL => multiplicand
                    Ø627Ø ;       A  => multiplier
                    Ø628Ø ;       HLA <= 24-bit result
                    Ø629Ø ;       DE destroyed
                    Ø63ØØ ;
Ø6C9 C5             Ø631Ø @MUL16  PUSH    BC              ;Save reg BC
Ø6CA EB             Ø632Ø         EX      DE,HL           ;Multiplicand to DE
Ø6CB 4F             Ø633Ø         LD      C,A             ;  & multiplier to C
Ø6CC 21ØØØØ         Ø634Ø         LD      HL,Ø            ;Init value to zero
Ø6CF 7D             Ø635Ø         LD      A,L             ;  in regs HLA
Ø6DØ Ø6Ø8           Ø636Ø         LD      B,8             ;Init for 8-bit mult
Ø6D2 29             Ø637Ø $E1     ADD     HL,HL           ;Shift to next place
Ø6D3 17             Ø638Ø         RLA                     ;Use A for bits 16-23
Ø6D4 CBØ1           Ø639Ø         RLC     C               ;Multiply this bit?
Ø6D6 3ØØ3           Ø64ØØ         JR      NC,$E2          ;Go if not
Ø6D8 19             Ø641Ø         ADD     HL,DE           ;Else add multiplicand
Ø6D9 CEØØ           Ø642Ø         ADC     A,Ø             ;  & any overflow to 16
Ø6DB 1ØF5           Ø643Ø $E2     DJNZ    $E1             ;Loop for 8 bits
Ø6DD 4F             Ø644Ø         LD      C,A             ;Tempy save
Ø6DE 7D             Ø645Ø         LD      A,L             ;Xfer low-order to A
Ø6DF 6C             Ø646Ø         LD      L,H             ;Xfer mid-order to L
Ø6EØ 61             Ø647Ø         LD      H,C             ;Xfer hi-order to H
Ø6E1 C1             Ø648Ø         POP     BC
Ø6E2 C9             Ø649Ø         RET
                    Ø65ØØ ;
                    Ø651Ø ;       Divide HL by A - SVC 94
                    Ø652Ø ;       HL => dividend
                    Ø653Ø ;       A  => divisor
                    Ø654Ø ;       HL <= resulting quotient
                    Ø655Ø ;       A  <= remainder
                    Ø656Ø ;
                    Ø657Ø *MOD
Ø6E3 D5             Ø658Ø @DIV16  PUSH    DE              ;Save this reg pair
Ø6E4 57             Ø659Ø         LD      D,A             ;Xfer divisor to D
Ø6E5 1E1Ø           Ø66ØØ         LD      E,16            ;Init for 16-bits
Ø6E7 AF             Ø661Ø         XOR     A
Ø6E8 29             Ø662Ø $F1     ADD     HL,HL           ;Rotate dividend
Ø6E9 17             Ø663Ø         RLA                     ;  & subtract divisor if
Ø6EA 38Ø3           Ø664Ø         JR      C,$F2           ;  carry into bit-16
Ø6EC BA             Ø665Ø         CP      D               ;Compare divisor
Ø6ED 38Ø2           Ø666Ø         JR      C,$F3           ;Go if no subtract
Ø6EF 92             Ø667Ø $F2     SUB     D               ;  else subtract divisor
Ø6FØ 2C             Ø668Ø         INC     L               ;Set lo-order
Ø6F1 1D             Ø669Ø $F3     DEC     E               ;Count down one bit
Ø6F2 2ØF4           Ø67ØØ         JR      NZ,$F1          ;Loop for 16-bits
Ø6F4 D1             Ø671Ø         POP     DE
Ø6F5 C9             Ø672Ø         RET
                    Ø673Ø ;
                    Ø674Ø ;       @HEXDEC - SVC 97
                    Ø675Ø ;       Routine to convert 16-bit hexadecimal to decimal
                    Ø676Ø ;       HL => value
                    Ø677Ø ;       DE => buffer pointer of 5-character buffer
                    Ø678Ø ;       HL <= destroyed (always set to zero)
                    Ø679Ø ;       DE <= Buffer+5
                    Ø68ØØ ;       BC <= destroyed
                    Ø681Ø ;       Z  <= set
                    Ø682Ø ;
```

16 X 8 multiply/divide

```
                    Ø683Ø *MOD
Ø6F6 Ø6Ø5           Ø684Ø @HEXDEC LD      B,5             ;Length max
Ø6F8 3E2Ø           Ø685Ø         LD      A,' '           ;Load blank
Ø6FA 12             Ø686Ø HEXDEC1 LD      (DE),A          ;To string
Ø6FB 13             Ø687Ø         INC     DE              ;Bump pointer
Ø6FC 1ØFC           Ø688Ø         DJNZ    HEXDEC1         ;Go for length
Ø6FE D5             Ø689Ø         PUSH    DE              ;Save end +1
Ø6FF 1B             Ø69ØØ         DEC     DE              ;Adjust back
Ø7ØØ 3EØA           Ø691Ø HEXDEC2 LD      A,1Ø            ;Base to convert to
Ø7Ø2 CDE3Ø6         Ø692Ø         CALL    @DIV16          ;HL+A  =  HL/A
Ø7Ø5 C63Ø           Ø693Ø         ADD     A,'Ø'           ;Add ASCII to result
Ø7Ø7 12             Ø694Ø         LD      (DE),A          ;  to user string
Ø7Ø8 1B             Ø695Ø         DEC     DE              ;Move back
                    Ø696Ø ;
                    Ø697Ø ;       Check if done
                    Ø698Ø ;
Ø7Ø9 7C             Ø699Ø         LD      A,H             ;Get subtotal remainder
Ø7ØA B5             Ø7ØØØ         OR      L               ;Done?
Ø7ØB 2ØF3           Ø7Ø1Ø         JR      NZ,HEXDEC2      ;Go till completed
Ø7ØD D1             Ø7Ø2Ø         POP     DE              ;Restore end+1
Ø7ØE C9             Ø7Ø3Ø         RET                     ;Return Z
                    Ø7Ø4Ø ;
Ø7ØF                Ø154Ø *GET    CLOCKS:3                ;Hardware task stuff
                    Ø7Ø5Ø ;CLOCKS/ASM - LS-DOS 6.2
Ø7ØF                Ø7Ø6Ø         SUBTTL  '<Heartbeat & Bank handling>'
```

Heartbeat & Bank handling

```
                    Ø7ØØ *MOD
                    Ø7Ø1Ø ;
                    Ø7Ø2Ø ;       Model IV time clock & blinking cursor
                    Ø7Ø3Ø ;
Ø7ØF 3C             Ø7Ø4Ø TIMETBL DB      6Ø,6Ø,24,3Ø     ;Sec/min, min/hr, hr/day
     3C 18 1E
                    Ø7Ø5Ø TIMTSK$
Ø713 3A97ØB         Ø7Ø6Ø         LD      A,(CRSAVE)      ;If cursor not on,
Ø716 B7             Ø7Ø7Ø         OR      A               ;  then don't blink
Ø717 217FØØ         Ø7Ø8Ø         LD      HL,VFLAG$       ;Point to video flag
Ø71A 2829           Ø7Ø9Ø         JR      Z,$H2
                    Ø71ØØ                                 ;Check if blinking
Ø71C CB7E           Ø711Ø         BIT     7,(HL)          ;Check system INHIBIT
Ø71E CBBE           Ø712Ø         RES     7,(HL)          ;Allow blink next time
Ø72Ø 2Ø23           Ø713Ø         JR      NZ,$H2
Ø722 34             Ø714Ø         INC     (HL)            ;Increment the counter
Ø723 CB5E           Ø715Ø         BIT     3,(HL)          ;  & see if to 8
Ø725 281E           Ø716Ø         JR      Z,$H2           ;Not this time
Ø727 CB9E           Ø717Ø         RES     3,(HL)          ;Reset counter
Ø729 CB76           Ø718Ø         BIT     6,(HL)          ;Check if SOLID cursor
Ø72B 28Ø2           Ø719Ø         JR      Z,NOSOLID       ;If not, then blink
Ø72D CBEE           Ø72ØØ         SET     5,(HL)          ;Force SOLID mode
Ø72F CD17Ø8         Ø721Ø NOSOLID CALL    ENADIS_DO_RAM   ;Bring up the video RAM
Ø732 7E             Ø722Ø         LD      A,(HL)          ;Grab the toggle bit
Ø733 EE2Ø           Ø723Ø         XOR     2ØH             ;  and flip it
Ø735 77             Ø724Ø         LD      (HL),A
Ø736 E62Ø           Ø725Ø         AND     2ØH             ;Was it on?
Ø738 ED5B95ØB       Ø726Ø         LD      DE,(CURSOR)     ;Get the cursor pos
Ø73C 3A97ØB         Ø727Ø         LD      A,(CRSAVE)      ;  and char under cursor
Ø73F 2ØØ3           Ø728Ø         JR      NZ,$H1          ;Put character if flip on
Ø741 3A98ØB         Ø729Ø         LD      A,(CRSCHAR)     ;  else put the cursor
Ø744 12             Ø73Ø0 $H1     LD      (DE),A          ;Put the char
Ø745 DD21ØFØ7       Ø731Ø $H2     LD      IX,TIMETBL      ;Point to data area
Ø749 DD35Ø3         Ø732Ø         DEC     (IX+3)          ;Count down by 3Ø
Ø74C CØ             Ø733Ø         RET     NZ              ;Back if not one second
                    Ø734Ø         IF      @HZ5Ø
                    Ø735Ø         LD      (IX+3),25       ;Set for 5Ø hertz
                    Ø736Ø HERTZ$  EQU     $-1
                    Ø737Ø         ELSE                    ;  else use 6Ø hertz
Ø74D DD36Ø31E       Ø738Ø         LD      (IX+3),3Ø       ;Reset for one second
Ø75Ø                Ø739Ø HERTZ$  EQU     $-1
                    Ø74ØØ         ENDIF
Ø751 CB66           Ø741Ø         BIT     4,(HL)          ;Is clock on? (VFLAG$)
Ø753 28Ø4           Ø742Ø         JR      Z,$H3           ;Go if off
Ø755 1187Ø7         Ø743Ø         LD      DE,CLOCK        ;Set to display clock
Ø758 D5             Ø744Ø         PUSH    DE
Ø759 Ø6Ø3           Ø745Ø $H3     LD      B,3
Ø75B 212DØØ         Ø746Ø         LD      HL,TIME$
Ø75E 11ØFØ7         Ø747Ø         LD      DE,TIMETBL      ;Pt to max sec, min, hr
Ø761 34             Ø748Ø TIMER1  INC     (HL)            ;Bump time parm
Ø762 1A             Ø749Ø         LD      A,(DE)
Ø763 96             Ø75ØØ         SUB     (HL)
Ø764 CØ             Ø751Ø         RET     NZ              ;Ret if not max
Ø765 77             Ø752Ø         LD      (HL),A          ;  else set to Ø
Ø766 2C             Ø753Ø         INC     L               ;Pt to next parm
Ø767 1C             Ø754Ø         INC     E
Ø768 1ØF7           Ø755Ø         DJNZ    TIMER1          ;Loop thru 3 parms
                    Ø756Ø ;
                    Ø757Ø ;       Update date at midnight
                    Ø758Ø ;
```

Heartbeat & Bank handling

```
076A 2E34     07670          LD      L,DATE$+1&0FFH  ;Point to day of month
076C 110204   07680          LD      DE,MAXDAY$+1    ;Point to test table
076F 34       07690          INC     (HL)            ;Bump the day
0770 2C       07700          INC     L               ;Point to month
0771 7E       07710          LD      A,(HL)          ;Get the month
0772 2D       07720          DEC     L
0773 3D       07730          DEC     A               ;Index into table
0774 83       07740          ADD     A,E
0775 5F       07750          LD      E,A
0776 1A       07760          LD      A,(DE)          ;P/u max days
0777 BE       07770          CP      (HL)            ;Is day in range?
0778 D0       07780          RET     NC              ;Return if it is
0779 3601     07790          LD      (HL),1          ;  else reset day to 1
077B 2C       07800          INC     L               ;  & bump the month
077C 34       07810          INC     (HL)
077D 7E       07820          LD      A,(HL)          ;If went past Dec,
077E D60D     07830          SUB     12+1            ;  then need to fix
0780 D8       07840          RET     C
0781 3601     07850          LD      (HL),1          ;Correct to Jan
0783 2D       07860          DEC     L               ;Backup to year
0784 2D       07870          DEC     L
0785 34       07880          INC     (HL)
0786 C9       07890          RET
              07900 ;
              07910 ;        Clock display processor
              07920 ;
              07930 CLOCK
0787 CD1708   07940          CALL    ENADIS_DO_RAM   ;Bring up the video
078A 2145F8   07950          LD      HL,CRTBGN$+69   ;Point to display CRT
078D 112F00   07960 @TIME    LD      DE,TIME$+2      ;Point to time$
0790 0E3A     07970          LD      C,':'           ;Set the separator
0792 0603     07980 TIME1    LD      B,3             ;Init for three fields
0794 1A       07990 TIME2    LD      A,(DE)          ;Get a field item
0795 362F     08000          LD      (HL),2FH        ;Init display
0797 34       08010 TIME3    INC     (HL)            ;Bump until proper digit
0798 D60A     08020          SUB     10
079A 30FB     08030          JR      NC,TIME3
079C C63A     08040          ADD     A,3AH           ;Correct the remainder
079E 23       08050          INC     HL              ;Bump to next display
079F 77       08060          LD      (HL),A          ;  & stuff the digit
07A0 23       08070          INC     HL
07A1 05       08080          DEC     B
07A2 C8       08090          RET     Z               ;Back when done
07A3 71       08100          LD      (HL),C          ;  else stuff separator
07A4 23       08110          INC     HL
07A5 1B       08120          DEC     DE              ;Point to next field
07A6 18EC     08130          JR      TIME2           ;  & loop
              08140 ;
              08150 ;        Return formatted date, HL => user buffer
              08160 ;
07A8 113500   08170 @DATE    LD      DE,DATE$+2
07AB 0E2F     08180          LD      C,'/'
07AD 18E3     08190          JR      TIME1
              08200 ;
07AF 0000     08210 PCSAVE$  DW      00              ;PC at entry to RST 38
              08220 ;
              08230 ;        Dynamic Trace routine
              08240 ;
              08250 TRACE_INT
```

Heartbeat & Bank handling

```
Ø7B1 B3Ø7     Ø826Ø          DW      $+2
Ø7B3 2AAFØ7   Ø827Ø          LD      HL,(PCSAVE$)
Ø7B6 EB       Ø828Ø          EX      DE,HL             ;Program counter to DE
Ø7B7 CD17Ø8   Ø829Ø          CALL    ENADIS_DO_RAM     ;Bring up the video
Ø7BA 213EF8   Ø83ØØ          LD      HL,CRTBGN$+62     ;Crt trace adr
              Ø831Ø ;
              Ø832Ø ;        Hexadecimal display routine
              Ø833Ø ;
Ø7BD 7A       Ø834Ø @HEX16   LD      A,D               ;Convert reg D to
Ø7BE CDC2Ø7   Ø835Ø          CALL    @HEX8             ;  two hex digits
Ø7C1 7B       Ø836Ø          LD      A,E               ;Convert reg E to
Ø7C2 F5       Ø837Ø @HEX8    PUSH    AF                ;  two hex digits
Ø7C3 1F       Ø838Ø          RRA                       ;Do left nybble first
Ø7C4 1F       Ø839Ø          RRA
Ø7C5 1F       Ø84ØØ          RRA
Ø7C6 1F       Ø841Ø          RRA
Ø7C7 CDCBØ7   Ø842Ø          CALL    HXD1              ;Bits Ø-3 stuffed in hex
Ø7CA F1       Ø843Ø          POP     AF                ;Reget the byte
Ø7CB E6ØF     Ø844Ø HXD1     AND     ØFH               ;  & use right nybble
Ø7CD C69Ø     Ø845Ø          ADD     A,9ØH             ;Convert nybble to hex
Ø7CF 27       Ø846Ø          DAA
Ø7DØ CE4Ø     Ø847Ø          ADC     A,4ØH
Ø7D2 27       Ø848Ø          DAA
Ø7D3 77       Ø849Ø          LD      (HL),A            ;Stuff in (HL)
Ø7D4 23       Ø85ØØ          INC     HL
Ø7D5 C9       Ø851Ø          RET
              Ø852Ø ;
              Ø853Ø ;        Scan for PAUSE or BREAK & set KFLAG$
              Ø854Ø ;
F48Ø          Ø855Ø SHIFT
              Ø856Ø          IF      @USA
F4Ø1          Ø857Ø KB1
              Ø858Ø          ENDIF
              Ø859Ø          IF      @GERMAN
              Ø86ØØ KB1
              Ø861Ø          ENDIF
              Ø862Ø          IF      @FRENCH
              Ø863Ø KB1
              Ø864Ø          ENDIF
F44Ø          Ø865Ø KB7
Ø7D6 CD17Ø8   Ø866Ø KCK@     CALL    ENADIS_DO_RAM     ;Bring up the keyboard
Ø7D9 217４ØØ   Ø867Ø          LD      HL,KFLAG$         ;Hang onto flag
Ø7DC 3A8ØF4   Ø868Ø          LD      A,(SHIFT)         ;P/u SHIFT row & ignore
Ø7DF E6Ø7     Ø869Ø          AND     7                 ;  CTRL key pressed
Ø7E1 2F       Ø87ØØ          CPL
Ø7E2 CB57     Ø871Ø          BIT     2,A
Ø7E4 C8       Ø872Ø          RET     Z                 ;Back if CTRL
              Ø873Ø ;
              Ø874Ø ;        Set carry flag if a SHIFT key is down
              Ø875Ø ;
Ø7E5 C6Ø1     Ø876Ø          ADD     A,1               ;Set CF if no SHIFT
Ø7E7 3F       Ø877Ø          CCF                       ;Set CF if SHIFT
Ø7E8 3ØØB     Ø878Ø          JR      NC,KCK1           ;No pause if no SHIFT
Ø7EA 3AØ1F4   Ø879Ø          LD      A,(KB1)           ;Test for "@"
              Ø88ØØ          IF      @USA
Ø7ED CB47     Ø881Ø          BIT     Ø,A
              Ø882Ø          ENDIF
              Ø883Ø          IF      @INTL
              Ø884Ø          BIT     4,A               ;Foriegn keyboard
```

Heartbeat & Bank handling

```
                    Ø885Ø          ENDIF
Ø7EF 28ØB           Ø886Ø          JR       Z,KCK1A          ;Bypass if no "@"
Ø7F1 CBCE           Ø887Ø          SET      1,(HL)           ;Turn on pause bit
Ø7F3 18Ø7           Ø888Ø          JR       KCK1A
                    Ø889Ø ;
                    Ø89ØØ ;        Inhibit test of unshifted BREAK if nested ENA_DO
                    Ø891Ø ;
Ø7F5 3A35Ø8         Ø892Ø KCK1     LD       A,(OPREG_SV_PTR) ;If not at highest level
Ø7F8 D66F           Ø893Ø          SUB      OPREG_SV_AREA+1&ØFFH ;  then don't allow
Ø7FA 2ØØ9           Ø894Ø          JR       NZ,KCK1B         ;  tasker BREAK handler
Ø7FC 3A4ØF4         Ø895Ø KCK1A    LD       A,(KB7)          ;Check on BREAK & ENTER
Ø7FF CB47           Ø896Ø          BIT      Ø,A              ;Check on ENTER
Ø8Ø1 28Ø2           Ø897Ø          JR       Z,KCK1B          ;Go if not
Ø8Ø3 CBD6           Ø898Ø          SET      2,(HL)           ;  else note set
Ø8Ø5 CB57           Ø899Ø KCK1B    BIT      2,A              ;Is <BREAK> depressed?
Ø8Ø7 F5             Ø9ØØØ          PUSH     AF
Ø8Ø8 28ØB           Ø9Ø1Ø          JR       Z,KCK2           ;Go if not
Ø8ØA 38Ø9           Ø9Ø2Ø          JR       C,KCK2           ;Ignore if shifted
Ø8ØC 3A7CØØ         Ø9Ø3Ø          LD       A,(SFLAG$)       ;Permit break bit only
Ø8ØF CB67           Ø9Ø4Ø          BIT      4,A              ;  if BREAK enabled?
Ø811 2ØØ2           Ø9Ø5Ø          JR       NZ,KCK2
Ø813 CBC6           Ø9Ø6Ø          SET      Ø,(HL)           ;Turn on BREAK bit
Ø815 F1             Ø9Ø7Ø KCK2     POP      AF               ;C=shift, NZ=break
Ø816 C9             Ø9Ø8Ø          RET
                    Ø9Ø9Ø ;
                    Ø91ØØ ;        Routine to enable video RAM & change stack if necessary
                    Ø911Ø ;
                    Ø912Ø *MOD
                    Ø913Ø ENADIS_DO_RAM
Ø817 F3             Ø914Ø          DI                        ;Can't while we test stack
Ø818 226BØ8         Ø915Ø          LD       (HLSAV),HL       ;Save HL but not on stack
Ø81B F5             Ø916Ø          PUSH     AF               ;Save AF
Ø81C E1             Ø917Ø          POP      HL
Ø81D 2266Ø8         Ø918Ø          LD       (AFSAV),HL
Ø82Ø 21Ø3ØC         Ø919Ø          LD       HL,ØCØ3H         ;Can't exceed X'F3FC'
Ø823 39             Ø92ØØ          ADD      HL,SP
Ø824 3ØØ9           Ø921Ø          JR       NC,$I1
                    Ø922Ø ;
                    Ø923Ø ;        Switch to the system stack
                    Ø924Ø ;
Ø826 E1             Ø925Ø          POP      HL               ;Transfer RET address
Ø827 ED7363Ø8       Ø926Ø          LD       (SPSAV),SP       ;Save stack pointer
Ø82B 316ØØ3         Ø927Ø          LD       SP,STACK$-2ØH    ;Keep room at top
Ø82E E5             Ø928Ø          PUSH     HL               ;Put RET back
Ø82F 2146Ø8         Ø929Ø $I1      LD       HL,DIS_DO_RAM    ;Stack return to disable
Ø832 E3             Ø93ØØ          EX       (SP),HL          ;  video RAM below RET
Ø833 E5             Ø931Ø          PUSH     HL
Ø834 216EØ8         Ø932Ø          LD       HL,OPREG_SV_AREA
Ø835                Ø933Ø OPREG_SV_PTR EQU  $-2
Ø837 23             Ø934Ø          INC      HL               ;Get next save location
Ø838 3A78ØØ         Ø935Ø          LD       A,(OPREG$)       ;P/u port mask
Ø83B 3ØØ2           Ø936Ø          JR       NC,$I2           ;Bypass if NC (no stack switch)
Ø83D E67F           Ø937Ø          AND      7FH              ;Strip bit 7 to use as flag
Ø83F 77             Ø938Ø $I2      LD       (HL),A           ;Save current state
Ø84Ø E6FC           Ø939Ø          AND      ØFCH             ;Strip SEL1 & SELØ
Ø842 F682           Ø94ØØ          OR       82H              ;Set SEL1,Ø = (1,Ø) & NZ condx
Ø844 1812           Ø941Ø          JR       DOOPREG          ;Set new assignment
                    Ø942Ø ;
                    Ø943Ø ;        Routine to disable video RAM
```

Heartbeat & Bank handling

```
                    Ø944Ø ;
                    Ø945Ø DIS_DO_RAM
Ø846 F3             Ø946Ø         DI                      ;Interrupts off
Ø847 226BØ8         Ø947Ø         LD      (HLSAV),HL      ;Save off of stack
Ø84A F5             Ø948Ø         PUSH    AF
Ø84B E1             Ø949Ø         POP     HL              ;Save AF
Ø84C 2266Ø8         Ø95ØØ         LD      (AFSAV),HL
Ø84F 2A35Ø8         Ø951Ø         LD      HL,(OPREG_SV_PTR)
Ø852 7E             Ø952Ø         LD      A,(HL)          ;P/u previous state
Ø853 CB7F           Ø953Ø         BIT     7,A             ;Test if we switch stack
Ø855 CBFF           Ø954Ø         SET     7,A             ;Make sure PAGE is set
Ø857 2B             Ø955Ø         DEC     HL
                    Ø956Ø ;
Ø858 2235Ø8         Ø957Ø DOOPREG LD      (OPREG_SV_PTR),HL
Ø85B 3278ØØ         Ø958Ø         LD      (OPREG$),A      ;Restore port image
Ø85E D384           Ø959Ø                                 ;  and the port
Ø86Ø 2ØØ3           Ø96ØØ         JR      NZ,$I3
                    Ø961Ø ;
                    Ø962Ø ;       Switch back to the old stack
                    Ø963Ø ;
Ø862 31ØØØØ         Ø964Ø         LD      SP,$-$          ;Get the old stack
Ø863                Ø965Ø SPSAV   EQU     $-2
Ø865 21ØØØØ         Ø966Ø $I3     LD      HL,$-$
Ø866                Ø967Ø AFSAV   EQU     $-2
Ø868 E5             Ø968Ø         PUSH    HL              ;Restore AF
Ø869 F1             Ø969Ø         POP     AF
Ø86A 21ØØØØ         Ø97ØØ         LD      HL,$-$          ;Restore HL
Ø86B                Ø971Ø HLSAV   EQU     $-2
Ø86D FB             Ø972Ø         EI                      ;Interrupts back on
Ø86E C9             Ø973Ø         RET
Ø86E                Ø974Ø OPREG_SV_AREA   EQU     $-1
Ø86F ØØ             Ø975Ø         DB      Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø
     ØØ ØØ ØØ ØØ ØØ ØØ ØØ
                    Ø976Ø ;
                    Ø977Ø ;       Bank selection SVC handler
                    Ø978Ø ;       HL=> transfer address for function B=Ø
                    Ø979Ø ;       C => Bank request <Ø-2>; Set bit 7 to transfer
                    Ø98ØØ ;       B => Request function
                    Ø981Ø ;               Ø => Select bank C
                    Ø982Ø ;               1 => Reset in-use bit of bank C
                    Ø983Ø ;               2 => Test in-use bit of bank C
                    Ø984Ø ;               3 => Set in-use bit of bank C
                    Ø985Ø ;
                    Ø986Ø *MOD
                    Ø987Ø @BANK
Ø877 E67F           Ø988Ø         AND     7FH             ;Strip possible bit 7
Ø879 FEØ3           Ø989Ø         CP      2+1             ;Bank out of range?
Ø87B D2EDØD         Ø99ØØ         JP      NC,PERR
Ø87E Ø5             Ø991Ø         DEC     B               ;Check option
Ø87F FAB3Ø8         Ø992Ø         JP      M,$J3           ;Go if bank select
Ø882 ØE86           Ø993Ø         LD      C,86H           ;Set for reset BUR$
Ø884 2819           Ø994Ø         JR      Z,$J1           ;Go if function 1
Ø886 ØE46           Ø995Ø         LD      C,46H           ;Set for test BUR$
Ø888 Ø5             Ø996Ø         DEC     B
Ø889 2814           Ø997Ø         JR      Z,$J1           ;Go if function 2
Ø88B Ø5             Ø998Ø         DEC     B
Ø88C 28Ø9           Ø999Ø         JR      Z,$JØ           ;Go on set BUR$
Ø88E Ø5             1ØØØØ         DEC     B
Ø88F C2EDØD         1ØØ1Ø PERRX   JP      NZ,PERR         ;SVC parameter error
```

Heartbeat & Bank handling

```
Ø892 3AØ2Ø2   1ØØ2Ø         LD      A,(LBANK$)      ;P/u current bank
Ø895 BF       1ØØ3Ø         CP      A
Ø896 C9       1ØØ4Ø         RET
Ø897 47       1ØØ5Ø $JØ     LD      B,A             ;Save the bank requested
Ø898 CD9FØ8   1ØØ6Ø         CALL    $J1             ;Test if in use already
Ø89B CØ       1ØØ7Ø         RET     NZ              ;Back if error
Ø89C 78       1ØØ8Ø         LD      A,B             ;Reget the request #
Ø89D ØEC6     1ØØ9Ø         LD      C,ØC6H          ;Set for set BUR$
Ø89F E6Ø7     1Ø1ØØ $J1     AND     7               ;Strip to bank Ø-7
Ø8A1 Ø7       1Ø11Ø         RLCA                    ;Shift <Ø-2> to <3-5>
Ø8A2 Ø7       1Ø12Ø         RLCA
Ø8A3 Ø7       1Ø13Ø         RLCA
Ø8A4 B1       1Ø14Ø         OR      C               ;Merge the code type
Ø8A5 32BØØ8   1Ø15Ø         LD      ($J2+1),A       ;Change the OP code
Ø8A8 AF       1Ø16Ø         XOR     A               ;Init Z-flag
Ø8A9 3EØ8     1Ø17Ø         LD      A,8             ;Init "Device not avail
Ø8AB E5       1Ø18Ø         PUSH    HL              ;Don't alter HL
Ø8AC 21ØØØ2   1Ø19Ø         LD      HL,BUR$         ;Point to bank-used-RAM
Ø8AF CB46     1Ø2ØØ $J2     BIT     Ø,(HL)          ;*** Modified instruction
Ø8B1 E1       1Ø21Ø         POP     HL
Ø8B2 C9       1Ø22Ø         RET
Ø8B3 E5       1Ø23Ø $J3     PUSH    HL              ;Ck if stack is in upper
Ø8B4 21Ø58Ø   1Ø24Ø         LD      HL,8ØØ5H        ;  bank area
Ø8B7 39       1Ø25Ø         ADD     HL,SP
Ø8B8 E1       1Ø26Ø         POP     HL
Ø8B9 DAEDØD   1Ø27Ø         JP      C,PERR          ;Error if > X'7FFE'
Ø8BC FEØ1     1Ø28Ø         CP      1               ;Change <Ø, 1, 2, 3>
Ø8BE 17       1Ø29Ø         RLA                     ;  to <1, 2, 4, 6>
Ø8BF 47       1Ø3ØØ         LD      B,A             ;  & save for later
Ø8CØ 3AØ1Ø2   1Ø31Ø         LD      A,(BAR$)        ;P/u Bank Avail Ram
Ø8C3 AØ       1Ø32Ø         AND     B               ;Is the bank installed?
Ø8C4 2ØC9     1Ø33Ø         JR      NZ,PERRX        ;Error if not in machine
Ø8C6 78       1Ø34Ø         LD      A,B             ;Get the requested bank
Ø8C7 1F       1Ø35Ø         RRA                     ;Change <1, 2, 4> to
Ø8C8 3F       1Ø36Ø         CCF                     ;  <Ø, 2, 3> {CF on Ø
Ø8C9 CEØØ     1Ø37Ø         ADC     A,Ø             ;  switched to 2 & 4}
Ø8CB Ø7       1Ø38Ø         RLCA                    ;Shift bits Ø-1
Ø8CC Ø7       1Ø39Ø         RLCA                    ;  to 4-5 (MBITØ,1)
Ø8CD Ø7       1Ø4ØØ         RLCA
Ø8CE Ø7       1Ø41Ø         RLCA
Ø8CF 47       1Ø42Ø         LD      B,A             ;Save bit mask
Ø8DØ 3A78ØØ   1Ø43Ø         LD      A,(OPREG$)      ;P/u current memory
Ø8D3 E68F     1Ø44Ø         AND     Ø8FH            ;  configuration &
Ø8D5 BØ       1Ø45Ø         OR      B               ;  mask off old &
Ø8D6 3278ØØ   1Ø46Ø         LD      (OPREG$),A      ;  merge the new
Ø8D9 D384     1Ø47Ø                                 ;Switch the hardware
Ø8DB 3AØ2Ø2   1Ø48Ø         LD      A,(LBANK$)      ;Get old bank #
Ø8DE 47       1Ø49Ø         LD      B,A             ;  & save it
Ø8DF 79       1Ø5ØØ         LD      A,C             ;P/u new bank #
Ø8EØ E67F     1Ø51Ø         AND     7FH             ;Strip any bit-7
Ø8E2 32Ø2Ø2   1Ø52Ø         LD      (LBANK$),A      ;  & save new bank #
Ø8E5 A9       1Ø53Ø         XOR     C               ;Keep bit-7
Ø8E6 BØ       1Ø54Ø         OR      B               ;Merge in new bank #
Ø8E7 4F       1Ø55Ø         LD      C,A             ;  & replace into C
Ø8E8 CB79     1Ø56Ø         BIT     7,C             ;Transfer to new bank?
Ø8EA Ø6ØØ     1Ø57Ø         LD      B,Ø             ;Init for invoke later
Ø8EC C8       1Ø58Ø         RET     Z               ;No if bit-7 = Ø
Ø8ED E3       1Ø59Ø         EX      (SP),HL         ;Exchange RET with new
Ø8EE BF       1Ø6ØØ         CP      A               ;  transfer & go to it
```

Heartbeat & Bank handling

```
Ø8EF C9       1Ø61Ø         RET
Ø8FØ          Ø155Ø @$SYS   EQU     $               ;Pointer for @GTMOD
              Ø156Ø         IF      @USA
Ø8FØ          Ø157Ø *GET    KIDVR:3                 ;Keyboard driver
              1Ø62Ø ;KIDVR/ASM - LS-DOS 6.2
Ø8FØ          1Ø63Ø         SUBTTL  '<Keyboard Driver>'
```

Keyboard Driver

```
                    1Ø65Ø *MOD
                    1Ø66Ø ;
ØØØA                1Ø67Ø LF       EQU     1Ø
ØØØD                1Ø68Ø CR       EQU     13
F4Ø1                1Ø69Ø KBØ                              ;Row Ø RAM address
F44Ø                1Ø7ØØ KB6                              ;Row 6 RAM address
F48Ø                1Ø71Ø SHIFT                            ;Row 7 RAM address
                    1Ø72Ø ;
Ø8FØ 1831           1Ø73Ø KIDVR    JR      KIBGN           ;Branch around linkage
Ø8F2 87ØB           1Ø74Ø          DW      KILAST          ;Last byte used
Ø8F4 Ø3             1Ø75Ø          DB      3,'$KI'
     24 4B 49
Ø8F8 Ø8Ø2           1Ø76Ø          DW      KIDCB$          ;Pointer to DCB
Ø8FA ØØØØ           1Ø77Ø          DW      Ø               ;Spare
Ø8FC ØØ             1Ø78Ø KIDATA$  DB      Ø               ;Last key entered
Ø8FD ØØ             1Ø79Ø          DB      Ø               ;Repeat time check
ØØØ2                1Ø8ØØ RPTINIT  EQU     $-KIDATA$
Ø8FE 16             1Ø81Ø          DB      22              ;22 * 33.3ms = .733 sec
ØØØ3                1Ø82Ø RPTRATE  EQU     $-KIDATA$
Ø8FF Ø2             1Ø83Ø          DB      2               ;2 x RTC rate
ØØØ4                1Ø84Ø KBROWØ   EQU     $-KIDATA$
Ø9ØØ FF             1Ø85Ø          DB      -1,-1,-1,-1     ;Image of rows Ø-3
     FF FF FF
ØØØ8                1Ø86Ø KBROW4   EQU     $-KIDATA$
Ø9Ø4 FF             1Ø87Ø          DB      -1,-1           ;Image of rows 4-5
     FF
ØØØA                1Ø88Ø KBROW6   EQU     $-KIDATA$
Ø9Ø6 FF             1Ø89Ø          DB      -1,-1           ;Image of rows 6-7
     FF
                    1Ø9ØØ ;
                    1Ø91Ø ;        Conversion table for keyboard row 7/8
                    1Ø92Ø ;
Ø9Ø8 ØD             1Ø93Ø KBTBL    DB      CR,1DH,1FH,1FH  ;<ENTER> <CLEAR>
     1D 1F 1F
Ø9ØC 8Ø             1Ø94Ø          DB      8ØH,Ø,ØBH,1BH   ;<BREAK> <UPARW>
     ØØ ØB 1B
Ø91Ø ØA             1Ø95Ø          DB      LF,1AH,8,18H    ;<DNARW> <LTARW>
     1A Ø8 18
Ø914 Ø9             1Ø96Ø          DB      9,19H,2ØH,2ØH   ;<RTARW> <SPACE>
     19 2Ø 2Ø
Ø918 81             1Ø97Ø          DB      81H,91H,82H,92H ;<F1> <F2>
     91 82 92
Ø91C 83             1Ø98Ø          DB      83H,93H         ;<F3>
     93
                    1Ø99Ø ;
                    11ØØØ ;        Table to generate 5B-5F, 7B-7F
                    11Ø1Ø ;
Ø91E 2C             11Ø2Ø SPCLTB   DB      ',/.;',CR
     2F 2E 3B ØD
                    11Ø3Ø ;
                    11Ø4Ø ;        Entry to keyboard driver
                    11Ø5Ø ;
Ø923 79             11Ø6Ø KIBGN    LD      A,C             ;Get the character
Ø924 F5             11Ø7Ø          PUSH    AF              ;Save flags
Ø925 CD89ØØ         11Ø8Ø          CALL    @KITSK          ;Hook for KI task
Ø928 F1             11Ø9Ø          POP     AF
                    111ØØ ;
                    1111Ø ;        Screen print (Control-*) processing
                    1112Ø ;
Ø929 CDD5ØA         1113Ø          CALL    TYPAHD          ;Chain downstream
```

Keyboard Driver

```
092C DØ       11140          RET     NC              ;Ret if not <CONTROL>
092D F5       11150          PUSH    AF              ;Save flag state
092E FE3A     11160          CP      ':'
0930 2802     11170          JR      Z,$K1           ;Go if screen print
0932 F1       11180          POP     AF
0933 C9       11190          RET
              11200 ;
              11210 ;        Perform a screen print
              11220 ;
0934 F1       11230 $K1      POP     AF              ;Clean the stack
0935 3A6DØØ   11240          LD      A,(DFLAG$)      ;Check on Graphic bit
0938 Ø7       11250          RLCA
0939 3E3E     11260          LD      A,3EH           ;Init for LD a,'.'
093B 3ØØ2     11270          JR      NC,$+4          ;Go if not Graphic
093D 3EFE     11280          LD      A,ØFEH          ;Change to CPR n
093F 325CØ9   11290          LD      ($K4),A         ;Stuff cpr or ld
0942 2174ØØ   11300          LD      HL,KFLAG$       ;Reset the BREAK bit
0945 CB86     11310          RES     Ø,(HL)
0947 E5       11320          PUSH    HL              ;Save on stack
0948 21ØØØØ   11330          LD      HL,Ø            ;Init for row,col
094B Ø6Ø1     11340 $K2      LD      B,1             ;Get a character at the
094D CD99ØB   11350          CALL    @VDCTL          ;  row-H, col-L
0950 2027     11360          JR      NZ,$K6          ;Go on error
0952 FE2Ø     11370          CP      2ØH
0954 3ØØ2     11380          JR      NC,$+4          ;Convert control codes
0956 C64Ø     11390          ADD     A,4ØH           ;  to cap A-Z, +
0958 FE8Ø     11400          CP      8ØH             ;Cvrt anything from X'8Ø'
095A 3802     11410          JR      C,$K5           ;  thru X'FF' to a '.'
095C 3E2E     11420 $K4      LD      A,'.'           ;  unless graphic bit set
095E CD3DØ6   11430 $K5      CALL    @PRT            ;Print the char & loop
0961 2Ø16     11440          JR      NZ,$K6
0963 2C       11450          INC     L               ;Bump column counter
0964 7D       11460          LD      A,L             ;Check for end-of-line
0965 D65Ø     11470          SUB     8Ø
0967 2ØE2     11480          JR      NZ,$K2          ;Loop if not EOL
0969 6F       11490          LD      L,A             ;Reset to column Ø
096A 2D       11500          DEC     L               ;Adj for CR force
096B E3       11510          EX      (SP),HL         ;Get KFLAG$
096C CB46     11520          BIT     Ø,(HL)          ;Exit with A=Ø on
096E E3       11530          EX      (SP),HL         ;  entrance of BREAK
096F 2ØØ8     11540          JR      NZ,$K6
0971 24       11550          INC     H               ;Bump row counter
0972 7C       11560          LD      A,H             ;Test for end of screen
0973 FE18     11570          CP      24
0975 3EØD     11580          LD      A,CR
0977 2ØE5     11590          JR      NZ,$K5          ;Put the CR & loop
0979 3EØD     11600 $K6      LD      A,CR            ;Close out with CR if
097B CD3DØ6   11610          CALL    @PRT            ;  BREAK key detected
097E E1       11620          POP     HL              ;Pop the KFLAG
097F CB86     11630          RES     Ø,(HL)          ;  & reset BREAK bit
0981 1832     11640          JR      NOCHAR
              11650 ;
              11660 ;        Driver to scan the keyboard
              11670 ;
              11680 *MOD
0983 DD21FCØ8 11690 KISCAN   LD      IX,KIDATA$      ;Point to data area
0987 21ØØØ9   11700          LD      HL,KIDATA$+KBROWØ ;Load kbd image start
098A Ø1Ø1F4   11710          LD      BC,KBØ          ;Load start of keyboard
098D 16ØØ     11720          LD      D,Ø             ;Zero the key counter
```

Keyboard Driver

```
Ø98F ØA       11730 $L1     LD      A,(BC)          ;Load 1st char from kbd
Ø99Ø 5F       11740         LD      E,A
Ø991 AE       11750         XOR     (HL)            ;XOR with old value
Ø992 2Ø26     11760         JR      NZ,$L2          ;Go if different
Ø994 14       11770         INC     D               ;Bump key counter
Ø995 23       11780         INC     HL              ;Bump image pointer
Ø996 CBØ1     11790         RLC     C               ;Go to next row
Ø998 F28FØ9   11800         JP      P,$L1           ;Loop until end of rows
Ø99B ØA       11810         LD      A,(BC)          ;Get row 7
Ø99C E678     11820         AND     Ø78H            ;Strip SHIFT, CTL
Ø99E 5F       11830         LD      E,A
Ø99F AE       11840         XOR     (HL)
Ø9AØ 2Ø18     11850         JR      NZ,$L2
Ø9A2 DD7EØØ   11860         LD      A,(IX+Ø)        ;Key down? It's same as
Ø9A5 B7       11870         OR      A               ;  the last if so
Ø9A6 28ØD     11880         JR      Z,NOCHAR        ;Ret if no key
Ø9A8 3A2CØØ   11890         LD      A,(TIMER$)      ;Do we repeat the
Ø9AB DD96Ø1   11900         SUB     (IX+1)          ;  same key?
Ø9AE 286E     11910         JR      Z,$L1Ø          ;Go repeat if time up
Ø9BØ DD96Ø2   11920         SUB     (IX+RPTINIT)    ;Beyond Ø.75 seconds?
Ø9B3 3869     11930         JR      C,$L1Ø          ;Go if yes
Ø9B5 F6Ø1     11940 NOCHAR  OR      1               ;Else don't repeat
Ø9B7 3EØØ     11950         LD      A,Ø             ;Show NZ with A=Ø
Ø9B9 C9       11960         RET
              11970 ;
              11980 ;       Found change in key matrix
              11990 ;
Ø9BA 73       12000 $L2     LD      (HL),E          ;Stuff KB image with new
Ø9BB A3       12010         AND     E               ;  KB row value
Ø9BC CA88ØA   12020         JP      Z,NOKEY         ;Go if new is none
              12030 ;
              12040 ;       Convert the depressed key
              12050 ;
Ø9BF 5F       12060         LD      E,A             ;Save the active bit
Ø9CØ 7A       12070         LD      A,D             ;Calculate 8 * row
Ø9C1 Ø7       12080         RLCA
Ø9C2 Ø7       12090         RLCA
Ø9C3 Ø7       12100         RLCA
Ø9C4 57       12110         LD      D,A             ;Save 8 * row
Ø9C5 ØEØ1     12120         LD      C,1             ;Add 8 * row + column
Ø9C7 79       12130 $L3     LD      A,C
Ø9C8 A3       12140         AND     E               ;Check if bits match
Ø9C9 2Ø19     12150         JR      NZ,$L6          ;Go if match
Ø9CB 14       12160         INC     D               ;  else bump value
Ø9CC CBØ1     12170         RLC     C               ;Shift compare bit
Ø9CE 18F7     12180         JR      $L3             ;Loop to test next
              12190 ;
              12200 ;       Key pressed was not an alpha
              12210 ;
Ø9DØ D69Ø     12220 $L4     SUB     9ØH             ;Adjust for non-alpha
Ø9D2 3Ø52     12230         JR      NC,$L9          ;Go if special key
Ø9D4 C64Ø     12240         ADD     A,4ØH           ;Cvrt to numeric/symbol
Ø9D6 FE3C     12250         CP      3CH             ;Manipulate to get
Ø9D8 38Ø2     12260         JR      C,$L5           ;  proper code
Ø9DA EE1Ø     12270         XOR     1ØH
Ø9DC CB43     12280 $L5     BIT     Ø,E             ;Check SHIFT
Ø9DE 286Ø     12290         JR      Z,$L11          ;Go if unshift
Ø9EØ EE1Ø     12300         XOR     1ØH             ;  else adjust for SHIFT
Ø9E2 185C     12310         JR      $L11
```

Keyboard Driver

```
                    12320 ;
                    12330 ;       Found a key - Set up the function codes
                    12340 ;
Ø9E4 3A8ØF4         12350 $L6     LD      A,(SHIFT)       ;P/u the SHIFT key
Ø9E7 5F             12360         LD      E,A             ;Merge RH & LH Shift keys
Ø9E8 E6Ø2           12370         AND     2               ;Only merge bit 1
Ø9EA ØF             12380         RRCA                    ;Bit 1 to bit Ø
Ø9EB B3             12390         OR      E               ;Merge bits Ø & 1
Ø9EC 5F             12400         LD      E,A             ;Value of (RHorLH) shift
Ø9ED 7A             12410         LD      A,D             ;Load semi-converted
Ø9EE C66Ø           12420         ADD     A,6ØH           ;If alpha, convert to
Ø9FØ FE8Ø           12430         CP      8ØH             ;  correct value
Ø9F2 217ØØØ         12440         LD      HL,KFLAG$
Ø9F5 3ØD9           12450         JR      NC,$L4          ;Go if not alpha
                    12460 ;
                    12470 ;       Alpha <@-Z> - If caps lock or <SHIFT>,
                    12480 ;       Convert to caps unless CLEAR
                    12490 ;
Ø9F7 CB53           12500         BIT     2,E             ;CTRL key down?
Ø9F9 2Ø18           12510         JR      NZ,CTLA2Z       ;CTRL sets <ØØ-1A>
Ø9FB FE6Ø           12520         CP      6ØH             ;Invert @ and `
Ø9FD 2ØØ4           12530         JR      NZ,$L7
Ø9FF EE2Ø           12540         XOR     2ØH             ;Invert & bypass test
ØAØ1 18ØA           12550         JR      $L8             ;  for CAPs lock
ØAØ3 DDCBØA4E       12560 $L7     BIT     1,(IX+KBROW6)   ;If CLEAR, don't test
ØAØ7 2ØØ4           12570         JR      NZ,$L8          ;  for CAPs lock
ØAØ9 CB6E           12580         BIT     5,(HL)          ;Caps lock?
ØAØB 2Ø31           12590         JR      NZ,TGLCASE
ØAØD CB43           12600 $L8     BIT     Ø,E             ;Shift key down?
ØAØF 282F           12610         JR      Z,$L11          ;Bypass if not shifted
ØA11 182B           12620         JR      TGLCASE         ;Convert to upper case
ØA13 D66Ø           12630 CTLA2Z  SUB     6ØH             ;Convert CTRL A-Z
ØA15 2Ø29           12640         JR      NZ,$L11         ;Go on A-Z
ØA17 CB43           12650         BIT     Ø,E             ;Shifted?
ØA19 37             12660         SCF                     ;Set C-flag for CTL-@
ØA1A C8             12670         RET     Z               ;  & return if unshifted
ØA1B 3E1C           12680         LD      A,1CH           ;  else set EOF error
ØA1D C9             12690         RET
ØA1E 3A2CØØ         12700 $L1Ø    LD      A,(TIMER$)      ;Advance time check
ØA21 DD86Ø3         12710         ADD     A,(IX+RPTRATE)  ;  by Ø.Ø67 seconds
ØA24 1872           12720         JR      $L12            ;Go output the key
                    12730 ;
                    12740 ;       Special keys - rows 6 & 7
                    12750 ;
ØA26 FEØB           12760 $L9     CP      11              ;Compress F1-F3 keys
ØA28 284F           12770         JR      Z,CAPSKEY       ;  while checking for CAP
ØA2A 38Ø2           12780         JR      C,$+4           ;  F1-F3 to 8-1Ø
ØA2C D6Ø4           12790         SUB     4
ØA2E 21Ø8Ø9         12800         LD      HL,KBTBL        ;Pt to special char table
ØA31 Ø7             12810         RLCA                    ;Index into table,
ØA32 CB43           12820         BIT     Ø,E             ;  shifted code is +1
ØA34 28Ø1           12830         JR      Z,$+3
ØA36 3C             12840         INC     A
ØA37 4F             12850         LD      C,A             ;Index the table
ØA38 Ø6ØØ           12860         LD      B,Ø
ØA3A Ø9             12870         ADD     HL,BC
ØA3B 7E             12880         LD      A,(HL)          ;Load char from table
ØA3C 18Ø2           12890         JR      $L11            ;Bypass restore of char
ØA3E EE2Ø           12900 TGLCASE XOR     2ØH             ;Toggle the case
```

Keyboard Driver

```
ØA4Ø FE8Ø     12910 $L11    CP      8ØH             ;BREAK key?
ØA42 2ØØF     12920         JR      NZ,$L11A        ;Ck on <BREAK> disable
ØA44 217CØØ   12930         LD      HL,SFLAG$       ;Break disabled?
ØA47 CB66     12940         BIT     4,(HL)
ØA49 2ØØ7     12950         JR      NZ,$L11B        ;Don't set bit if disabl
ØA4B 2174ØØ   12960         LD      HL,KFLAG$
ØA4E CBC6     12970         SET     Ø,(HL)          ;  otherwise set it
ØA5Ø 18Ø1     12980         JR      $L11A
ØA52 17       12990 $L11B   RLA                     ;Rotate bit-7 out
ØA53 DDCBØA4E 13000 $L11A   BIT     1,(IX+KBROW6)   ;CLEAR key pressed?
ØA57 28ØE     13010         JR      Z,NOTALPH       ;Go if not down
ØA59 57       13020         LD      D,A             ;Save code
ØA5A CBAF     13030         RES     5,A             ;Set to upper-case for
ØA5C D641     13040         SUB     'A'             ;  test A-Z
ØA5E FE1A     13050         CP      'Z'-'A'+1
ØA6Ø 7A       13060         LD      A,D             ;Get back actual char
ØA61 3ØØ2     13070         JR      NC,$+4          ;Go if not A-Z
ØA63 EE2Ø     13080         XOR     2ØH             ;Shift keyboard case
ØA65 F68Ø     13090         OR      8ØH             ;Set bit 7 for CLEAR key
ØA67 CB43     13100 NOTALPH BIT     Ø,E             ;SHIFT key down?
ØA69 2819     13110         JR      Z,FIXCLR        ;Go if not
ØA6B FE9F     13120 GOTSHFT CP      9FH             ;Shift-clear?
ØA6D 2813     13130         JR      Z,FIXSCL        ;Go if so
ØA6F FE2Ø     13140 TSTSPA  CP      2ØH             ;Shift Ø or shift sp?
ØA71 2Ø16     13150         JR      NZ,KEYOK        ;Go if not
ØA73 DDCBØ846 13160         BIT     Ø,(IX+KBROW4)   ;Ck zero key
ØA77 281Ø     13170         JR      Z,KEYOK         ;Go if not down
              13180 ;
              13190 ;       Toggle the caps lock bit in the KFLAG$
              13200 ;
ØA79 3E2Ø     13210 CAPSKEY LD      A,2ØH           ;CAPs wasn't 2ØH
ØA7B 2174ØØ   13220 CASHK$  LD      HL,KFLAG$       ;Reverse case by
ØA7E AE       13230         XOR     (HL)            ;  flipping bit 5
ØA7F 77       13240         LD      (HL),A
ØA8Ø 18Ø6     13250         JR      NOKEY
ØA82 EE8Ø     13260 FIXSCL  XOR     8ØH             ;Reset bit 7
ØA84 FE9F     13270 FIXCLR  CP      9FH             ;Clear key?
ØA86 2ØØ1     13280         JR      NZ,KEYOK        ;Go if not
ØA88 AF       13290 NOKEY   XOR     A
ØA89 DD77ØØ   13300 KEYOK   LD      (IX+Ø),A
ØA8C Ø184Ø1   13310         LD      BC,Ø184H        ;Delay
ØA8F CD82Ø3   13320 TYPHK$  CALL    PAUSE@
ØA92 3A2CØØ   13330         LD      A,(TIMER$)      ;Set initialization
ØA95 DD86Ø2   13340 DELAY2  ADD     A,(IX+RPTINIT)  ;  repeat key delay
ØA98 DD77Ø1   13350 $L12    LD      (IX+1),A        ;Save new repeat time
ØA9B DD7EØØ   13360         LD      A,(IX+Ø)        ;Check if any key
ØA9E B7       13370         OR      A               ;  code was saved
ØA9F CAB5Ø9   13380         JP      Z,NOCHAR        ;Ret if none
ØAA2 CB53     13390         BIT     2,E             ;Shift key down?
ØAA4 37       13400         SCF                     ;Init carry
ØAA5 2ØØ4     13410         JR      NZ,SPECL        ;Ret if CTRL
ØAA7 3F       13420         CCF
ØAA8 CB7F     13430 DVREXIT BIT     7,A             ;Z-flag set on non-CLEAR
ØAAA C8       13440         RET     Z               ;Go if not CLEAR+key
ØAAB F5       13450 SPECL   PUSH    AF              ;Save code
ØAAC 211EØ9   13460 $L13    LD      HL,SPCLTB       ;Special char table
ØAAF CBBF     13470         RES     7,A             ;Turn off "CLEAR"
ØAB1 Ø15BØ5   13480         LD      BC,5<8!5BH      ;5 chars, starting char
ØAB4 3ØØ1     13490         JR      NC,$+3          ;  if not CTRL
```

Keyboard Driver

```
ØAB6 Ø5       135ØØ          DEC     B               ;  else only 4
ØAB7 BE       1351Ø SPCLLP   CP      (HL)            ;Is this it?
ØAB8 2812     1352Ø          JR      Z,HIT           ;Go if so
ØABA EE1Ø     1353Ø          XOR     1ØH             ;Flip shift state
ØABC BE       1354Ø          CP      (HL)            ;Is that it?
ØABD 28ØB     1355Ø          JR      Z,HITWS         ;Go if so
ØABF EE1Ø     1356Ø          XOR     1ØH             ;Flip back
ØAC1 23       1357Ø          INC     HL              ;Bump specl table ptr
ØAC2 ØC       1358Ø          INC     C               ;Bump "convert to" char
ØAC3 1ØF2     1359Ø          DJNZ    SPCLLP          ;Loop through table
ØAC5 F1       136ØØ          POP     AF              ;Not found in table
ØAC6 38ØA     1361Ø          JR      C,CKCTL2        ;Ck CTL for C-flag
ØAC8 BF       1362Ø CKCTL1   CP      A               ;Set Z-flag
ØAC9 C9       1363Ø          RET
ØACA CBE9     1364Ø HITWS    SET     5,C             ;Move to LC set
ØACC F1       1365Ø HIT      POP     AF              ;Restore orig char
ØACD 79       1366Ø          LD      A,C             ;Load converted one
ØACE 3ØF8     1367Ø CKCTL    JR      NC,CKCTL1       ;Go if ctl key not down
ØADØ E61F     1368Ø          AND     1FH             ;Force ctl code
ØAD2 BF       1369Ø CKCTL2   CP      A               ;Set Z-flag
ØAD3 37       137ØØ          SCF                     ;Set C-flag for CTRL
ØAD4 C9       1371Ø          RET
              1372Ø ;
              1373Ø ;        Check the type ahead buffer for any character
              1374Ø ;
              1375Ø *MOD
              1376Ø TYPAHD
ØAD5 CD17Ø8   1377Ø          CALL    ENADIS_DO_RAM   ;Bring up Keyboard ram
ØAD8 218ØFF   1378Ø          LD      HL,TYPBUF       ;P/u start of type buffer
ØADB 36FF     1379Ø          LD      (HL),ØFFH       ;Turn off type ahead
ØADD 381D     138ØØ          JR      C,$M1           ;Go on @GET
ØADF 2842     1381Ø          JR      Z,TYPON         ;No PUT to *KI
ØAE1 FEØ3     1382Ø          CP      3               ;CTL 3 function?
ØAE3 CA65ØB   1383Ø          JP      Z,CLRTYP        ;Clear buffer if so
ØAE6 3C       1384Ø          INC     A
ØAE7 28Ø3     1385Ø          JR      Z,CTLFF         ;Go if CTL 255 function
ØAE9 AF       1386Ø          XOR     A               ;Nothing done, No error
ØAEA 1837     1387Ø          JR      TYPON
              1388Ø ;
              1389Ø ;        Handle CTL-255 - scan keyboard into user rowbuf
              139ØØ ;
              1391Ø CTLFF
ØAEC 21Ø1F4   1392Ø          LD      HL,KBØ          ;Start of keyboard image
ØAEF Ø6Ø8     1393Ø          LD      B,8             ;Do 8 rows
ØAF1 7E       1394Ø $MØ      LD      A,(HL)          ;P/u the image
ØAF2 FD77ØØ   1395Ø          LD      (IY),A          ;  and xfer to user buffer
ØAF5 FD23     1396Ø          INC     IY
ØAF7 CB15     1397Ø          RL      L
ØAF9 1ØF6     1398Ø          DJNZ    $MØ
ØAFB C9       1399Ø          RET
              14ØØØ ;
ØAFC E5       14Ø1Ø $M1      PUSH    HL
ØAFD 23       14Ø2Ø          INC     HL              ;Bump to PUT pointer
ØAFE 7E       14Ø3Ø          LD      A,(HL)          ;  & pick it up
ØAFF 23       14Ø4Ø          INC     HL              ;Bump to GET pointer
ØBØØ BE       14Ø5Ø          CP      (HL)            ;The same?
ØBØ1 281C     14Ø6Ø          JR      Z,$M4           ;Go if so
ØBØ3 E5       14Ø7Ø          PUSH    HL              ;Save pointer to GETPTR
ØBØ4 5E       14Ø8Ø          LD      E,(HL)          ;P/u offset to buffer
```

Keyboard Driver

```
ØBØ5 23       14Ø9Ø          INC     HL              ;Pt to buffer start
ØBØ6 16ØØ     141ØØ          LD      D,Ø             ;Add offset to start
ØBØ8 19       1411Ø          ADD     HL,DE           ;  to point to char posn
ØBØ9 46       1412Ø          LD      B,(HL)          ;GET the stored char
ØBØA E1       1413Ø          POP     HL              ;Rcvr GETPTR
ØBØB 34       1414Ø          INC     (HL)            ;Bump by 1 for char
ØBØC 3E5Ø     1415Ø          LD      A,8Ø            ;Check for >8Ø
ØBØE BE       1416Ø          CP      (HL)            ;  after INC
ØBØF 3ØØ2     1417Ø          JR      NC,$M2          ;Go if not at end
ØB11 36ØØ     1418Ø          LD      (HL),Ø          ;Reset to start of buf
ØB13 7E       1419Ø $M2      LD      A,(HL)          ;If we emptied the
ØB14 2B       142ØØ          DEC     HL              ;  type-ahead buffer,
ØB15 BE       1421Ø          CP      (HL)            ;  update KFLAG$
ØB16 CC6AØB   1422Ø          CALL    Z,R7KFLG        ;Reset bit-7 if empty
ØB19 E1       1423Ø          POP     HL              ;Pointed to & get switch
ØB1A 36ØØ     1424Ø          LD      (HL),Ø          ;Turn type back on
ØB1C 78       1425Ø          LD      A,B             ;Transfer char/flag
ØB1D BF       1426Ø          CP      A               ;Set flag "Z"
ØB1E C9       1427Ø          RET
              1428Ø ;
              1429Ø ;        No character in type ahead buffer - get from kbd
              143ØØ ;
ØB1F CD83Ø9   1431Ø $M4      CALL    KISCAN          ;Call keyboard driver
ØB22 E1       1432Ø          POP     HL              ;Rcvr switch
ØB23 36ØØ     1433Ø TYPON    LD      (HL),Ø          ;Type ahead back on
ØB25 C9       1434Ø          RET
              1435Ø ;
              1436Ø ;        Type ahead task 1Ø - scans keyboard & saves key
              1437Ø ;
ØB26 28ØB     1438Ø TYPTSK$  DW      $M5             ;Task entry for processor
ØB28 3A6DØØ   1439Ø $M5      LD      A,(DFLAG$)      ;If type-ahead suppressed
ØB2B E6Ø2     144ØØ          AND     2H              ;  then return
ØB2D C8       1441Ø          RET     Z
ØB2E CD17Ø8   1442Ø          CALL    ENADIS_DO_RAM   ;Bring up the keyboard
ØB31 218ØFF   1443Ø          LD      HL,TYPBUF       ;P/u type switch
ØB34 7E       1444Ø          LD      A,(HL)          ;If previous driver is
ØB35 B7       1445Ø          OR      A               ;  currently executing,
ØB36 CØ       1446Ø          RET     NZ              ;  do not stack more keys
ØB37 23       1447Ø          INC     HL              ;Bump to PUTPTR
ØB38 E5       1448Ø          PUSH    HL              ;  & save it
ØB39 CD83Ø9   1449Ø KIHOOK   CALL    KISCAN          ;  and scan for a character
ØB3C E1       145ØØ          POP     HL
ØB3D CØ       1451Ø          RET     NZ              ;Ret if no char
ØB3E F5       1452Ø          PUSH    AF              ;  else xfer char
ØB3F C1       1453Ø          POP     BC              ;  & flag to BC
ØB4Ø FE8Ø     1454Ø          CP      8ØH             ;Check for <BREAK>
ØB42 F5       1455Ø          PUSH    AF
ØB43 E5       1456Ø          PUSH    HL
ØB44 CC66ØB   1457Ø          CALL    Z,$M6           ;If so clear type buf
ØB47 E1       1458Ø          POP     HL              ;Restore
ØB48 F1       1459Ø          POP     AF
ØB49 FECØ     146ØØ          CP      ØCØH            ;If CLEAR @, reset keybuf
ØB4B 2819     1461Ø          JR      Z,$M6
ØB4D 5E       1462Ø          LD      E,(HL)          ;P/u PUTPTR & compare
ØB4E 7B       1463Ø          LD      A,E             ;GETPTR
ØB4F 23       1464Ø          INC     HL
ØB5Ø BE       1465Ø          CP      (HL)
ØB51 2821     1466Ø          JR      Z,$M8           ;Jump if key buffer empty
ØB53 3A2CØØ   1467Ø          LD      A,(TIMER$)      ;Check if we expired the
```

Keyboard Driver

```
ØB56 DD86Ø3   14680          ADD     A,(IX+RPTRATE)  ;  time interval between
ØB59 DDBEØ1   14690          CP      (IX+1)          ;  repeating keys
ØB5C 2Ø12     14700          JR      NZ,$M7          ;Go if time not up
ØB5E DD86Ø3   14710          ADD     A,(IX+RPTRATE)  ;Re-adjust time check so
ØB61 DD77Ø1   14720          LD      (IX+1),A        ;  we don't repeat in
ØB64 C9       14730          RET                     ;  type-ahead task
              14740 ;
              14750 ;        CLEAR @ control key entered, clear the buffer
              14760 ;
ØB65 23       14770 CLRTYP   INC     HL              ;Bump to PUT pointer
ØB66 AF       14780 $M6      XOR     A
ØB67 77       14790          LD      (HL),A          ;1st PUT is loc'n Ø
ØB68 23       14800          INC     HL              ;Pt to GETPTR
ØB69 77       14810          LD      (HL),A          ;1st GET is loc'n Ø
ØB6A 217400   14820 R7KFLG   LD      HL,KFLAG$       ;Show buffer empty
ØB6D CBBE     14830          RES     7,(HL)
ØB6F C9       14840          RET
              14850 ;
              14860 ;        Char to stuff - check if buffer will overflow
              14870 ;
ØB7Ø 7B       14880 $M7      LD      A,E             ;P/u current PUT pointer
ØB71 3C       14890          INC     A               ;If the next loc'n wraps
ØB72 BE       14900          CP      (HL)            ;  to the GET loc'n,
ØB73 C8       14910          RET     Z               ;  don't permit overrun
ØB74 E5       14920 $M8      PUSH    HL              ;Save ptr to GETPTR
ØB75 23       14930          INC     HL              ;Pt to start of keybuf
ØB76 16ØØ     14940          LD      D,Ø             ;  & calculate PUT loc'n
ØB78 19       14950          ADD     HL,DE
ØB79 7Ø       14960          LD      (HL),B          ;Store the char
ØB7A 217400   14970          LD      HL,KFLAG$       ;Show type buffer
ØB7D CBFE     14980          SET     7,(HL)          ;  is not empty
ØB7F E1       14990          POP     HL              ;Rcvr ptr to GETPTR
ØB8Ø 2B       15000          DEC     HL              ;Backup to PUTPTR
ØB81 34       15010          INC     (HL)            ;Bump past the char
ØB82 3E5Ø     15020          LD      A,8Ø            ;Check for >8Ø
ØB84 BE       15030          CP      (HL)
ØB85 DØ       15040          RET     NC              ;Back if not over 8Ø
ØB86 72       15050          LD      (HL),D          ;  else reset to 1st
ØB87 C9       15060          RET                     ;  position in buf (Ø)
              15070 ;
              15080 ;        Type ahead buffer area
              15090 ;
FF8Ø          15100 TYPBUF
              15110 ;
              15120 ;        TYPBUF+Ø = On/Off Flag
              15130 ;        TYPBUF+1 = Storage pointer
              15140 ;        TYPBUF+2 = Retrieve pointer
              15150 ;        TYPBUF+3 = Start of actual buffer
              15160 ;
ØB87          15170 KILAST   EQU     $-1
              Ø1580          ENDIF
              Ø1590          IF      @GERMAN
              Ø1600 FREN     EQU     ØØ
              Ø1610 GERM     EQU     -1
              Ø1630          ENDIF
              Ø1640          IF      @FRENCH
              Ø1650 FREN     EQU     -1
              Ø1660 GERM     EQU     ØØ
              Ø1680          ENDIF
```

Keyboard Driver

```
ØB88          Ø169Ø *GET    DODVR:3                 ;Video driver
              1518Ø ;DODVR/ASM - LS-DOS 6.2
ØB88          1519Ø         SUBTTL  '<Video Driver>'
```

Video Driver

```
                15210 *MOD
ØØ84            15220 @OPREG                            ;Mem mgt & video control
ØØ88            15230 CRTCADD                           ;CRTC address port
ØØ89            15240 CRTCDAT                           ;CRTC data port
ØØ5Ø            15250 LINESIZ EQU     8Ø
ØØ18            15260 NUMROWS EQU     24
FFBØ            15270 NEGLINE EQU     -LINESIZ
Ø78Ø            15280 CRTSIZE EQU     LINESIZ*NUMROWS
Ø8ØØ            15290 RAMSIZE EQU     2Ø48
F8ØØ            15300 CRTBGN$
FF7F            15310 CRTEND  EQU     CRTBGN$+CRTSIZE-1
                15320 ;
                15330 ;       Driver entry point
                15340 ;
ØB88 1812       15350 DODVR   JR      DOBGN             ;Branch around linkage
ØB8A ØØØE       15360         DW      DOEND             ;Last memory location used
ØB8C Ø3         15370         DB      3,'$DO'
     24 44 4F
ØB9Ø 1ØØ2       15380         DW      DODCB$            ;DCB used
ØB92 ØØØØ       15390         DW      Ø                 ;Reserved
ØB94            15400 DODATA$ EQU     $
ØØØØ            15410 DO_MASK EQU     $-DODATA$
ØØØ7            15420 SCRPROT EQU     7                 ;Bits Ø-2: scroll protect
ØØØ3            15430 TABS    EQU     3                 ;Bit 3: Ø=tabs, 1=chars
ØØØ4            15440 CTL     EQU     4                 ;Bit 4, display controls
                15450         IF      @USA
ØB94 ØØ         15460         DB      Ø                 ;Tab/Spec, Scroll protect
                15470         ENDIF
                15480         IF      @INTL
                15490         DB      Ø8                ;Space compression off
                15500         ENDIF
ØB95 ØØF8       15510 CURSOR  DW      CRTBGN$
ØB97 2Ø         15520 CRSAVE  DB      2ØH               ;Character under cursor
ØB98 5F         15530 CRSCHAR DB      '_'               ;Cursor character
                15540 ;
                15550 ;       Entry from SVC 15, @VDCTL
                15560 ;
ØB99 C342ØD     15570 @VDCTL  JP      @_VDCTL
                15580 ;
                15590 ;       Continue regular driver functions
                15600 ;
ØB9C DD2194ØB   15610 DOBGN   LD      IX,DODATA$
ØBAØ CD17Ø8     15620         CALL    ENADIS_DO_RAM     ;Bring up the video RAM
ØBA3 DAB3ØD     15630         JP      C,$NØ             ;Go on 'GET' request
ØBA6 CDB3ØD     15640         CALL    $NØ               ;Handle cursor
ØBA9 C5         15650         PUSH    BC                ;Need to save C
ØBAA 79         15660         LD      A,C               ;Get char to display
ØBAB DDCBØØ66   15670         BIT     CTL,(IX+DO_MASK)  ;Display controls set?
ØBAF 2ØØ9       15680         JR      NZ,$N1A           ;Go if so
ØBB1 B7         15690         OR      A                 ;Char a Ø?
ØBB2 CA9FØC     15700         JP      Z,TGGLCTL         ;Switch Bit CTL if so
ØBB5 FE2Ø       15710         CP      2ØH               ;Video control char?
ØBB7 DA44ØC     15720         JP      C,DO_CONTROL      ;Go if so
ØBBA FECØ       15730 $N1A    CP      ØCØH              ;Tab or special?
ØBBC 38Ø6       15740         JR      C,DONORM          ;Go on normal characters
                15750 ;
                15760 ;       Character is => ØCØH
                15770 ;
ØBBE DDCBØØ5E   15780         BIT     TABS,(IX+DO_MASK) ;Tabs or spec chars
ØBC2 2826       15790         JR      Z,DO_TABS         ;Go if video tabs
```

Video Driver

```
                    15ØØ ;
                    1581Ø ;       Character is not tab expansion - do it
                    1582Ø ;
ØBC4 CDB8ØC         1583Ø DONORM  CALL    DO_DSPCHAR       ;Display the char
ØBC7 DDCBØØA6       1584Ø         RES     CTL,(IX+DO_MASK) ;Turn off CTL bit
ØBCB C1             1585Ø DO_RET  POP     BC               ;Get orig char
ØBCC F3             1586Ø DO_RET1 DI                       ;Disable intr
ØBCD 3A97ØB         1587Ø         LD      A,(CRSAVE)       ;If a cursor is on, then
ØBDØ B7             1588Ø         OR      A                ;  we need to save the
ØBD1 281Ø           1589Ø         JR      Z,$N1            ;  current char & display
ØBD3 1A             1590Ø         LD      A,(DE)           ;  the cursor character
ØBD4 3297ØB         1591Ø         LD      (CRSAVE),A       ;Save current char
ØBD7 3A7FØØ         1592Ø         LD      A,(VFLAG$)       ;Allow tasker to blink
ØBDA CBBF           1593Ø         RES     7,A
ØBDC 327FØØ         1594Ø         LD      (VFLAG$),A
ØBDF 3A98ØB         1595Ø         LD      A,(CRSCHAR)      ;P/u cusor character
ØBE2 12             1596Ø         LD      (DE),A           ;Put it on the screen
ØBE3 ED5395ØB       1597Ø $N1     LD      (CURSOR),DE      ;Update cursor position
ØBE7 BF             1598Ø         CP      A                ;Clear status
ØBE8 79             1599Ø         LD      A,C              ;Restore the char
ØBE9 C9             16ØØØ         RET
                    1601Ø ;
                    1602Ø ;       Perform a tab expansion {CØH-FFH}
                    1603Ø ;
                    1604Ø DO_TABS
ØBEA D6CØ           1605Ø         SUB     ØCØH             ;Compute spaces
ØBEC 28DD           1606Ø         JR      Z,DO_RET         ;Forget it if TAB(Ø)
ØBEE 47             1607Ø         LD      B,A              ;Display requested
ØBEF ØE2Ø           1608Ø $N2     LD      C,' '            ;  number of spaces
ØBF1 CDB8ØC         1609Ø         CALL    DO_DSPCHAR
ØBF4 1ØF9           161ØØ         DJNZ    $N2
ØBF6 18D3           1611Ø         JR      DO_RET
                    1612Ø ;
                    1613Ø ;       Routine to move the cursor to begin of line {29}
                    1614Ø ;
                    1615Ø CRSBOL
ØBF8 EB             1616Ø         EX      DE,HL            ;Cursor addr to HL
ØBF9 CDF4ØD         1617Ø         CALL    ADDR1            ;Find row,col
ØBFC 6F             1618Ø         LD      L,A              ;Set col to start
ØBFD C3DØØD         1619Ø         JP      ROWCOL_2_ADDR    ;Calc address of BOL
                    162ØØ ;
                    1621Ø ;       Routines to turn on/off the cursor {14/15}
                    1622Ø ;
ØCØØ 1A             1623Ø CRSON   LD      A,(DE)           ;Get screen character
ØCØ1 3297ØB         1624Ø CRSOFF  LD      (CRSAVE),A       ;Save zero or CRT char
ØCØ4 C9             1625Ø         RET
                    1626Ø ;
                    1627Ø ;       Routine moves cursor to start of video page {28}
                    1628Ø ;        set to 8Ø column, and turns off inverse video
                    1629Ø ;
                    163ØØ CRSHOME
ØCØ5 11ØØF8         1631Ø         LD      DE,CRTBGN$       ;Home the cursor
ØCØ8 3A76ØØ         1632Ø         LD      A,(MODOUT$)      ;P/u the mask &
ØCØB E6FB           1633Ø         AND     ØFBH             ;  set to 8Ø cpl
ØCØD CDB2ØC         1634Ø         CALL    SETMOD
ØC1Ø 187A           1635Ø         JR      DO_INVERT_DIS    ;Set to normal video
                    1636Ø ;
                    1637Ø ;       Routine to backspace & erase cursor {Ø8}
                    1638Ø ;
```

Video Driver

```
                16390 BACKSPA
ØC12 CD1BØC     16400         CALL    CRSBKSP         ;Backspace the cursor
ØC15 C8         16410         RET     Z               ;If not at start,
ØC16 ØE2Ø       16420         LD      C,' '           ;  put a space at
ØC18 C3CAØD     16430         JP      PUT_@           ;  at the new loc'n
                16440 ;
                16450 ;       Routine to backspace the cursor {24}
                16460 ;
                16470 CRSBKSP
ØC1B 3A76ØØ     16480         LD      A,(MODOUT$)     ;If double width chars,
ØC1E E6Ø4       16490         AND     4               ;  need to do twice
ØC2Ø C423ØC     16500         CALL    NZ,$+3
ØC23 21ØØF8     16510         LD      HL,CRTBGN$      ;See if at home position
ØC26 ED52       16520         SBC     HL,DE           ;  prior to adjusting
ØC28 C8         16530         RET     Z
ØC29 1B         16540         DEC     DE              ;Decrement the cursor pos
ØC2A C9         16550         RET
                16560 ;
                16570 ;       Routine to move the cursor up one line {27}
                16580 ;
                16590 CRSUP
ØC2B 21BØFF     16600         LD      HL,NEGLINE      ;Move up one line
ØC2E 18Ø3       16610         JR      MOVCRS
                16620 ;
                16630 ;       Routine to move the cursor down one line {26}
                16640 ;
                16650 CRSDOWN
ØC3Ø 215ØØØ     16660         LD      HL,LINESIZ      ;Add the line length
ØC33 19         16670 MOVCRS  ADD     HL,DE           ;  to the current pos
ØC34 7C         16680         LD      A,H             ;Make sure we did not
ØC35 FEF8       16690         CP      CRTBGN$<-8      ;  go over the top
ØC37 D8         16700         RET     C
ØC38 EB         16710         EX      DE,HL           ;  & switch back to DE
ØC39 1B         16720         DEC     DE              ;Adjust for fall thru
ØC3A C3C3ØC     16730         JP      CRSFRWØ
                16740 ;
                16750 ;       Set to 4Ø cpl mode {23}
                16760 ;
ØC3D 3A76ØØ     16770 SET4Ø   LD      A,(MODOUT$)     ;Get image of the port
ØC4Ø F6Ø4       16780         OR      Ø4H             ;Merge in 4Ø cpl bit
ØC42 186E       16790         JR      SETMOD
                16800 ;
                16810 ;       Routines to parse control functions
                16820 ;
                16830 DO_CONTROL
ØC44 21CBØB     16840         LD      HL,DO_RET       ;Establish RET
ØC47 E5         16850         PUSH    HL
ØC48 FEØ8       16860         CP      Ø8H             ;Backspace?
ØC4A 28C6       16870         JR      Z,BACKSPA
ØC4C FEØA       16880         CP      ØAH             ;Line feed?
ØC4E 28Ø2       16890         JR      Z,$+4           ;  is same as <ENTER>
ØC5Ø D6ØD       16900         SUB     ØDH             ;Carriage return?
ØC52 CAØ2ØD     16910         JP      Z,LINFEED
ØC55 3D         16920         DEC     A               ;Cursor on?
ØC56 28A8       16930         JR      Z,CRSON
ØC58 3D         16940         DEC     A               ;Cursor off?
ØC59 28A6       16950         JR      Z,CRSOFF
ØC5B 3D         16960         DEC     A               ;Reverse video?
ØC5C 282B       16970         JR      Z,DO_INVERT_ENA
```

Video Driver

```
ØC5E 3D       16980          DEC     A
ØC5F 283A     16990          JR      Z,DO_INVERT_OFF
ØC61 D6Ø4     17000          SUB     4               ;Swap tab/alternate?
ØC63 2841     17010          JR      Z,TGGLTAB
ØC65 3D       17020          DEC     A               ;Special/alternate?
ØC66 2845     17030          JR      Z,TGGLALT
ØC68 3D       17040          DEC     A               ;4Ø cpl?
ØC69 28D2     17050          JR      Z,SET4Ø
ØC6B 3D       17060          DEC     A               ;Cursor backspace?
ØC6C 28AD     17070          JR      Z,CRSBKSP
ØC6E 3D       17080          DEC     A               ;Cursor forward?
ØC6F 284A     17090          JR      Z,CRSFRWD
ØC71 3D       17100          DEC     A               ;Cursor down?
ØC72 28BC     17110          JR      Z,CRSDOWN
ØC74 3D       17120          DEC     A               ;Cursor up?
ØC75 28B4     17130          JR      Z,CRSUP
ØC77 3D       17140          DEC     A               ;Cursor home?
ØC78 CAØ5ØC   17150          JP      Z,CRSHOME
ØC7B 3D       17160          DEC     A               ;Cursor BOL?
ØC7C CAF8ØB   17170          JP      Z,CRSBOL
ØC7F 3D       17180          DEC     A               ;Clear to EOL?
ØC8Ø CA12ØD   17190          JP      Z,CLREOL
ØC83 3D       17200          DEC     A
ØC84 CA1EØD   17210          JP      Z,CLREOF        ;Clear to end-of-frame?
ØC87 AF       17220          XOR     A               ;Clear A reg.
ØC88 C9       17230          RET
              17240 ;
              17250 ;        Routine to enable inverse video
              17260 ;
              17270 DO_INVERT_ENA
ØC89 Ø6Ø8     17280          LD      B,8             ;Set for enable
ØC8B 21       17290          DB      21H             ;Ignore next load
              17300 DO_INVERT_DIS
ØC8C Ø6ØØ     17310          LD      B,Ø
ØC8E 2A35Ø8   17320          LD      HL,(OPREG_SV_PTR)       ;Real OPREG$
ØC91 7E       17330          LD      A,(HL)          ;P/u OPREG mask
ØC92 E6F7     17340          AND     ØF7H            ;Strip bit 3
ØC94 BØ       17350          OR      B               ;Set/reset invideo bit
ØC95 77       17360          LD      (HL),A          ;  and restuff
ØC96 78       17370          LD      A,B             ;Get mode mask byte
ØC97 Ø7       17380          RLCA                    ;Rotate left 4 times to
ØC98 Ø7       17390          RLCA                    ;  make an 8 into 8ØH
ØC99 Ø7       17400          RLCA                    ;  for inverse on
ØC9A Ø7       17410          RLCA                    ;Inverse off remains Ø
              17420 DO_INVERT_OFF
ØC9B 32CBØD   17430          LD      (INVIDEO),A     ;Set the mask byte
ØC9E C9       17440          RET
              17450 ;
              17460 ;        Routine to toggle display of controls
              17470 ;
ØC9F 21CBØB   17480 TGGLCTL  LD      HL,DO_RET       ;Establish ret addr
ØCA2 E5       17490          PUSH    HL
ØCA3 3E1Ø     17500          LD      A,1ØH           ;Toggle bit 4
ØCA5 21       17510          DB      21H             ;Ignore next
              17520 ;
              17530 ;        Toggle tabs & alternate character set
              17540 ;
              17550 TGGLTAB
ØCA6 3EØ8     17560          LD      A,8             ;Toggle bit 3
```

Video Driver

```
ØCA8 DDAEØØ   1757Ø          XOR     (IX+DO_MASK)    ;P/u mask value
ØCAB 185Ø     1758Ø          JR      SETMASK
              1759Ø ;
              176ØØ ;        Toggle special & alternate character set
              1761Ø ;
              1762Ø TGGLALT
ØCAD 3A76ØØ   1763Ø          LD      A,(MODOUT$)     ;P/u port mask
ØCBØ EEØ8     1764Ø          XOR     8               ;Flip the bit
ØCB2 3276ØØ   1765Ø SETMOD   LD      (MODOUT$),A     ;Resave port mask
ØCB5 D3EC     1766Ø                                  ;  and send the byte
ØCB7 C9       1767Ø          RET
              1768Ø ;
              1769Ø ;        Display character <C> at current cursor position
              177ØØ ;
              1771Ø DO_DSPCHAR
ØCB8 CDCAØD   1772Ø          CALL    PUT_@           ;Display the char
              1773Ø ;
              1774Ø ;        Routine to perform cursor forward {25}
              1775Ø ;
              1776Ø CRSFRWD
ØCBB 3A76ØØ   1777Ø          LD      A,(MODOUT$)     ;If double width chars,
ØCBE E6Ø4     1778Ø          AND     4               ;  need to do twice
ØCCØ 28Ø1     1779Ø          JR      Z,CRSFRWØ
ØCC2 13       178ØØ          INC     DE              ;Move cursor forward
ØCC3 13       1781Ø CRSFRWØ  INC     DE
ØCC4 217FFF   1782Ø          LD      HL,CRTEND       ;Off the screen?
ØCC7 ED52     1783Ø          SBC     HL,DE
ØCC9 DØ       1784Ø          RET     NC              ;Back if not
ØCCA CD2BØC   1785Ø          CALL    CRSUP           ;Put cursor back on
ØCCD D5       1786Ø          PUSH    DE              ;Save cursor position
              1787Ø DO_SCROLL
ØCCE DD7EØØ   1788Ø          LD      A,(IX+DO_MASK)  ;Get scroll protect
ØCD1 E6Ø7     1789Ø          AND     SCRPROT
ØCD3 21ØØF8   179ØØ          LD      HL,CRTBGN$      ;Point to CRT start
ØCD6 118ØØ7   1791Ø          LD      DE,CRTSIZE      ;P/u CRT size
ØCD9 C5       1792Ø          PUSH    BC
ØCDA Ø15ØØØ   1793Ø          LD      BC,LINESIZ      ;Set line size
ØCDD 3C       1794Ø          INC     A               ;Adjust scroll protect
ØCDE Ø9       1795Ø $N4      ADD     HL,BC           ;Move logical start
ØCDF EB       1796Ø          EX      DE,HL           ;  down one line
ØCEØ B7       1797Ø          OR      A               ;  and subtract one line
ØCE1 ED42     1798Ø          SBC     HL,BC           ;  from the CRTSIZE for
ØCE3 EB       1799Ø          EX      DE,HL           ;  each protected line
ØCE4 3D       18ØØØ          DEC     A               ;Dec scroll protect
ØCE5 2ØF7     18Ø1Ø          JR      NZ,$N4          ;Loop until done
ØCE7 D5       18Ø2Ø          PUSH    DE              ;Save the move length
ØCE8 E5       18Ø3Ø          PUSH    HL              ;Save the move-from
ØCE9 ED42     18Ø4Ø          SBC     HL,BC           ;Move start back one
ØCEB EB       18Ø5Ø          EX      DE,HL           ;  line, Source =
ØCEC E1       18Ø6Ø          POP     HL              ;  start + one
ØCED C1       18Ø7Ø          POP     BC              ;Get back dest locn
ØCEE EDBØ     18Ø8Ø          LDIR                    ;Scroll unprotected
ØCFØ C1       18Ø9Ø          POP     BC              ;Recover line size
ØCF1 182C     181ØØ          JR      CLREOF1         ;Clear to EOF from DE
              1811Ø ;
              1812Ø ;        Set scroll protect value
              1813Ø ;                C = scroll protect <Ø-7>
              1814Ø ;                B = 7
              1815Ø ;                SVC = 15, @VDCTL
```

Video Driver

```
                    18160 ;
                    18170 SET_SCROLL
ØCF3 79             18180         LD      A,C             ;Get user value
ØCF4 E607           18190         AND     7               ;Make modulo 8
ØCF6 4F             18200         LD      C,A
ØCF7 3A940B         18210         LD      A,(DODATA$)     ;P/u current mask
ØCFA E6F8           18220         AND     ØF8H            ;Remove current scroll
ØCFC B1             18230         OR      C               ;Merge in the new value
ØCFD 32940B         18240 SETMASK LD      (DODATA$),A     ;  & reload mask
ØDØØ AF             18250         XOR     A               ;Z-flag return
ØDØ1 C9             18260         RET
                    18270 ;
                    18280 ;       Routine to move down one line {1Ø/13}
                    18290 ;
ØDØ2 CDF8ØB         18300 LINFEED CALL    CRSBOL          ;Move to BOL
ØDØ5 D5             18310         PUSH    DE              ;Save cursor position
ØDØ6 CD3ØØC         18320         CALL    CRSDOWN         ;Move down one line
ØDØ9 B7             18330         OR      A               ;Reset the carry flag
ØDØA 218ØFF         18340         LD      HL,CRTEND+1<    ;  & check if off of
ØDØD ED52           18350         SBC     HL,DE           ;  the screen
ØDØF 28BD           18360         JR      Z,DO_SCROLL     ;Scroll if so
ØD11 E1             18370         POP     HL              ;Discard old position
ØD12 D5             18380 CLREOL  PUSH    DE              ;Save new cursor pos
ØD13 CDF8ØB         18390         CALL    CRSBOL          ;Get start of line
ØD16 214FØØ         18400         LD      HL,79           ;Calculate end of line
ØD19 19             18410         ADD     HL,DE           ;HL = end of line
ØD1A D1             18420         POP     DE              ;DE = current position
ØD1B D5             18430         PUSH    DE
ØD1C 18Ø4           18440         JR      CLREOF2         ;Clear the line
                    18450 ;
                    18460 ;       Clear to the end of the frame
                    18470 ;
ØD1E D5             18480 CLREOF  PUSH    DE              ;Save current cursor pos
ØD1F 217FFF         18490 CLREOF1 LD      HL,CRTEND       ;Point to last RAM byte
ØD22 3ACBØD         18500 CLREOF2 LD      A,(INVIDEO)     ;P/u normal/reverse
ØD25 CBEF           18510         SET     5,A             ;  & make it a space
ØD27 12             18520         LD      (DE),A          ;Stuff the "space"
ØD28 B7             18530         OR      A               ;Reset carry for subtract
ØD29 ED52           18540         SBC     HL,DE           ;Calculate length
ØD2B 28Ø9           18550         JR      Z,CLREOF3       ;Back if at end already
ØD2D C5             18560         PUSH    BC
ØD2E 44             18570         LD      B,H             ;Xfer length to BC
ØD2F 4D             18580         LD      C,L
ØD3Ø 62             18590         LD      H,D             ;Xfer start to HL
ØD31 6B             18600         LD      L,E
ØD32 13             18610         INC     DE              ;Bump up by one
ØD33 EDBØ           18620         LDIR                    ;Propagate the space
ØD35 C1             18630         POP     BC
ØD36 D1             18640 CLREOF3 POP     DE
ØD37 C9             18650         RET
                    18660 ;
                    18670 ;       Routine to stuff the video cursor RAM address
                    18680 ;
ØD38 CDDØØD         18690 @VDCTL3 CALL    ROWCOL_2_ADDR   ;Calculate video address
ØD3B CØ             18700         RET     NZ              ;Back on error
ØD3C F3             18710         DI                      ;Disable any video tasks
ØD3D ED5395ØB       18720         LD      (CURSOR),DE     ;  until cursor is updated
ØD41 C9             18730         RET
                    18740 ;
```

Video Driver

```
                    18750 ;       Video control SVC processor
                    18760 ;
                    18770 @_VDCTL
ØD42 CD17Ø8         18780         CALL    ENADIS_DO_RAM   ;Bring up the video RAM
                    18790 ;
                    18800 ;       Test if in Task processor
                    18810 ;
ØD45 3A77ØØ         18820         LD      A,(NFLAG$)      ;P/u NFLAG$
ØD48 CB77           18830         BIT     6,A             ;Test for task process
ØD4A 2Ø15           18840         JR      NZ,VDCTL        ;If so skip setup
                    18850 ;
                    18860 ;       HANDLES @VDCTL screen set up for normal use
                    18870 ;
ØD4C D5             18880         PUSH    DE
ØD4D CDB3ØD         18890         CALL    $NØ             ;Normalize character at cursor
ØD5Ø D1             18900         POP     DE              ;Recover value
ØD51 D5             18910         PUSH    DE
ØD52 CD61ØD         18920         CALL    VDCTL           ;Do function request
ØD55 F5             18930         PUSH    AF              ;Save the error status
ØD56 F3             18940         DI                      ;Stop video tasks tempy
ØD57 ED5B95ØB       18950         LD      DE,(CURSOR)
ØD5B CDCCØB         18960         CALL    DO_RET1         ;Normalize screen and cursor
ØD5E F1             18970         POP     AF
ØD5F D1             18980         POP     DE
ØD6Ø C9             18990         RET
                    19000 ;
ØD61 3EØ9           19010 VDCTL   LD      A,9             ;Check for VIDLINE,
ØD63 B8             19020         CP      B               ;  function 9
ØD64 2825           19030         JR      Z,VIDLIN
ØD66 3E2B           19040         LD      A,43            ;Prepare for user ERROR
ØD68 Ø5             19050         DEC     B
ØD69 2843           19060         JR      Z,GET_@_ROWCOL  ;<C> from row-H, col-L
ØD6B Ø5             19070         DEC     B
ØD6C 2858           19080         JR      Z,PUT_@_ROWCOL  ;<C> to row-H, col-L
ØD6E Ø5             19090         DEC     B
ØD6F 28C7           19100         JR      Z,@VDCTL3       ;Set cursor to H,L
ØD71 Ø5             19110         DEC     B
ØD72 287D           19120         JR      Z,ADDR_2_ROWCOL ;Cursor row,col to H,L
ØD74 1100F8         19130         LD      DE,CRTBGN$      ;Init to start of video
ØD77 Ø5             19140         DEC     B
ØD78 282D           19150         JR      Z,VIDMOV1       ;User RAM to video
ØD7A Ø5             19160         DEC     B
ØD7B 2822           19170         JR      Z,VIDMOVE       ;Video RAM to user
ØD7D Ø5             19180         DEC     B
ØD7E CAF3ØC         19190         JP      Z,SET_SCROLL    ;Set scroll protect
ØD81 Ø5             19200         DEC     B
ØD82 CØ             19210         RET     NZ              ;Return if bad request
                    19220 ;
                    19230 ;       Establish cursor character
                    19240 ;
ØD83 E5             19250         PUSH    HL
ØD84 2198ØB         19260         LD      HL,CRSCHAR      ;Point to cursor char storage
ØD87 7E             19270         LD      A,(HL)          ;P/u current cursor character
ØD88 71             19280         LD      (HL),C          ;  & update with new one
ØD89 E1             19290         POP     HL
ØD8A C9             19300         RET
                    19310 ;
                    19320 ;       VIDLIN routine function - 9 in register B
                    19330 ;
```

Video Driver

```
ØD8B 2EØØ      19340 VIDLIN  LD      L,Ø             ;Always starts at col Ø
ØD8D D5        19350         PUSH    DE              ;Save user buffer
ØD8E CDDØØD    19360         CALL    ROWCOL_2_ADDR   ;Get address to DE
ØD91 E1        19370         POP     HL              ;Recover user buffer
ØD92 CØ        19380         RET     NZ              ;Quit on bad address
ØD93 ØC        19390         INC     C               ;Check direction
ØD94 ØD        19400         DEC     C               ;If Z then to screen
ØD95 28Ø1      19410         JR      Z,MOVLIN        ;Set to go
ØD97 EB        19420         EX      DE,HL           ;Reverse direction
ØD98 Ø15ØØØ    19430 MOVLIN  LD      BC,LINESIZ      ;Set line size
ØD9B EDBØ      19440         LDIR                    ;Move it
ØD9D AF        19450         XOR     A               ;Z on RET
ØD9E C9        19460         RET
               19470 ;
               19480 ;       Routine to move video RAM
               19490 ;
ØD9F 7C        19500 VIDMOVE LD      A,H             ;Check on user buffer
ØDAØ C6Ø8      19510         ADD     A,8             ;  not above X'F8ØØ' &
ØDA2 FE2C      19520         CP      24H+8           ;  not below X'24ØØ'
ØDA4 3847      19530         JR      C,PERR
ØDA6 EB        19540         EX      DE,HL           ;Xchng user buffer,screen
ØDA7 Ø18ØØ7    19550 VIDMOV1 LD      BC,CRTSIZE      ;Set for full screen xfer
ØDAA EDBØ      19560         LDIR
ØDAC BF        19570         CP      A               ;Set Z flag
ØDAD C9        19580         RET
               19590 ;
               19600 ;       Routine to get the character at row,col
               19610 ;
               19620 GET_@_ROWCOL
ØDAE CDDØØD    19630         CALL    ROWCOL_2_ADDR   ;Get Address of req
ØDB1 1A        19640         LD      A,(DE)          ;P/u the character
ØDB2 C9        19650         RET                     ;Back on error or no error
               19660 ;
               19670 ;       Routine to halt blinking cursor & restore char
               19680 ;
ØDB3 E5        19690 $NØ     PUSH    HL
ØDB4 217FØØ    19700         LD      HL,VFLAG$
ØDB7 CBFE      19710         SET     7,(HL)          ;Disable blinking cursor
ØDB9 E1        19720         POP     HL
ØDBA ED5B95ØB  19730         LD      DE,(CURSOR)     ;Get cursor pos in DE
ØDBE 3A97ØB    19740         LD      A,(CRSAVE)      ;P/u saved character
ØDC1 B7        19750         OR      A               ;If one is saved, put
               19760                                 ;  it on screen, else
ØDC2 2ØØ9      19770         JR      NZ,PUTA@DE      ;  ignore it
ØDC4 1A        19780         LD      A,(DE)          ;Cursor not ON but get
ØDC5 C9        19790         RET                     ;  character anyway
               19800 ;
               19810 ;       Routine to put a character at row,col
               19820 ;
               19830 PUT_@_ROWCOL
ØDC6 CDDØØD    19840         CALL    ROWCOL_2_ADDR   ;Get address of req
ØDC9 CØ        19850         RET     NZ              ;Back on error
ØDCA 3EØØ      19860 PUT_@   LD      A,Ø             ;Merge in reverse video
ØDCB           19870 INVIDEO EQU     $-1
ØDCC B1        19880         OR      C
ØDCD 12        19890 PUTA@DE LD      (DE),A          ;Put the character
ØDCE BF        19900         CP      A               ;Set Z-flag for return
ØDCF C9        19910         RET
               19920 ;
```

Video Driver

```
                    19930 ;       Routine to calculate cursor position from row,col
                    19940 ;
                    19950 ROWCOL_2_ADDR
ØDDØ 3E4F           19960         LD      A,79
ØDD2 BD             19970         CP      L
ØDD3 3818           19980         JR      C,PERR          ;Error if > 79
ØDD5 7C             19990         LD      A,H             ;P/u row number
ØDD6 FE18           20000         CP      24
ØDD8 3Ø13           20010         JR      NC,PERR         ;Error if > 23
ØDDA E5             20020         PUSH    HL
ØDDB C5             20030         PUSH    BC
ØDDC 4D             20040         LD      C,L             ;Save column
ØDDD Ø6F8           20050         LD      B,CRTBGN$<-8    ;Set to start of DO RAM
ØDDF 215ØØØ         20060         LD      HL,LINESIZ
ØDE2 CDC9Ø6         20070         CALL    @MUL16          ;Rows * line size
ØDE5 65             20080         LD      H,L             ;Shift to HL
ØDE6 6F             20090         LD      L,A
ØDE7 Ø9             20100         ADD     HL,BC           ;Add in col & RAM start
ØDE8 EB             20110         EX      DE,HL           ;Address to DE
ØDE9 C1             20120         POP     BC
ØDEA E1             20130         POP     HL
ØDEB AF             20140         XOR     A               ;Set Z flag
ØDEC C9             20150         RET
ØDED 3E2B           20160 PERR    LD      A,43            ;SVC parameter error
ØDEF B7             20170         OR      A               ;Set NZ condition
ØDFØ C9             20180         RET
                    20190 ;
                    20200 ;       Routine to get row,col of video cursor
                    20210 ;
                    20220 ADDR_2_ROWCOL
ØDF1 2A95ØB         20230         LD      HL,(CURSOR)     ;Get addr in HL
ØDF4 7C             20240 ADDR1   LD      A,H             ;Make address relative
ØDF5 E6Ø7           20250         AND     7               ;  to origin Ø
ØDF7 67             20260         LD      H,A
ØDF8 3E5Ø           20270         LD      A,LINESIZ       ;Set divisor
ØDFA CDE3Ø6         20280         CALL    @DIV16
ØDFD 65             20290         LD      H,L             ;Row to register H
ØDFE 6F             20300         LD      L,A             ;Column to register L
ØDFF AF             20310         XOR     A               ;Set zero return code
ØEØØ C9             20320         RET
ØEØØ                20330 DOEND   EQU     $-1
ØEØ1                01700 *GET    PRDVR:3                 ;Printer driver & filter
                    20350 ;PRDVR/ASM  -  LS-DOS 6.2
ØEØ1                20360         SUBTTL  '<Printer Driver>'
```

Printer Driver

```
                  20380 *MOD
ØØF8              20390 PRPORT
                  20400 ;
                  20410 ;       PR driver entry point
                  20420 ;       It passes X'ØØ'-X'FF'
                  20430 ;       Unless INTL version
                  20440 ;
ØE01 18ØA         20450 PRDVR   JR      PRBGN           ;Branch around linkage
ØE03 3CØE         20460         DW      PREND           ;Last byte used
ØE05 Ø3           20470         DB      3,'$PR'
     24 5Ø 52
ØE09 18Ø2         20480         DW      PRDCB$          ;Pointer to its DCB
ØE0B ØØØØ         20490         DW      Ø               ;Reserved
                  20500 ;
                  20510 ;       Driver code
                  20520 ;
ØE0D 28ØA         20530 PRBGN   JR      Z,$02           ;Go if output
ØE0F 38Ø4         20540         JR      C,$01           ;Go if input req
                  20550 ;
                  20560 ;       Character CTL request
                  20570 ;
ØE11 79           20580         LD      A,C             ;If CTL Ø, return
ØE12 B7           20590         OR      A               ;  status else
ØE13 2821         20600         JR      Z,$04           ;  treat as a Get
                  20610 ;
                  20620 ;       Character GET request
                  20630 ;
ØE15 F6FF         20640 $01     OR      ØFFH            ;Set nz
ØE17 2F           20650         CPL                     ;  & A=Ø        to show
ØE18 C9           20660         RET                     ;  no char available
                  20670 ;
                  20680 ;       Character PUT request
                  20690 ;
ØE19 11DØØ7       20700 $02     LD      DE,2ØØØ         ;Check status 2ØØØ times
ØE1C CD36ØE       20710 $02A    CALL    $04             ;PR ready?
ØE1F 2811         20720         JR      Z,$03           ;Go if so
                  20730 ;
                  20740 ;       Ten second timout delay loop
                  20750 ;
ØE21 C5           20760         PUSH    BC              ;Printer was not ready
ØE22 Ø154Ø1       20770         LD      BC,34Ø
ØE25 CD82Ø3       20780         CALL    PAUSE@          ;Delay a bit
ØE28 C1           20790         POP     BC
ØE29 1B           20800         DEC     DE              ;Time up?
ØE2A 7A           20810         LD      A,D
ØE2B B3           20820         OR      E
ØE2C 2ØEE         20830         JR      NZ,$02A         ;Nope, continue check
ØE2E 3EØ8         20840         LD      A,8             ;Device not avail...
ØE3Ø B7           20850         OR      A               ;Set NZ condition
ØE31 C9           20860         RET
ØE32              20870 $03     EQU     $
                  20880 ;
                  20890         IF      @INTL
                  20900         LD      A,(IFLAG$)
                  20910         BIT     6,A             ;Special DMP PR?
                  20920         ENDIF
                  20930 ;
ØE32 79           20940         LD      A,C
                  20950 ;
                  20960         IF      @INTL
```

Printer Driver

```
                2097Ø           JR      Z,PVAL3
                2098Ø           CP      ØCØH            ;Values CØ-FF (-2ØH)
                2099Ø           JR      C,PVAL2         ;Go if less
                21ØØØ           SUB     2ØH             ;Shift to European chars
                21Ø1Ø           JR      PVAL3
                21Ø2Ø PVAL2     CP      ØAØH            ;AØ-BF (+4ØH)
                21Ø3Ø           JR      C,PVAL3         ;Go if less
                21Ø4Ø           ADD     A,4ØH           ;Shift to graphics
                21Ø5Ø           ENDIF
                21Ø6Ø ;
ØE33 D3F8       21Ø7Ø PVAL3                             ;Put out char
                21Ø8Ø ;
                21Ø9Ø           IF      @INTL
                211Ø0           LD      A,C             ;Restore original
                2111Ø           CP      A               ;Set Z
                2112Ø           ENDIF
                2113Ø ;
ØE35 C9         2114Ø           RET
                2115Ø ;
ØE36 DBF8       2116Ø $O4       IN      A,(PRPORT)      ;Scan PR status
ØE38 E6FØ       2117Ø           AND     ØFØH            ;Mask unused positions
ØE3A FE3Ø       2118Ø           CP      3ØH             ;PR ready?
ØE3C C9         2119Ø           RET                     ;Return with answer
ØE3C            212ØØ PREND     EQU     $-1
ØE3D            Ø171Ø *GET      FDCDVR:3                ;Floppy disk driver
                2121Ø ;FDCDVR/ASM - LS-DOS 6.2
ØE3D            2122Ø           SUBTTL  '<Floppy Disk Driver>'
```

Floppy Disk Driver

```
                21240 ;
                21250 ;         HL=> buffer address
                21260 ;          D=> track desired
                21270 ;          E=> sector desired
                21280 ;          C=> drive desired
                21290 ;          B=> disk primitive command
                21300 ;
ØØE4            21310 WRNMIPORT                         ;NMI mask register
ØØFØ            21320 FDCADR                            ;FDC command
ØØFØ            21330 FDCSTAT                           ;FDC status
ØØF1            21340 TRKREG                            ;FDC track register
ØØF2            21350 SECREG                            ;FDC sector register
ØØF3            21360 DATREG                            ;FDC data register
ØØF4            21370 DSELCT                            ;Drive select port
                21380 ;
                21390 ;
                21400 ;         Disk Driver Entry Point
                21410 ;
ØE3D 184F       21420 FDCDVR    JR      FDCBGN          ;Branch to entry code
ØE3F F3ØF       21430           DW      FDCEND          ;Last byte used
ØE41 Ø3         21440           DB      3,'$FD'         ;Module name
     24 46 44
                21450 ;
                21460 ;         Automatic density recognition and retry density switch
                21470 ;
                21480 SWDEN
ØE45 3EØ3       21490           LD      A,3             ;Check counter for 2
ØE47 B8         21500           CP      B               ;  tries left after this one
ØE48 285E       21510           JR      Z,RESTOR        ;If so try a RESTORE
                21520 ;
ØE4A FD7EØ3     21530           LD      A,(IY+3)        ;Flip the density bit,
ØE4D EE4Ø       21540           XOR     4ØH             ;  Bit 6, (IY+3)
ØE4F FD77Ø3     21550           LD      (IY+3),A
ØE52 Ø1Ø924     21560           LD      BC,24Ø9H        ;Set alloc to SDEN
ØE55 CB77       21570           BIT     6,A             ;Test SDEN/DDEN
ØE57 28Ø3       21580           JR      Z,SDEN          ;Do SDEN if it was DDEN
ØE59 Ø11145     21590           LD      BC,4511H        ;  else set alloc to DDEN
ØE5C FD71Ø7     21600 SDEN      LD      (IY+7),C
ØE5F FD7ØØ8     21610           LD      (IY+8),B
ØE62 C9         21620           RET
                21630 ;
                21640 ;         Verify routine
                21650 ;
ØE63 21E8ØF     21660 VERFIN    LD      HL,BUCKET       ;Set byte bucket
ØE66 3E2D       21670           LD      A,2DH           ;Set for DEC L,...
ØE68 1E         21680           DB      1EH             ;Ignore next with LD E,n
                21690 ;
                21700 ;         Read routine
                21710 ;
ØE69 AF         21720 RDIN      XOR     A               ;Set for NOP
ØE6A 327DØE     21730           LD      (CKVER),A
ØE6D CD2BØF     21740           CALL    RWINIT          ;Initialize
ØE7Ø 1E16       21750           LD      E,16H           ;Status mask
ØE72 DBFØ       21760 RDIN1     IN      A,(FDCSTAT)     ;Get status
ØE74 A3         21770           AND     E               ;Loop until DRQ
ØE75 28FB       21780           JR      Z,RDIN1         ;  or error
ØE77 EDA2       21790           INI                     ;Grab byte
ØE79 F3         21800           DI
ØE7A 7A         21810           LD      A,D             ;Get drive sel + WSGEN
ØE7B D3F4       21820 RDIN2     OUT     (DSELCT),A      ;Initiate wait state
```

Floppy Disk Driver

```
ØE7D ØØ          21830 CKVER   NOP                       ;DEC L: if verify
ØE7E EDA2        21840         INI                       ;Xfer byte
ØE8Ø 2ØF9        21850         JR      NZ,RDIN2          ;Loop then TSTBSY
                 21860 ;
                 21870 ;       Reselect drive while controller is busy
                 21880 ;
ØE82 DBFØ        21890 TSTBSY  IN      A,(FDCSTAT)       ;Ck FDC status
ØE84 CB47        21900         BIT     Ø,A               ;Busy?
ØE86 C8          21910         RET     Z                 ;RET if not
ØE87 3A1BØØ      21920         LD      A,(PDRV$)         ;P/u drive
ØE8A D3F4        21930         OUT     (DSELCT),A        ;  & reselect
ØE8C 18F4        21940         JR      TSTBSY            ;Loop until idle
                 21950 ;
                 21960 ;       Driver start
                 21970 ;
ØE8E 78          21980 FDCBGN  LD      A,B               ;P/u primitive request
ØE8F A7          21990         AND     A                 ;NOP?
ØE9Ø C8          22000         RET     Z                 ;Quit if so
ØE91 FEØ7        22010         CP      7
ØE93 28ED        22020         JR      Z,TSTBSY          ;Jump on TSTBSY request
ØE95 D244ØF      22030         JP      NC,IORQST         ;Jump on I/O request
ØE98 FEØ6        22040         CP      6
ØE9A 284C        22050         JR      Z,SEEKTRK         ;Jump on track seek
ØE9C 3D          22060         DEC     A
ØE9D 2811        22070         JR      Z,SELECT          ;Jump on drive select
ØE9F FD34Ø5      22080         INC     (IY+5)            ;Bump current cylinder
ØEA2 FEØ4        22090         CP      4
ØEA4 Ø658        22100         LD      B,58H             ;FDC step-in command
ØEA6 2872        22110         JR      Z,STEPIN
ØEA8 FD36Ø5ØØ    22120 RESTOR  LD      (IY+5),Ø          ;Set track to Ø
ØEAC Ø6Ø8        22130         LD      B,8               ;Restore drive
ØEAE 186A        22140         JR      STEPIN
                 22150 ;
ØEBØ CD82ØE      22160 SELECT  CALL    TSTBSY            ;Check drive status
ØEB3 Ø7          22170         RLCA
ØEB4 F5          22180         PUSH    AF                ;Save NOT READY flag
ØEB5 C5          22190         PUSH    BC
ØEB6 FD7EØ3      22200         LD      A,(IY+3)          ;P/u SDEN/DDEN
ØEB9 17          22210         RLA                       ;Bit 6=>7, bit 4=>4
ØEBA CB2F        22220         SRA     A
ØEBC E69Ø        22230         AND     9ØH               ;Keep only DDEN & side 1
ØEBE 4F          22240         LD      C,A               ;Save the bits
ØEBF CB7F        22250         BIT     7,A               ;Check if SDEN or DDEN
ØEC1 28Ø8        22260         JR      Z,NOPCMP          ;No precomp if SDEN
ØEC3 FD7EØ9      22270         LD      A,(IY+9)          ;Set precomp on all
ØEC6 BA          22280         CP      D                 ;  tracks above DIR
ØEC7 3ØØ2        22290         JR      NC,NOPCMP         ;Go if no precomp needed
ØEC9 CBE9        22300         SET     5,C               ;Request precomp
ØECB FD7EØ4      22310 NOPCMP  LD      A,(IY+4)          ;Get drive sel code
ØECE E6ØF        22320         AND     ØFH               ;Keep only sel bits
ØEDØ B1          22330         OR      C                 ;Merge in bits 4,5,7
ØED1 C1          22340         POP     BC
ØED2 D3F4        22350         OUT     (DSELCT),A        ;Select drive
ØED4 321BØØ      22360         LD      (PDRV$),A         ;Store port byte
ØED7 F1          22370         POP     AF                ;Retrieve Not Ready bit
ØED8 DØ          22380         RET     NC                ;Ret if was ready
ØED9 FDCBØ356    22390         BIT     2,(IY+3)          ;Check DELAY=Ø.5 or 1.Ø
ØEDD CCEØØE      22400         CALL    Z,FDCDLY          ;Double delay if 1.Ø
ØEEØ C5          22410 FDCDLY  PUSH    BC                ;Delay routine
```

Floppy Disk Driver

```
ØEE1 Ø67F      22420          LD      B,7FH
ØEE3 CD82Ø3    22430          CALL    PAUSE@
ØEE6 C1        22440          POP     BC
ØEE7 C9        22450          RET
               22460 ;
               22470 ;        Routine to seek a track
               22480 ;
ØEE8 CD82ØE    22490 SEEKTRK  CALL    TSTBSY          ;Wait till not busy
ØEEB FD7EØ5    22500          LD      A,(IY+5)        ;P/u current cylinder
ØEEE D3F1      22510          OUT     (TRKREG),A      ;  & set FDC to current
ØEFØ FD7EØ7    22520          LD      A,(IY+7)        ;P/u alloc data
ØEF3 E61F      22530          AND     1FH             ;Get highest # sector
ØEF5 93        22540          SUB     E               ;Form req sector minus
ØEF6 2F        22550          CPL                     ;  max, setting CY flag if
ØEF7 FDCBØ3A6  22560          RES     4,(IY+3)        ;  init side select to Ø
ØEFB 3ØØB      22570          JR      NC,SETSECT      ;Go if sector on side Ø
ØEFD FDCBØ46E  22580          BIT     5,(IY+4)        ;If not 2-sided media,
ØFØ1 28Ø6      22590          JR      Z,FRCSIDØ       ;  don't set side 1
ØFØ3 FDCBØ3E6  22600          SET     4,(IY+3)        ;Set side 1
ØFØ7 1E        22610          DB      1EH             ;Ignore next with LD E,n
ØFØ8 7B        22620 SETSECT  LD      A,E             ;Restore unaltered sec #
ØFØ9 D3F2      22630 FRCSIDØ  OUT     (SECREG),A      ;Set sector
ØFØB 7A        22640          LD      A,D
ØFØC D3F3      22650          OUT     (DATREG),A      ;Set desired track
ØFØE FDBE05    22660          CP      (IY+5)          ;If at desired track,
ØF11 Ø618      22670          LD      B,18H           ;  use seek, else use
ØF13 28Ø5      22680          JR      Z,STEPIN        ;  seek w/verify
ØF15 FD72Ø5    22690          LD      (IY+5),D        ;Update current cylinder
ØF18 Ø61C      22700          LD      B,1CH           ;Seek w/verify ocmmand
ØF1A CDBØØE    22710 STEPIN   CALL    SELECT          ;Select drive
ØF1D FD7EØ3    22720          LD      A,(IY+3)
ØF2Ø E6Ø3      22730          AND     3               ;Strip all but step rate
ØF22 BØ        22740          OR      B
ØF23 D3FØ      22750 PASSCMD  OUT     (FDCADR),A      ;Give FDC its command
ØF25 Ø612      22760          LD      B,12H
ØF27 1ØFE      22770          DJNZ    $               ;Wait
ØF29 AF        22780          XOR     A
ØF2A C9        22790 FDCRET   RET
               22800 ;
               22810 ;        Read and write init routines
               22820 ;
ØF2B 7A        22830 RWINIT   LD      A,D             ;Restuff trk reg
ØF2C D3F1      22840          OUT     (TRKREG),A
ØF2E 3A1BØØ    22850          LD      A,(PDRV$)       ;Get select code
ØF31 F64Ø      22860          OR      4ØH             ;Set WSGEN bit
ØF33 57        22870          LD      D,A             ;Save code in D
ØF34 E61Ø      22880          AND     1ØH             ;Get side sel bit
ØF36 ØF        22890          RRCA                    ;  to bit 3
ØF37 CB49      22900          BIT     1,C             ;Check if doing side cmp
ØF39 2ØØ1      22910          JR      NZ,GETCMD       ;Go if so
ØF3B AF        22920          XOR     A
ØF3C B1        22930 GETCMD   OR      C
ØF3D ØEF3      22940          LD      C,DATREG        ;Get port into C
ØF3F CDØEØØ    22950          CALL    FDDINT$         ;Interrupts on or off?
ØF42 18DF      22960          JR      PASSCMD         ;Pass command to ctrlr
               22970 ;
               22980 ;        I/O request handler
               22990 ;
ØF44 CB5Ø      23000 IORQST   BIT     2,B             ;Write command?
```

Floppy Disk Driver

```
ØF46 ED4B7AØØ 23Ø1Ø         LD      BC,(RFLAG$-1)   ;P/u retry count
ØF4A ØE82     23Ø2Ø         LD      C,82H           ;FDC cmd=readsec
ØF4C 2Ø1Ø     23Ø3Ø         JR      NZ,WRCMD        ;Go if write command
ØF4E FEØA     23Ø4Ø         CP      1Ø              ;Verify sector?
ØF5Ø 28Ø6     23Ø5Ø         JR      Z,VERFY
ØF52 CD79ØF   23Ø6Ø         CALL    GRABNDO         ;Grab next code & insert
ØF55 Ø1       23Ø7Ø         DB      1               ;Error code start
ØF56 69ØE     23Ø8Ø         DW      RDIN            ;Read entry point
ØF58 CD79ØF   23Ø9Ø VERFY   CALL    GRABNDO         ;Stuff I/O direction
ØF5B Ø1       231ØØ         DB      1               ;Error code start
ØF5C 63ØE     2311Ø         DW      VERFIN          ;Verify entry point
ØF5E FDCBØ37E 2312Ø WRCMD   BIT     7,(IY+3)        ;Software WP?
ØF62 28Ø3     2313Ø         JR      Z,WRCMD1        ;Bypass if not
ØF64 3EØF     2314Ø         LD      A,15            ;Else set WP error
ØF66 C9       2315Ø         RET
ØF67 ØEA2     2316Ø WRCMD1  LD      C,ØA2H          ;Write sector FDC command
ØF69 FEØE     2317Ø         CP      14              ;Directory sector?
ØF6B 38Ø6     2318Ø         JR      C,DOWRIT
ØF6D ØEA3     2319Ø         LD      C,ØA3H          ;Change DAM if directory
ØF6F 28Ø2     232ØØ         JR      Z,DOWRIT
ØF71 ØEFØ     2321Ø         LD      C,ØFØH          ;  else write track
ØF73 CD79ØF   2322Ø DOWRIT  CALL    GRABNDO         ;Switch code
ØF76 Ø9       2323Ø         DB      9               ;Error code start
ØF77 BCØF     2324Ø         DW      WROUT           ;Write entry point
              2325Ø ;
              2326Ø ;       Routine stuffs error start byte & I/O vector
              2327Ø ;
ØF79 E3       2328Ø GRABNDO EX      (SP),HL         ;Save HL & get ret addr
ØF7A 7E       2329Ø         LD      A,(HL)          ;P/u & stuff error code
ØF7B 23       233ØØ         INC     HL              ;  start byte
ØF7C 32B5ØF   2331Ø         LD      (ERRSTRT+1),A
ØF7F 7E       2332Ø         LD      A,(HL)          ;Set up data transfer
ØF8Ø 23       2333Ø         INC     HL              ;  direction vector
ØF81 66       2334Ø         LD      H,(HL)
ØF82 6F       2335Ø         LD      L,A
ØF83 2293ØF   2336Ø         LD      (CALLIO),HL     ;Stuff CALL vector
ØF86 E1       2337Ø         POP     HL              ;Restore buffer addr
              2338Ø ;
              2339Ø ;       Main I/O handler routine
              234ØØ ;
ØF87 C5       2341Ø RETRY   PUSH    BC              ;Save retry & FDC command
ØF88 D5       2342Ø         PUSH    DE              ;Save track/sector
ØF89 E5       2343Ø         PUSH    HL              ;Save buffer
ØF8A CB61     2344Ø         BIT     4,C             ;Test for track command
ØF8C CCE8ØE   2345Ø         CALL    Z,SEEKTRK       ;Seek if not track write
ØF8F CD82ØE   2346Ø         CALL    TSTBSY          ;Wait till not busy
ØF92 CDØØØØ   2347Ø         CALL    Ø               ;Call I/O routine
ØF93          2348Ø CALLIO  EQU     $-2             ;Data xfer direction
ØF95 ØØ       2349Ø DISKEI  NOP                     ;Will be changed to a EI after
              235ØØ                                 ;  BOOT has read in SYSØ
ØF96 DBFØ     2351Ø         IN      A,(FDCSTAT)     ;Get status
ØF98 E67C     2352Ø         AND     7CH             ;Strip all but 2-6
ØF9A E1       2353Ø         POP     HL
ØF9B D1       2354Ø         POP     DE              ;Rcvr track & sector
ØF9C C1       2355Ø         POP     BC              ;Rcvr retry count & cmd
ØF9D C8       2356Ø         RET     Z               ;Ret if no error
ØF9E CB57     2357Ø         BIT     2,A             ;Lost data?
ØFAØ 2ØE5     2358Ø         JR      NZ,RETRY        ;Don't count this retry
ØFA2 F5       2359Ø         PUSH    AF
```

Floppy Disk Driver

```
ØFA3 E618     236ØØ          AND     18H             ;Record not found or CRC
ØFA5 28ØB     2361Ø          JR      Z,DISKDUN       ;No retries if otherwise
ØFA7 CB67     2362Ø          BIT     4,A             ;Record not found?
ØFA9 C5       2363Ø          PUSH    BC              ;If so, switch
ØFAA C4450E   2364Ø          CALL    NZ,SWDEN        ;  density or restore
ØFAD C1       2365Ø          POP     BC
ØFAE F1       2366Ø          POP     AF
ØFAF 1ØD6     2367Ø          DJNZ    RETRY           ;Count down retry
ØFB1 Ø6       2368Ø          DB      6               ;Ignore next with "LD B,nn"
ØFB2 F1       2369Ø DISKDUN  POP     AF              ;Adjust ret code
ØFB3 47       237ØØ          LD      B,A
ØFB4 3EØØ     2371Ø ERRSTRT  LD      A,Ø             ;Start with R=1, W=9
ØFB6 CBØ8     2372Ø ERRTRAN  RRC     B
ØFB8 D8       2373Ø          RET     C
ØFB9 3C       2374Ø          INC     A
ØFBA 18FA     2375Ø          JR      ERRTRAN
              2376Ø ;
              2377Ø ;        Write routine
              2378Ø ;
ØFBC CD2BØF   2379Ø WROUT    CALL    RWINIT          ;Set up initialization
ØFBF 1E76     238ØØ          LD      E,76H           ;Status mask
ØFC1 DBFØ     2381Ø WRO1     IN      A,(FDCSTAT)     ;P/u status
ØFC3 A3       2382Ø          AND     E               ;Fall out on DRQ or error
ØFC4 28FB     2383Ø          JR      Z,WRO1          ;  else loop
ØFC6 EDA3     2384Ø          OUTI                    ;Xfer byte to FDC
ØFC8 F3       2385Ø          DI                      ;Now kill the interrupts
ØFC9 DBFØ     2386Ø          IN      A,(FDCSTAT)     ;Check for errors
ØFCB 1F       2387Ø          RRA                     ;Did BUSY drop?
ØFCC DØ       2388Ø          RET     NC              ;Quit now if so
ØFCD 3ECØ     2389Ø          LD      A,ØCØH          ;Enable INTRQ and timeout
ØFCF D3E4     239ØØ          OUT     (WRNMIPORT),A
ØFD1 Ø65Ø     2391Ø          LD      B,5ØH           ;Time delay for WRSEC
ØFD3 1ØFE     2392Ø          DJNZ    $
ØFD5 46       2393Ø          LD      B,(HL)          ;Get next byte early
ØFD6 23       2394Ø          INC     HL
ØFD7 7A       2395Ø WRO3     LD      A,D             ;Enable wait states
ØFD8 D3F4     2396Ø          OUT     (DSELCT),A
ØFDA DBFØ     2397Ø          IN      A,(FDCSTAT)     ;Check if timed out
ØFDC A3       2398Ø          AND     E               ;Loop back if it timed
ØFDD 28F8     2399Ø          JR      Z,WRO3          ;  out (must be WRTRK)
ØFDF ED41     24ØØØ          OUT     (C),B           ;Pass 2nd byte
ØFE1 7A       24Ø1Ø          LD      A,D             ;Get sel code + WSGEN bit
ØFE2 D3F4     24Ø2Ø WRO2     OUT     (DSELCT),A      ;Pass until FDC times out
ØFE4 EDA3     24Ø3Ø          OUTI                    ;  & generates NMI
ØFE6 18FA     24Ø4Ø          JR      WRO2
              24Ø5Ø          IFEQ    $&ØFFH,ØFFH
              24Ø6Ø          ERR     'Warning... BUCKET position error
              24Ø7Ø          ENDIF
ØFE8 53       24Ø8Ø BUCKET   DB      'S'
              24Ø9Ø ;
ØFE9 AF       241ØØ @RSTNMI  XOR     A               ;NMI vectors here
ØFEA D3E4     2411Ø          OUT     (WRNMIPORT),A   ;Disable INTRQ & timeout
ØFEC Ø164ØØ   2412Ø          LD      BC,1ØØ          ;Need to wait a moment
ØFEF CD82Ø3   2413Ø          CALL    PAUSE@          ;Call pause
ØFF2 E1       2414Ø          POP     HL              ;Discard return
ØFF3 C9       2415Ø          RET
ØFF3          2416Ø FDCEND   EQU     $-1
ØFF4          Ø172Ø DVREND$  EQU     $               ;Start of low I/O area, to 12FFH
              Ø173Ø          IFGT    $,12ØØH+START$
```

Floppy Disk Driver

```
                01740           ERR     'Drivers overflow available RAM
                01750           ENDIF
1300            01760           ORG     1300H+START$
1300            01770 @BYTEIO   EQU     $
0000            01780           END
00000 Total errors
```

Origin	Symbolic Label	Value	Line#	Usage	Line#'s of References
$MAIN	@$SYS	08F0	01550		
$MAIN	+@@1	0000	01780		
$MAIN	+@@2	0000	01780		
$MAIN	+@@3	0000	01780		
$MAIN	+@@4	0000	01780		
CLOCKS	@BANK	0877	09870	IODVR	05530 05730
$MAIN	@BYTEIO	1300	01770	IODVR	05870
IODVR	@CHNIO	0689	05830	IODVR	05640 05980
IODVR	@CKBRKC	0553	03600		
IODVR	@CLS	0545	03490	IODVR	04410
IODVR	@CTL	0623	05170	IODVR	03750
CLOCKS	@DATE	07A8	08170		
MULDIV	@DIV16	06E3	06580	MULDIV	06920
				DODVR	20280
IODVR	@DSP	0642	05330	IODVR	03540 04360 04610 05040
IODVR	@DSPLY	052D	03290	IODVR	03030 04730
$MAIN	@FRENCH	0000	00210	$MAIN	00220 00250
				CLOCKS	08620
				$MAIN	01640
$MAIN	@GERMAN	0000	00200	$MAIN	00220 00250
				CLOCKS	08590
				$MAIN	01590
IODVR	@GET	0638	05280	IODVR	05260
CLOCKS	@HEX16	07BD	08340		
CLOCKS	@HEX8	07C2	08370	CLOCKS	08350
MULDIV	@HEXDEC	06F6	06840		
$MAIN	@HZ50	0000	00320	CLOCKS	07420
$MAIN	@INTL	0000	00300	CLOCKS	08830
				DODVR	15480
				PRDVR	20890 20960 21090
IODVR	@JCL	0630	05250	IODVR	04040
IODVR	@KBD	0635	05270	IODVR	05200
IODVR	@KEY	0628	05200	IODVR	04000 05230
IODVR	@KEYIN	0585	03980		
$MAIN	@KITSK	0089	00570	KIDVR	11080
IODVR	@LOGER	0503	03070		
IODVR	@LOGOT	0500	03030		
$MAIN	@MOD2	0000	00120		
$MAIN	@MOD4	FFFF	00130		
IODVR	@MSG	0530	03340	IODVR	03170 03190 03250
MULDIV	@MUL16	06C9	06310	DODVR	20070
DODVR	@OPREG	0084	15220	BOOT4	01670
				CLOCKS	09590 10470
IODVR	@PRINT	0528	03240		
IODVR	@PRT	063D	05310	KIDVR	11430 11610
IODVR	@PUT	0645	05340	IODVR	03400 03430 05320
FDCDVR	@RSTNMI	0FE9	24100		
IODVR	@RSTREG	0680	05750	IODVR	05420
CLOCKS	@TIME	078D	07960	IODVR	03140
$MAIN	@USA	FFFF	00310	CLOCKS	08560 08800
				$MAIN	01560
				DODVR	15450
DODVR	@VDCTL	0B99	15570	KIDVR	11350
DODVR	@VDCTL3	0D38	18690	DODVR	19100
DODVR	@_VDCTL	0D42	18770	DODVR	15570
DODVR	ADDR1	0DF4	20240	DODVR	16170
DODVR	ADDR_2_ROWCOL	0DF1	20220	IODVR	04780
				DODVR	19120
CLOCKS	AFSAV	0866	09670	CLOCKS	09180 09500

Origin	Symbolic Label	Value	Line#	Usage	Line#'s of References
DODVR	BACKSPA	0C12	1639	DODVR	1687
$MAIN	BAR$	0201	0064	CLOCKS	1031
BOOT4	BOOT	4300	0162		
BOOT4	BOOTBUF	43F6	0010	BOOT4	0212
BOOT4	BOOTST$	439D	0247	BOOT4	0248
IODVR	BRKTEST	055E	0368	IODVR	0372
FDCDVR	BUCKET	0FE8	2408	FDCDVR	2166
BOOT4	BUFFER	1200	0009	BOOT4	0201 0208 0211 0218 0225
$MAIN	BUR$	0200	0063	CLOCKS	1019
FDCDVR	CALLIO	0F93	2348	FDCDVR	2336
KIDVR	CAPSKEY	0A79	1321	KIDVR	1277
KIDVR	CASHK$	0A7B	1322		
$MAIN	CFLAG$	006C	0047	IODVR	0465
KIDVR	CKCTL	0ACE	1367		
KIDVR	CKCTL1	0AC8	1362	KIDVR	1367
KIDVR	CKCTL2	0AD2	1369	KIDVR	1361
FDCDVR	CKVER	0E7D	2183	FDCDVR	2173
CLOCKS	CLOCK	0787	0793	CLOCKS	0751
DODVR	CLREOF	0D1E	1848	DODVR	1721
DODVR	CLREOF1	0D1F	1849	DODVR	1810
DODVR	CLREOF2	0D22	1850	DODVR	1844
DODVR	CLREOF3	0D36	1864	DODVR	1855
DODVR	CLREOL	0D12	1838	DODVR	1719
IODVR	CLRFRM	001F	0299	IODVR	0352
KIDVR	CLRTYP	0B65	1477	KIDVR	1383
BOOT4	+CORE$	0300	0160	BOOT4	0292
KIDVR	CR	000D	1068	IODVR	0338
				KIDVR	1093 1102 1158 1160
DODVR	CRSAVE	0B97	1552	CLOCKS	0714 0735
				DODVR	1587 1591 1624 1974
DODVR	CRSBKSP	0C1B	1647	DODVR	1640 1707
DODVR	CRSBOL	0BF8	1615	DODVR	1717 1830 1839
DODVR	CRSCHAR	0B98	1553	CLOCKS	0737
				DODVR	1595 1926
DODVR	CRSDOWN	0C30	1665	DODVR	1711 1832
DODVR	CRSFRW0	0CC3	1781	DODVR	1673 1779
DODVR	CRSFRWD	0CBB	1776	DODVR	1709
DODVR	CRSHOME	0C05	1630	DODVR	1715
DODVR	CRSOFF	0C01	1624	DODVR	1695
DODVR	CRSON	0C00	1623	DODVR	1693
DODVR	CRSUP	0C2B	1659	DODVR	1713 1785
DODVR	CRTBGN$	F800	1530	BOOT4	0169 0170 0239
				CLOCKS	0795 0830
				DODVR	1531 1551 1631 1651 1669 1790
				DODVR	1913 2005
DODVR	CRTCADD	0088	1523		
DODVR	CRTCDAT	0089	1524		
BOOT4	CRTCTAB	02FD	0155	BOOT4	0132
DODVR	CRTEND	FF7F	1531	DODVR	1782 1834 1849
DODVR	CRTSIZE	078	1528	BOOT4	0171
				DODVR	1531 1791 1955
DODVR	CTL	0004	1544	DODVR	1567 1584
KIDVR	CTLA2Z	0A13	1263	KIDVR	1251
KIDVR	CTLFF	0AEC	1391	KIDVR	1385
DODVR	CURSOR	0B95	1551	CLOCKS	0734
				DODVR	1597 1872 1895 1973 2023
$MAIN	DATE$	0033	0044	CLOCKS	0767 0817
FDCDVR	DATREG	00F3	2136	FDCDVR	2265 2294
$MAIN	DAYTBL$	04C7	0147		

Origin	Symbolic Label	Value	Line#	Usage	Line#'s of References
$MAIN	DCBKL$	ØØ31	ØØ85Ø		
$MAIN	DCT$	Ø47Ø	Ø115Ø	BOOT4	ØØ16Ø ØØ78Ø Ø197Ø Ø2Ø3Ø
KIDVR	DELAY2	ØA95	1334Ø		
$MAIN	DFLAG$	ØØ6D	ØØ48Ø	KIDVR	1124Ø 1439Ø
BOOT4	DIRTRK	43Ø2	Ø164Ø	BOOT4	Ø196Ø
FDCDVR	DISKDUN	ØFB2	2369Ø	FDCDVR	2361Ø
FDCDVR	DISKEI	ØF95	2349Ø	BOOT4	ØØ38Ø
BOOT4	DISKERR	43EØ	Ø288Ø	BOOT4	Ø235Ø
CLOCKS	DIS_DO_RAM	Ø846	Ø945Ø	CLOCKS	Ø929Ø
DODVR	DOBGN	ØB9C	1561Ø	DODVR	1535Ø
DODVR	DODATA$	ØB94	154ØØ	DODVR	1541Ø 1561Ø 1821Ø 1824Ø
$MAIN	DODCB$	Ø21Ø	ØØ71Ø	$MAIN	ØØ81Ø
				IODVR	Ø329Ø Ø533Ø
				DODVR	1538Ø
DODVR	DODVR	ØB88	1535Ø	$MAIN	ØØ72Ø
DODVR	DOEND	ØEØØ	2Ø33Ø	DODVR	1536Ø
DODVR	DONORM	ØBC4	1583Ø	DODVR	1574Ø
CLOCKS	DOOPREG	Ø858	Ø957Ø	CLOCKS	Ø941Ø
FDCDVR	DOWRIT	ØF73	2322Ø	FDCDVR	2318Ø 232ØØ
DODVR	DO_CONTROL	ØC44	1683Ø	DODVR	1572Ø
DODVR	DO_DSPCHAR	ØCB8	1771Ø	DODVR	1583Ø 16Ø9Ø
DODVR	DO_INVERT_DIS	ØC8C	173ØØ	DODVR	1635Ø
DODVR	DO_INVERT_ENA	ØC89	1727Ø	DODVR	1697Ø
DODVR	DO_INVERT_OFF	ØC9B	1742Ø	DODVR	1699Ø
DODVR	DO_MASK	ØØØØ	1541Ø	DODVR	1567Ø 1578Ø 1584Ø 1757Ø 1788Ø
DODVR	DO_RET	ØBCB	1585Ø	DODVR	16Ø6Ø 1611Ø 1684Ø 1748Ø
DODVR	DO_RET1	ØBCC	1586Ø	DODVR	1896Ø
DODVR	DO_SCROLL	ØCCE	1787Ø	DODVR	1836Ø
DODVR	DO_TABS	ØBEA	16Ø4Ø	DODVR	1579Ø
FDCDVR	DSELCT	ØØF4	2137Ø	FDCDVR	2182Ø 2193Ø 2235Ø 2396Ø 24Ø2Ø
$MAIN	DSKTYP$	Ø4CØ	Ø141Ø		
IODVR	DSPBYT	Ø54D	Ø353Ø	IODVR	Ø35ØØ Ø491Ø
BOOT4	DSPLY	Ø21B	ØØ8Ø		
$MAIN	DTPMT$	Ø4C2	Ø143Ø		
$MAIN	DVREND$	ØFF4	Ø172Ø	$MAIN	ØØ67Ø
KIDVR	DVREXIT	ØAA8	1343Ø		
$MAIN	DVRHI$	Ø2Ø6	ØØ67Ø		
CLOCKS	ENADIS_DO_RAM	Ø817	Ø913Ø	CLOCKS	Ø729Ø Ø794Ø Ø829Ø Ø866Ø
				KIDVR	1377Ø 1442Ø
				DODVR	1562Ø 1878Ø
BOOT4	ERRLEN	ØØØA	Ø29ØØ	BOOT4	Ø238Ø
FDCDVR	ERRSTRT	ØFB4	2371Ø	FDCDVR	2331Ø
FDCDVR	ERRTRAN	ØFB6	2372Ø	FDCDVR	2375Ø
FDCDVR	FDCADR	ØØFØ	2132Ø	FDCDVR	2275Ø
FDCDVR	FDCBGN	ØE8E	2198Ø	FDCDVR	2142Ø
FDCDVR	FDCDLY	ØEEØ	2241Ø	FDCDVR	224ØØ
FDCDVR	FDCDVR	ØE3D	2142Ø	$MAIN	Ø116Ø Ø118Ø Ø121Ø Ø124Ø
FDCDVR	FDCEND	ØFF3	2416Ø	FDCDVR	2143Ø
BOOT4	FDCMD	43D9	Ø284Ø	BOOT4	Ø252Ø Ø264Ø
FDCDVR	FDCRET	ØF2A	2279Ø	$MAIN	Ø127Ø Ø13ØØ Ø133Ø Ø136Ø
FDCDVR	FDCSTAT	ØØFØ	2133Ø	FDCDVR	2176Ø 2189Ø 2351Ø 2381Ø 2386Ø 2397Ø
$MAIN	FDDINT$	ØØØE	ØØ39Ø	BOOT4	ØØ22Ø
				FDCDVR	2295Ø
KIDVR	FIXCLR	ØA84	1327Ø	KIDVR	1311Ø
KIDVR	FIXSCL	ØA82	1326Ø	KIDVR	1313Ø
$MAIN	FLGTAB$	ØØ6A	ØØ46Ø	$MAIN	ØØ47Ø ØØ48Ø ØØ49Ø ØØ5ØØ ØØ51Ø ØØ52Ø
				$MAIN	ØØ53Ø ØØ54Ø ØØ55Ø ØØ56Ø ØØ57Ø
				BOOT4	ØØ2ØØ
FDCDVR	FRCSIDØ	ØFØ9	2263Ø	FDCDVR	2259Ø

Origin	Symbolic Label	Value	Line#	Usage	Line#'s of References
BOOT4	GETADR	Ø288	ØØ61Ø	BOOT4	ØØ44Ø ØØ52Ø
FDCDVR	GETCMD	ØF3C	2293Ø	FDCDVR	2291Ø
BOOT4	GETEXT	Ø2B1	ØØ95Ø	BOOT4	ØØ33Ø ØØ88Ø
DODVR	GET_@_ROWCOL	ØDAE	1962Ø	DODVR	19Ø6Ø
BOOT4	GOTERR	4385	Ø235Ø		
KIDVR	GOTSHFT	ØA6B	1312Ø		
FDCDVR	GRABNDO	ØF79	2328Ø	FDCDVR	23Ø6Ø 23Ø9Ø 2322Ø
BOOT4	HALTS	4394	Ø241Ø	BOOT4	Ø241Ø
CLOCKS	HERTZ$	Ø75Ø	Ø747Ø		
MULDIV	HEXDEC1	Ø6FA	Ø686Ø	MULDIV	Ø688Ø
MULDIV	HEXDEC2	Ø7ØØ	Ø691Ø	MULDIV	Ø7Ø1Ø
$MAIN	HIGH$	Ø4ØE	Ø1Ø5Ø		
KIDVR	HIT	ØACC	1365Ø	KIDVR	1352Ø
KIDVR	HITWS	ØACA	1364Ø	KIDVR	1355Ø
CLOCKS	HLSAV	Ø86B	Ø971Ø	CLOCKS	Ø915Ø Ø947Ø
IODVR	HOME	ØØ1C	Ø298Ø	IODVR	Ø349Ø
CLOCKS	HXD1	Ø7CB	Ø844Ø	CLOCKS	Ø842Ø
$MAIN	IFLAG$	ØØ72	ØØ49Ø		
$MAIN	INBUF$	Ø42Ø	Ø11ØØ		
BOOT4	INITCRTC	Ø2DD	Ø13ØØ	BOOT4	Ø192Ø
$MAIN	INTVC$	ØØ3E	ØØ45Ø		
DODVR	INVIDEO	ØDCB	1987Ø	DODVR	1743Ø 185ØØ
IODVR	IOBGN	Ø648	Ø536Ø	IODVR	Ø519Ø Ø53ØØ
FDCDVR	IORQST	ØF44	23ØØØ	FDCDVR	22Ø3Ø
$MAIN	JCLCB$	Ø2Ø3	ØØ66Ø	IODVR	Ø525Ø
$MAIN	JLDCB$	Ø23Ø	ØØ83Ø	$MAIN	ØØ85Ø
				IODVR	Ø3Ø7Ø Ø316Ø
KIDVR	KBØ	F4Ø1	1Ø69Ø	KIDVR	1171Ø 1392Ø
CLOCKS	KB1	F4Ø1	Ø857Ø	CLOCKS	Ø879Ø
KIDVR	KB6	F44Ø	1Ø7ØØ		
CLOCKS	KB7	F44Ø	Ø865Ø	CLOCKS	Ø895Ø
KIDVR	KBROWØ	ØØØ4	1Ø84Ø	KIDVR	117ØØ
KIDVR	KBROW4	ØØØ8	1Ø86Ø	KIDVR	1316Ø
KIDVR	KBROW6	ØØØA	1Ø88Ø	KIDVR	1256Ø 13ØØØ
KIDVR	KBTBL	Ø9Ø8	1Ø93Ø	KIDVR	128ØØ
CLOCKS	KCK1	Ø7F5	Ø892Ø	CLOCKS	Ø878Ø
CLOCKS	KCK1A	Ø7FC	Ø895Ø	CLOCKS	Ø886Ø Ø888Ø
CLOCKS	KCK1B	Ø8Ø5	Ø899Ø	CLOCKS	Ø894Ø Ø897Ø
CLOCKS	KCK2	Ø815	Ø9Ø7Ø	CLOCKS	Ø9Ø1Ø Ø9Ø2Ø Ø9Ø5Ø
CLOCKS	KCK@	Ø7D6	Ø866Ø		
BOOT4	KEYIN	ØØ4Ø	ØØØ6Ø		
KIDVR	KEYOK	ØA89	133ØØ	KIDVR	1315Ø 1317Ø 1328Ø
$MAIN	KFLAG$	ØØ74	ØØ5ØØ	IODVR	Ø362Ø
				CLOCKS	Ø867Ø
				KIDVR	113ØØ 1244Ø 1296Ø 1322Ø 1482Ø 1497Ø
KIDVR	KIBGN	Ø923	11Ø6Ø	KIDVR	1Ø73Ø
KIDVR	KIDATA$	Ø8FC	1Ø78Ø	KIDVR	1Ø8ØØ 1Ø82Ø 1Ø84Ø 1Ø86Ø 1Ø88Ø 1169Ø
				KIDVR	117ØØ
$MAIN	KIDCB$	Ø2Ø8	ØØ68Ø	$MAIN	ØØ78Ø
				IODVR	Ø373Ø Ø527Ø
				KIDVR	1Ø76Ø
KIDVR	KIDVR	Ø8FØ	1Ø73Ø	$MAIN	ØØ69Ø
KIDVR	KIHOOK	ØB39	1449Ø		
KIDVR	KILAST	ØB87	1517Ø	KIDVR	1Ø74Ø
KIDVR	KISCAN	Ø983	1169Ø	KIDVR	1431Ø 1449Ø
$MAIN	LBANK$	Ø2Ø2	ØØ65Ø	IODVR	Ø543Ø
				CLOCKS	1ØØ2Ø 1Ø48Ø 1Ø52Ø
BOOT4	LBOOT	Ø238	ØØ16Ø	BOOT4	Ø22ØØ
KIDVR	LF	ØØØA	1Ø67Ø	KIDVR	1Ø95Ø

Origin	Symbolic Label	Value	Line#	Usage	Line#'s of References
DODVR	LINESIZ	005Ø	1525Ø	DODVR	1527Ø 1528Ø 1666Ø 1793Ø 1943Ø 2006Ø
				DODVR	2027Ø
DODVR	LINFEED	ØD02	1830Ø	DODVR	1691Ø
BOOT4	LOAD	Ø268	0041Ø	BOOT4	0036Ø 0049Ø 0057Ø
BOOT4	LOAD1	Ø271	0045Ø	BOOT4	0048Ø
BOOT4	LOAD2	Ø27A	0051Ø	BOOT4	0043Ø
BOOT4	LOAD3	Ø281	0055Ø	BOOT4	0056Ø
IODVR	LOGBUF	Ø51D	Ø32Ø0	IODVR	Ø312Ø
$MAIN	MAXDAY$	Ø4Ø1	Ø1Ø3Ø	CLOCKS	Ø768Ø
BOOT4	MLTCA	Ø2D3	Ø12Ø0	BOOT4	Ø124Ø
BOOT4	MLTCA1	Ø2D9	Ø124Ø	BOOT4	Ø122Ø
$MAIN	MODOUT$	0076	0051Ø	DODVR	1632Ø 1648Ø 1677Ø 1763Ø 1765Ø 1777Ø
$MAIN	MONTBL$	Ø4DC	Ø148Ø		
DODVR	MOVCRS	ØC33	1667Ø	DODVR	1661Ø
DODVR	MOVLIN	ØD98	1943Ø	DODVR	1941Ø
BOOT4	MULTCA	Ø2CE	Ø116Ø	BOOT4	Ø1Ø4Ø Ø1Ø9Ø
DODVR	NEGLINE	FFBØ	1527Ø	DODVR	1660Ø
$MAIN	NFLAG$	0077	0052Ø	DODVR	1882Ø
BOOT4	NMIRET	43CD	Ø276Ø	BOOT4	Ø174Ø
BOOT4	NMIVECT	0066	0007Ø	BOOT4	Ø175Ø Ø177Ø
IODVR	NOBRK	Ø575	Ø379Ø	IODVR	Ø364Ø
KIDVR	NOCHAR	Ø9B5	1194Ø	KIDVR	1164Ø 1188Ø 1338Ø
KIDVR	NOKEY	ØA88	1329Ø	KIDVR	12Ø2Ø 1325Ø
FDCDVR	NOPCMP	ØECB	2231Ø	FDCDVR	2226Ø 2229Ø
CLOCKS	NOSOLID	Ø72F	Ø729Ø	CLOCKS	Ø727Ø
BOOT4	NOSYS	43EA	Ø289Ø	BOOT4	Ø237Ø Ø29Ø0
KIDVR	NOTALPH	ØA67	131Ø0	KIDVR	131Ø1Ø
BOOT4	NOTDBL	Ø255	0027Ø	BOOT4	0025Ø
BOOT4	NOTSYS	4389	Ø237Ø	BOOT4	Ø21Ø0
DODVR	NUMROWS	0018	1526Ø	DODVR	1528Ø
$MAIN	OPREG$	0078	0053Ø	BOOT4	Ø168Ø
				CLOCKS	Ø935Ø Ø958Ø 1Ø43Ø 1Ø46Ø
CLOCKS	OPREG_SV_AREA	Ø86E	Ø974Ø	CLOCKS	Ø893Ø Ø932Ø
CLOCKS	OPREG_SV_PTR	Ø835	Ø933Ø	CLOCKS	Ø892Ø Ø951Ø Ø957Ø
				DODVR	1732Ø
$MAIN	PAKNAM$	Ø41Ø	Ø1Ø6Ø		
FDCDVR	PASSCMD	ØF23	2275Ø	FDCDVR	2296Ø
$MAIN	PAUSE@	Ø382	0097Ø	IODVR	Ø37Ø0
				KIDVR	1332Ø
				PRDVR	2Ø78Ø
				FDCDVR	2243Ø 2413Ø
CLOCKS	PCSAVE$	Ø7AF	Ø821Ø	CLOCKS	Ø827Ø
$MAIN	PDRV$	001B	0040Ø	FDCDVR	2192Ø 2236Ø 2285Ø
DODVR	PERR	ØDED	2Ø16Ø	CLOCKS	Ø99Ø0 1Ø01Ø 1Ø27Ø
				DODVR	1953Ø 1998Ø 2Ø01Ø
CLOCKS	PERRX	Ø88F	1Ø01Ø	CLOCKS	1Ø33Ø
PRDVR	PRBGN	ØE0D	2Ø53Ø	PRDVR	2Ø45Ø
$MAIN	PRDCB$	Ø218	0074Ø	IODVR	Ø324Ø Ø531Ø
				PRDVR	2Ø48Ø
PRDVR	PRDVR	ØE01	2Ø45Ø	$MAIN	0075Ø
PRDVR	PREND	ØE3C	212Ø0	PRDVR	2Ø46Ø
PRDVR	PRPORT	00F8	2Ø39Ø	PRDVR	21Ø7Ø 2116Ø
DODVR	PUTA@DE	ØDCD	1989Ø	DODVR	1977Ø
DODVR	PUT_@	ØDCA	1986Ø	DODVR	1643Ø 1772Ø
DODVR	PUT_@_ROWCOL	ØDC6	1983Ø	DODVR	19Ø8Ø
PRDVR	PVAL3	ØE33	21Ø7Ø		
KIDVR	R7KFLG	ØB6A	1482Ø	KIDVR	1422Ø
DODVR	RAMSIZE	Ø8Ø0	1529Ø		
BOOT4	RDB1	Ø2AA	0087Ø	BOOT4	0084Ø

Origin	Symbolic Label	Value	Line#	Usage	Line#'s of References
BOOT4	RDB2	02AE	0089	BOOT4	0075
BOOT4	RDBOOT	432D	0186	BOOT4	0191
BOOT4	RDBYTE	0297	0073	BOOT4	0041 0045 0053 0055 0061 0063
				BOOT4	0066
FDCDVR	RDIN	0E69	2172	FDCDVR	2308
FDCDVR	RDIN1	0E72	2176	FDCDVR	2178
FDCDVR	RDIN2	0E7B	2182	FDCDVR	2185
BOOT4	RDS1	4379	0227	BOOT4	0234
BOOT4	RDSECT	4374	0225	BOOT4	0200 0207
BOOT4	RDSEQ	4377	0226	BOOT4	0186
BOOT4	READ	4396	0243	BOOT4	0229
BOOT4	READLP1	43BD	0267	BOOT4	0269
BOOT4	READLP2	43C5	0272	BOOT4	0274
FDCDVR	RESTOR	0EA8	2212	FDCDVR	2151
FDCDVR	RETRY	0F87	2341	FDCDVR	2358 2367
$MAIN	RFLAG$	007B	0054	FDCDVR	2301
DODVR	ROWCOL_2_ADDR	0DD0	1995	DODVR	1619 1869 1936 1963 1984
KIDVR	RPTINIT	0002	1080	KIDVR	1192 1334
KIDVR	RPTRATE	0003	1082	KIDVR	1271 1468 1471
IODVR	RSTBNK	0679	0570	IODVR	0558
$MAIN	RSTOR$	04C4	0145		
FDCDVR	RWINIT	0F2B	2283	FDCDVR	2174 2379
$MAIN	S1DCB$	0238	0084		
DODVR	SCRPROT	0007	1542	DODVR	1789
FDCDVR	SDEN	0E5C	2160	FDCDVR	2158
FDCDVR	SECREG	00F2	2135	FDCDVR	2263
BOOT4	SECTRK	02A5	0083	BOOT4	0027
BOOT4	SEEK1	43A3	0253	BOOT4	0255
FDCDVR	SEEKTRK	0EE8	2249	FDCDVR	2205 2345
FDCDVR	SELECT	0EB0	2216	FDCDVR	2207 2271
DODVR	SET40	0C3D	1677	DODVR	1705
DODVR	SETMASK	0CFD	1824	DODVR	1758
DODVR	SETMOD	0CB2	1765	DODVR	1634 1679
FDCDVR	SETSECT	0F08	2262	FDCDVR	2257
DODVR	SET_SCROLL	0CF3	1817	DODVR	1919
$MAIN	SFLAG$	007C	0055	IODVR	0401
				CLOCKS	0903
				KIDVR	1293
KIDVR	SHIFT	F480	1071	CLOCKS	0868
				KIDVR	1235
CLOCKS	SHIFT	F480	0855		
$MAIN	SIDCB$	0220	0077		
$MAIN	SODCB$	0228	0080		
KIDVR	SPCLLP	0AB7	1351	KIDVR	1359
KIDVR	SPCLTB	091E	1102	KIDVR	1346
KIDVR	SPECL	0AAB	1345	KIDVR	1341
CLOCKS	SPSAV	0863	0965	CLOCKS	0926
$MAIN	STACK$	0380	0096	$MAIN	0097
				CLOCKS	0927
$MAIN	START$	0000	0035	$MAIN	0059
				BOOT4	0183
				$MAIN	0173 0176
FDCDVR	STEPIN	0F1A	2271	FDCDVR	2211 2214 2268
FDCDVR	SWDEN	0E45	2148	FDCDVR	2364
DODVR	TABS	0003	1543	DODVR	1578
DODVR	TGGLALT	0CAD	1762	DODVR	1703
DODVR	TGGLCTL	0C9F	1748	DODVR	1570
DODVR	TGGLTAB	0CA6	1755	DODVR	1701
KIDVR	TGLCASE	0A3E	1290	KIDVR	1259 1262

Origin	Symbolic Label	Value	Line#	Usage	Line#'s of References
$MAIN	TIME$	002D	0043	CLOCKS	0754 0796
CLOCKS	TIME1	0792	0798	CLOCKS	0819
CLOCKS	TIME2	0794	0799	CLOCKS	0813
CLOCKS	TIME3	0797	0801	CLOCKS	0803
$MAIN	TIMER$	002C	0042	$MAIN	0043
				KIDVR	1189 1270 1333 1467
CLOCKS	TIMER1	0761	0756	CLOCKS	0763
CLOCKS	TIMETBL	070F	0712	CLOCKS	0739 0755
$MAIN	TIMSL$	002B	0041		
CLOCKS	TIMTSK$	0713	0713		
$MAIN	TMPMT$	04C3	0144		
CLOCKS	TRACE_INT	07B1	0825		
FDCDVR	TRKREG	00F1	2134	FDCDVR	2251 2284
FDCDVR	TSTBSY	0E82	2189	FDCDVR	2194 2202 2216 2249 2346
KIDVR	TSTSPA	0A6F	1314		
KIDVR	TYPAHD	0AD5	1376	KIDVR	1113
KIDVR	TYPBUF	FF80	1510	KIDVR	1378 1443
KIDVR	TYPHK$	0A8F	1332		
KIDVR	TYPON	0B23	1433	KIDVR	1381 1387
KIDVR	TYPTSK$	0B26	1438		
DODVR	VDCTL	0D61	1901	DODVR	1884 1892
FDCDVR	VERFIN	0E63	2166	FDCDVR	2311
FDCDVR	VERFY	0F58	2309	FDCDVR	2305
$MAIN	VFLAG$	007F	0056	CLOCKS	0716
				DODVR	1592 1594 1970
DODVR	VIDLIN	0D8B	1934	DODVR	1903
DODVR	VIDMOV1	0DA7	1955	DODVR	1915
DODVR	VIDMOVE	0D9F	1950	DODVR	1917
FDCDVR	WRCMD	0F5E	2312	FDCDVR	2303
FDCDVR	WRCMD1	0F67	2316	FDCDVR	2313
FDCDVR	WRNMIPORT	00E4	2131	FDCDVR	2390 2411
FDCDVR	WRO1	0FC1	2381	FDCDVR	2383
FDCDVR	WRO2	0FE2	2402	FDCDVR	2404
FDCDVR	WRO3	0FD7	2395	FDCDVR	2399
FDCDVR	WROUT	0FBC	2379	FDCDVR	2324
$MAIN	ZERO$	0401	0102		

0034 Symbols declared - 0541 References

NOTES:

NOTES:

SYSRES - Resident portion of the DOS

This source code assembles the load module file SYSØ/SYS, commonly referred to as Sysres. It contains the BDOS and other routines. It is origined at 13ØØH, which puts it above Lowcore and the low memory driver zone. It is loaded into position by the boot loader in Lowcore. Once loaded, execution is begun at the SYSINIT entry point. This initializes the remaining machine hardware, loads the configuration file if present, executes any Auto command, and brings in the command interpreter. The main subsections of Sysres are:

Certain low memory values and the system flag table.
FILPOSN - File read/write, positioning, and allocation routines.
LOADER - SVC handling, overlay loading, command file loading.
TASKER - Interrupt processing.
SYSINIT - System initialization after cold or warm boot.
SOUND - Sound, Pause, and some other miscellaneous low memory routines.
LOGO - The signon graphics display in direct load to screen format.

A cross reference listing of non-local labels follows the assembly listing.

NOTES:

```
                    00100 ;SYSRES/ASM - LS-DOS 6.2
0000                00110         TITLE   <SYSRES - LS-DOS 6.2>
000A                00120 LF      EQU     10
000D                00130 CR      EQU     13
                    00140 ;
                    00150 *LIST   OFF                     ;Xref of Lowcore
                    00170 *LIST   ON
0000                00180 *GET    COPYCOM:3               ;Embed copyright notice
                    01300 ; COPYCOM - File for Copyright COMment block
                    01310 ;
0000                01320         COM     '<*(C) 1982,83,84 by LSI*>'
                    01330 ;
                    00190 ;
0000                00200         SUBTTL  <'System low core assignments>'
                    00210 ;
                    00220 ;       LDOS 6.2 Low Core RAM storage assignments
                    00230 ;       Copyright (C) 1982 by Logical Systems, Inc.
                    00240 ;
0000                00250 START$  EQU     0
0000                00260         ORG     0+START$
                    00270 ;
                    00280 ;       Page 0 - RST's, data, and buffers
                    00290 ;
0000 F3             00300 @RST00  DI                      ;IPL Entry for R/S 4-P
0001 3E01           00310         LD      A,00000001B     ;Set image in A
0003 D39C           00320         OUT     (9CH),A         ;toggle in BOOT/ROM
0005 00             00330         DB      0,0,0           ;CP/M emulator SVC
     00 00
0008 C9             00340 @RST08  RET
0009 0000           00350         DW      0
000B 0000           00360 SVCRET$ DW      0               ;Return address from SVC
000D 00             00370 LSVC$   DB      0               ;Last SVC executed
000E F3             00380 FDDINT$ DI                      ;NOP or DI (F3H) for
000F C9             00390         RET                     ;  System (Smooth)
0010 C9             00400 @RST10  RET
0011 0000           00410         DW      0
0005                00420 USTOR$  DS      5               ;User storage area
0018 C9             00430 @RST18  RET
0019 0000           00440         DW      0
001B 01             00450 PDRV$   DB      1               ;Current drive, physical
001C 0000           00460 PHIGH$  DW      0               ;Physical HIGH$
001E 0030           00470 LOW$    DW      3000H           ;Lowest usable memory
0020 C9             00480 @RST20  RET
0021 0000           00490         DW      0
0023 00             00500 LDRV$   DB      0               ;Current drive, logical
0024 0000           00510 JDCB$   DW      0               ;Saved FCB pointer
0026 0000           00520 JRET$   DW      0               ;Saved I/O return address
0028 C35B1A         00530 @RST28  JP      RST28           ;System SVC processor
002B 55             00540 TIMSL$  DB      55H             ;Fast=55, slow=FF
002C 00             00550 TIMER$  DB      0               ;RTC counter
002D 00             00560 TIME$   DC      3,0             ;SS:MM:HH storage area
     00 00
0030 C3A019         00570 @RST30  JP      @DEBUG          ;DEBUG call address
0005                00580 DATE$   DS      5               ;YY/DD/MM/packed
0038 C3FF1B         00590 @RST38  JP      RST38@          ;Interrupt RST
003B 00             00600 OSRLS$  DB      00H             ;OS Release #
                    00610 ;
                    00620 ;       INTIM$ stores the image read from RDINTSTATUS*
                    00630 ;
003C 00             00640 INTIM$  DB      0               ;Interrupt latch image
                    00650 ;
```

```
'System low core assignments

                00660 ;       INTMSK$ masks the image read from RDINTSTATUS*
                00670 ;       LDOS 6.x permits only RS-232 RCV INT, IOBUS INT,
                00680 ;       and RTC INT to be used by the TASKER off of RST38
                00690 ;
003D 2C         00700 INTMSK$ DB      2CH             ;Mask for INTIM$
                00710 ;
                00720 ;       INTVC$ stores the eight vectors associated
                00730 ;       with the INTIM$ bit assignments
                00740 ;
003E 481C       00750 INTVC$  DW      RETINST         ;Primary interrupts
0040 481C       00760         DW      RETINST,RTCPROC,RETINST
     941C 481C
0046 481C       00770         DW      RETINST,RETINST,RETINST,RETINST
     481C 481C 481C
                00780 ;
                00790 ;       TCB$ stores the TCB vectors for task slots 0-11
                00800 ;
0018            00810 TCB$    DS      24              ;Interrupt task vectors
                00820 ;
                00830 ;       NMI vector used in disk I/O
                00840 ;
0003            00850 @NMI    DS      3               ;Don't overlay this
                00860 ;
                00870 ;       OVRLY$ stores the system's overlay request #
                00880 ;
0069 00         00890 OVRLY$  DB      0               ;Current overlay resident
                00900 ;
                00910 ;       FLGTAB$ stores 26 flags and images. A pointer
                00920 ;       to this table is obtained from SVC-@FLAGS
                00930 ;
006A            00940 FLGTAB$ EQU     $
                00950 ;
                00960 ;
                00970 ;       AFLAG$ - Start CYL for Allocation search
                00980 ;
006A 01         00990 AFLAG$  DB      01              ;AFLAG
006B 00         01000         DB      0               ;BFLAG
                01010 ;
                01020 ;       CFLAG$ assignments:
                01030 ;        0 - Cannot change HIGH$ via SVC-100
                01040 ;        1 - @CMNDR in execution
                01050 ;        2 - @KEYIN request from SYS1
                01060 ;        3 - System request for drivers, filters, DCTs
                01070 ;        4 - @CMNDR to only execute LIB commands
                01080 ;        5 - Sysgen inhibit bit
                01090 ;        6 - @ERROR inhibit display
                01100 ;        7 - @ERROR to use user (DE) buffer
                01110 ;
006C 00         01120 CFLAG$  DB      0               ;Condition flag
                01130 ;
                01140 ;       DFLAG$ assignments:
                01150 ;        0 - SPOOL is active
                01160 ;        1 - TYPE ahead is active
                01170 ;        2 - VERIFY is on
                01180 ;        3 - SMOOTH active
                01190 ;        4 - MemDISK active
                01200 ;        5 - FORMS active
                01210 ;        6 - KSM active
                01220 ;        7 - accept GRAPHICS in screen print
```

```
'System low core assignments

                    01230 ;
006D 0A             01240 DFLAG$  DB      00001010B       ;DEV Flag (SMOOTH,TYPE)
                    01250 ;
                    01260 ;       EFLAG$ - Assignments: (sys13 usage)
                    01270 ;       use only bits 4, 5 and 6 to indicate user
                    01280 ;       entry code to be passed to SYS13. SYS13
                    01290 ;       will be executed from SYS1 if this byte
                    01300 ;       is NON/0, bit 4, 5 and 6 will be merged into
                    01310 ;       the SYS13 (1000,1111b) overlay request
                    01320 ;
006E 00             01330 EFLAG$  DB      0               ;Flag E
006F 00             01340 FEMSK$  DB      0               ;Port FE mask
0070 00             01350         DC      2,0             ;Flags G-H
     00
                    01360 ;
                    01370 ;       IFLAG$ - Assignments: (INTERNATIONAL)
                    01380 ;        0 - FRENCH
                    01390 ;        1 - GERMAN
                    01400 ;        2 - SWISS
                    01410 ;        3 -
                    01420 ;        4 -
                    01430 ;        5 -
                    01440 ;        6 - Special DMP mode ON/OFF
                    01450 ;        7 - '7' bit mode ON/OFF
                    01460 ;
0072                01470 IFLAG$  EQU     $
                    01480         IF      @FRENCH
                    01490         DB      01000001B
                    01500         ENDIF
                    01510         IF      @GERMAN
                    01520         DB      01000010B
                    01530         ENDIF
                    01540         IF      @USA
0072 00             01550         DB      0
                    01560         ENDIF
0073 00             01570         DB      0               ;Flag J
                    01580 ;
                    01590 ;       KFLAG$ assignments:
                    01600 ;        0 - BREAK latch
                    01610 ;        1 - PAUSE latch
                    01620 ;        2 - ENTER latch
                    01630 ;        3 - reserved
                    01640 ;        4 - reserved
                    01650 ;        5 - CAPs lock
                    01660 ;        6 - reserved
                    01670 ;        7 - character in TYPE ahead
                    01680 ;
0074 00             01690 KFLAG$  DB      0               ;Keyboard flag
                    01700 ;
                    01710 ;       LFLAG$ assignments:
                    01720 ;        0 - inhibit step rate question in FORMAT
                    01730 ;        4 - inhibit 8" query in FLOPPY/DCT
                    01740 ;        5 - inhibit # sides question in FORMAT
                    01750 ;        6,7 - Reserved for IM 2 hardware
                    01760 ;
0075 31             01770 LFLAG$  DB      00110001B       ;LDOS feature inhibit
                    01780 ;
                    01790 ;       MODOUT$ mask assignments:
                    01800 ;        0 -
```

```
'System low core assignments

                    01810 ;         1 - cassette motor on/off
                    01820 ;         2 - mode select (0 = 80/64, 1 = 40/32)
                    01830 ;         3 - enable alternate character set
                    01840 ;         4 - enable external I/O
                    01850 ;         5 - video wait states (0 = disable, 1 = enable)
                    01860 ;         6 - clock speed ( 1 = 4 Mhz, 0 = 2 MHz)
                    01870 ;         7 -
                    01880 ;
                    01890           IF      @INTL
                    01900 MODOUT$   DB      70H             ;MODOUT international
                    01910           ELSE
0076 78             01920 MODOUT$   DB      78H             ;MODOUT port image (FAST)
                    01930           ENDIF
                    01940 ;
                    01950 ;
                    01960 ;         NFLAG$ - Network flag$
                    01970 ;         0 - Allow setting of file open bit in DIR
                    01980 ;         1 / 5 - Reserved
                    01990 ;         6 - Set if in Task Processor
                    02000 ;         7 - Reserved
                    02010 ;
0077 00             02020           DB      0               ;Inhibit open bit in DIR
                    02030 ;
                    02040 ;         OPREG$ memory management image port
                    02050 ;         0 - SEL0 - Select map overlay bit 0
                    02060 ;         1 - SEL1 - Select map overlay bit 1
                    02070 ;         2 - 80/64 - 1 = 80 x 24
                    02080 ;         3 - Inverse video
                    02090 ;         4 - MBIT0 - memory map bit 0
                    02100 ;         5 - MBIT1 - memory map bit 1
                    02110 ;         6 - FXUPMEM - fix upper memory
                    02120 ;         7 - PAGE - page 1K video RAM (set for 80x24)
                    02130 ;
0078 87             02140 OPREG$    DB      87H             ;Memory management image
                    02150 ;
                    02160 ;         PFLAG$ - Printer flag
                    02170 ;         7 = Printer spooler is paused
                    02180 ;         0 - 6 = Reserved
                    02190 ;
0079 00             02200           DB      0
007A 00             02210           DB      0               ;QFLAG$
                    02220 ;
                    02230 ;         RFLAG$ - Retry init for FDC driver
                    02240 ;
007B 08             02250 RFLAG$    DB      08              ;FDC retry count >=2
                    02260 ;
                    02270 ;         SFLAG$ assignments:
                    02280 ;         0 - inhibit file open bit
                    02290 ;         1 - set to 1 if bit-2 set & EXEC file opened
                    02300 ;         2 - set by @RUN to permit load of EXEC file
                    02310 ;         3 - SYSTEM (FAST)
                    02320 ;         4 - BREAK key disabled
                    02330 ;         5 - JCL active
                    02340 ;         6 - force extended error messages
                    02350 ;         7 - DEBUG to be turned on after load
                    02360 ;
007C 08             02370 SFLAG$    DB      8               ;System flag (FAST)
                    02380 ;
                    02390 ;
```

```
'System low core assignments

                        02400 ;       Machine TYPE assignment:
                        02410 ;       All values are in decimal
                        02420 ;
                        02430 ;        2 = TRS-80 Model 2
                        02440 ;        4 = TRS-80 Model 4
                        02450 ;        5 = TRS-80 MODEL 4P
                        02460 ;       12 = TRS-80 Model 12
                        02470 ;       16 = TRS-80 Model 16
                        02480 ;
                        02490         IF      @MOD4
007D 04                 02500 TFLAG$  DB      04              ;Model 4 assignment
                        02510         ELSE
                        02520         ERR     'Undefined machine TYPE for TFLAG'
                        02530         ENDIF
007E 00                 02540         DB      0               ;Flag U
                        02550 ;
                        02560 ;       Video FLAG$ assignments:
                        02570 ;        0-3 - Set blink rate (1=fastest,7=slowest)
                        02580 ;        4 - display CLOCK
                        02590 ;        5 - cursor blink toggle bit
                        02600 ;        6 - Inhibit blinking cursor (user)
                        02610 ;        7 - Inhibit blinking cursor (system)
                        02620 ;
007F 00                 02630 VFLAG$  DB      0               ;Blink,Slow,No clock
                        02640 ;
                        02650 ;       WRINT$ - interrupt mask register
                        02660 ;        0 - enable 1500 baud rising edge
                        02670 ;        1 - enable 1500 baud falling edge
                        02680 ;        2 - enable real time clock
                        02690 ;        3 - enable I/O bus interrupts
                        02700 ;        4 - enable RS-232 transmit interrupts
                        02710 ;        5 - enable RS-232 receive data interrupts
                        02720 ;        6 - enable RS-232 error interrupt
                        02730 ;
0080 04                 02740 WRINT$  DB      4               ;WRINTMASK port image
0081 00                 02750         DC      3,0             ;Flags X-Z
     00 00
                        02760 ;
                        02770 ;       Contents are high-order byte of SVC table
                        02780 ;
0084 01                 02790         DB      SVCTAB$<-8      ;MSB of SVC table
                        02800 ;
                        02810 ;       OSVER$ stores the operating system version
                        02820 ;
0085 62                 02830 OSVER$  DB      62H             ;OS version #
                        02840 ;
                        02850 ;       Vector for config initialization
                        02860 ;
0086 C9                 02870 @ICNFG  RET                     ;Initialization config
0087 0000               02880         DW      0
                        02890 ;
                        02900 ;       Chain vector for KI task processor
                        02910 ;
0089 C9                 02920 @KITSK  RET                     ;Keyboard task routine
008A 0000               02930         DW      0
                        02940 ;
                        02950 ;       System File Control Block for overlays
                        02960 ;
008C 80                 02970 SFCB$   DB      80H,0,0         ;System /SYS FCB
```

```
'System low core assignments

     00 00
008F 001D     02980          DW      SBUFF$
0091 00       02990          DB      0
0092 0000     03000          DW      0,0,0,-1,0,-1,-1
     0000 0000 FFFF 0000 FFFF FFFF
               03010 ;
               03020 ;        32-byte DEBUG save area
               03030 ;
0020           03040 DBGSV$   DS      32
               03050 ;
               03060 ;        Job Control Language File Control Block
               03070 ;
00C0 00       03080 JFCB$    DC      3,0
     00 00
00C3 001D     03090          DW      SBUFF$
001B           03100          DS      27
               03110 ;
               03120 ;        System Command Line file control block
               03130 ;
00E0           03140 CFCB$    EQU     $               ;Command Interpreter FCB
00E0 43       03150 CFGFCB$ DB      'CONFIG/SYS.CCC:0',3
     4F 4E 46 49 47 2F 53 59
     53 2E 43 43 43 3A 30 03
000F           03160          DS      15
               03170 ;
               03180 ;        Page 1 - System Supervisor Call Table
               03190 ;
0100           03200 SVCTAB$ EQU     $
               03210          IFNE $,100H
               03220          ERR     'SVCTBL location violation'
               03230          ENDIF
               03240 ;
               03250 ;        Initial version
               03260 ;
2400           03270 MAXCOR$ EQU     2400H+START$
3000           03280 MINCOR$ EQU     3000H+START$
1300           03290          ORG     @BYTEIO
               03300 ;
               03310 ;        file positioning routines - MUST BE FIRST
               03320 ;
1300           03330          SUBTTL  '<File positioning subroutines>'
```

File positioning subroutines

```
1300           03350 *GET    FILPOSN:3
               01340 ;FILPOSN/ASM - LS-DOS 6.2
               01350 ;
               01360 ;       Entry for byte I/O from @GET & @PUT
               01370 ;
1300 DDE5      01380 BYTEIO  PUSH    IX
1302 D1        01390         POP     DE              ;Transfer DCB to DE
1303 CD6815    01400         CALL    CKOPEN@         ;Ck file open, save regs
1306 DDCB01FE  01410         SET     7,(IX+1)        ;Denote byte or LRec
130A 78        01420         LD      A,B             ;Get type code & test
130B FE02      01430         CP      2               ;For get/put
130D 79        01440         LD      A,C
130E 281F      01450         JR      Z,WRCHAR        ;Go on PUT
1310 3058      01460         JR      NC,IORETZ       ;Ignore if CTL
               01470 ;
               01480 ;       Get a byte from a file
               01490 ;
1312 CD9215    01500 RDCHAR  CALL    CKEOF1          ;Ck for end of file
1315 C0        01510         RET     NZ              ;Return if at end
1316 DDCB016E  01520         BIT     5,(IX+1)        ;If buffer not current,
131A C47913    01530         CALL    NZ,NSEC1        ;  read next sector
131D C0        01540         RET     NZ
131E CD1314    01550         CALL    BFRPOS          ;Pt to byte posn in bfr
1321 1A        01560         LD      A,(DE)          ;P/u the byte
1322 DD3405    01570         INC     (IX+5)          ;Inc NEXT ptr
1325 CC2A13    01580         CALL    Z,SET5          ;Set bit 5 if zero
1328 BF        01590         CP      A               ;Set Z flag--no error
1329 C9        01600         RET
               01610 ;
132A DDCB01EE  01620 SET5     SET     5,(IX+1)
132E C9        01630         RET
               01640 ;
               01650 ;       Write a byte to a file
               01660 ;
132F DDCB0076  01670 WRCHAR   BIT     6,(IX+0)        ;Prot level give write acc?
1333 CAC613    01680         JP      Z,RWRIT3        ;  go if not
1336 F5        01690         PUSH    AF              ;Save byte
1337 DDCB016E  01700         BIT     5,(IX+1)        ;Get next sector if
133B C46C13    01710         CALL    NZ,WRCH2        ;  buffer is not current
133E 2803      01720         JR      Z,WRCH1         ;Skip if read was ok
1340 E3        01730         EX      (SP),HL         ;Pop stack but keep
1341 E1        01740         POP     HL              ;  error # in AF
1342 C9        01750         RET
               01760 ;
1343 CD1314    01770 WRCH1   CALL    BFRPOS          ;Next bfr byte posn
1346 F1        01780         POP     AF
1347 12        01790         LD      (DE),A          ;Stuff the byte
1348 DDCB01E6  01800         SET     4,(IX+1)        ;Buffer contains updated data
134C DD3405    01810         INC     (IX+5)          ;Inc NEXT byte
134F F5        01820         PUSH    AF              ;Save Z or NZ flag
1350 CC2A13    01830         CALL    Z,SET5          ;Set bit 5 if offset 0
1353 CD9215    01840         CALL    CKEOF1          ;Check for EOF
1356 2006      01850         JR      NZ,ATEOFW       ;Go if there
1358 DDCB0176  01860         BIT     6,(IX+1)        ;Jump if EOF set to next
135C 2009      01870         JR      NZ,DNTSET       ;  only if at EOF
135E DD7108    01880 ATEOFW  LD      (IX+8),C        ;Set EOF
1361 DD750C    01890         LD      (IX+12),L
1364 DD740D    01900         LD      (IX+13),H
1367 F1        01910 DNTSET  POP     AF              ;Restore offset flag
1368 2846      01920         JR      Z,RWRIT1        ;Go to write sector if 00
```

File positioning subroutines

```
136A AF        Ø193Ø IORETZ  XOR     A               ;Set Z flag--no error
136B C9        Ø194Ø         RET
               Ø195Ø ;
               Ø196Ø ;       WRCHR needs the next sector - if UPDATE, ck EOF
               Ø197Ø ;
136C DD7EØ1    Ø198Ø WRCH2   LD      A,(IX+1)        ;Ck if UPD bit set
136F E6Ø7      Ø199Ø         AND     7               ;Mask for prot level
1371 FEØ4      Ø2ØØØ         CP      4               ;Check for UPD
1373 2ØØ4      Ø2Ø1Ø         JR      NZ,NSEC1        ;Bypass EOF ck on > UPD
1375 CD9215    Ø2Ø2Ø NXTSECT CALL    CKEOF1          ;Ck for end of file
1378 CØ        Ø2Ø3Ø         RET     NZ              ;Can't extend in update mode
1379 DD7EØ1    Ø2Ø4Ø NSEC1   LD      A,(IX+1)        ;Read access?
137C E6Ø7      Ø2Ø5Ø         AND     7
137E FEØ6      Ø2Ø6Ø         CP      6
138Ø 3Ø44      Ø2Ø7Ø         JR      NC,RWRIT3       ;"Illegal access..." if not
1382 CDCB15    Ø2Ø8Ø NSEC2   CALL    IOREC           ;Calc cylinder/sector
1385 CØ        Ø2Ø9Ø         RET     NZ
1386 DDCBØ1AE  Ø21ØØ         RES     5,(IX+1)        ;Show buffer current
138A DD6EØ3    Ø211Ø         LD      L,(IX+3)        ;P/u buffer address
138D DD66Ø4    Ø212Ø         LD      H,(IX+4)
139Ø CDF419    Ø213Ø         CALL    @RDSEC          ;Read the sector
1393 28Ø3      Ø214Ø         JR      Z,BUMPNRN       ;Go if no error
1395 FEØ6      Ø215Ø         CP      6               ;Test for prot sector
1397 CØ        Ø216Ø         RET     NZ              ;Quit if error not 6
1398 DD34ØA    Ø217Ø BUMPNRN INC     (IX+1Ø)         ;Inc the NRN ptr LSB
139B 2ØØ3      Ø218Ø         JR      NZ,ZEROA@
139D DD34ØB    Ø219Ø         INC     (IX+11)         ;  and MSB if necessary
13AØ AF        Ø22ØØ ZEROA@  XOR     A
13A1 C9        Ø221Ø         RET
               Ø222Ø ;
               Ø223Ø ;       Repositioning needs to write out the buffer
               Ø224Ø ;
13A2 DD7EØ1    Ø225Ø RWRIT@  LD      A,(IX+1)
13A5 E69Ø      Ø226Ø         AND     9ØH             ;Test for non-sector i/o and
13A7 FE9Ø      Ø227Ø         CP      9ØH             ;  buffer contents changed
13A9 28Ø5      Ø228Ø         JR      Z,RWRIT1        ;Go if conditions true
13AB 18F3      Ø229Ø         JR      ZEROA@          ;  else no need to write
13AD CD6815    Ø23ØØ @RWRIT  CALL    CKOPEN@         ;Ck file open, save regs
13BØ CDØC14    Ø231Ø RWRIT1  CALL    GETNRN          ;P/u NRN
13B3 7C        Ø232Ø         LD      A,H             ;Ignore if rewound
13B4 B5        Ø233Ø         OR      L
13B5 C8        Ø234Ø         RET     Z
13B6 2B        Ø235Ø         DEC     HL              ;Dec & reset NRN
13B7 DD75ØA    Ø236Ø         LD      (IX+1Ø),L
13BA DD74ØB    Ø237Ø         LD      (IX+11),H
               Ø238Ø ;
               Ø239Ø ;       Check access protection level
               Ø24ØØ ;
13BD DD7EØ1    Ø241Ø RWRIT2  LD      A,(IX+1)        ;Get prot
13CØ E6Ø7      Ø242Ø         AND     7
13C2 FEØ5      Ø243Ø         CP      5               ;Update access or better?
13C4 38Ø4      Ø244Ø         JR      C,RWRIT4
13C6 3E25      Ø245Ø RWRIT3  LD      A,25H           ;Illegal access error code
13C8 B7        Ø246Ø         OR      A               ;Return NZ
13C9 C9        Ø247Ø         RET
               Ø248Ø ;
13CA E6Ø4      Ø249Ø RWRIT4  AND     4               ;If UPDATE access, then
13CC 28Ø5      Ø25ØØ         JR      Z,RWRIT5        ;  can't extend if at EOF
13CE CD9215    Ø251Ø         CALL    CKEOF1
```

File positioning subroutines

```
13D1 20F3      02520          JR      NZ,RWRIT3       ;  so show "Illegal access...
13D3 CDCB15    02530 RWRIT5   CALL    IOREC           ;Calculate cylinder & sector
13D6 C0        02540          RET     NZ
13D7 DD6E03    02550          LD      L,(IX+3)        ;P/u buffer addr
13DA DD6604    02560          LD      H,(IX+4)
13DD DDCB01A6  02570          RES     4,(IX+1)        ;Altered buffer flag off
13E1 DDCB00D6  02580          SET     2,(IX+0)        ;Show modification done
13E5 CDE819    02590          CALL    @WRSEC          ;  for directory mod flag
13E8 C0        02600          RET     NZ
13E9 3E00      02610 VEROP    LD      A,0             ;Verify operation if set
13EB B7        02620          OR      A
13EC C4DC19    02630          CALL    NZ,@VRSEC       ;Verify if no write error
13EF C0        02640          RET     NZ              ;Return if wrt/ver error
13F0 CD9813    02650          CALL    BUMPNRN         ;Increment NRN
               02660 ;
               02670 ;        Check if ERN to be set to NRN
               02680 ;        Should be done for byte i/o, but not random i/o
               02690 ;
13F3 CD9215    02700          CALL    CKEOF1          ;Returns 0 if not at EOF
13F6 3D        02710          DEC     A               ;Set bit 6 if retcod=0
13F7 DDA601    02720          AND     (IX+1)          ;If IX+1, bit 6 set, then
13FA E640      02730          AND     40H             ;  don't update EOF unless at
13FC 20A2      02740          JR      NZ,ZEROA@       ;  or past the old EOF
13FE DD750C    02750 YESEOF   LD      (IX+12),L       ;Update ERN
1401 DD740D    02760          LD      (IX+13),H
1404 DDCB015E  02770          BIT     3,(IX+1)        ;Test if ending '!'
1408 C2F214    02780          JP      NZ,WEOF1        ;Upd dir if so
140B C9        02790          RET
               02800 ;
140C DD6E0A    02810 GETNRN   LD      L,(IX+10)       ;Xfer NRN to HL
140F DD660B    02820          LD      H,(IX+11)
1412 C9        02830          RET
               02840 ;
1413 DD7E05    02850 BFRPOS   LD      A,(IX+5)        ;P/u byte offset in buffer
1416 DD8603    02860          ADD     A,(IX+3)        ;Add to buffer lsb
1419 5F        02870          LD      E,A
141A DD7E04    02880          LD      A,(IX+4)        ;  and adjust buffer MSB
141D CE00      02890          ADC     A,0             ;  if needed
141F 57        02900          LD      D,A             ;Return DE = posn
1420 C9        02910          RET
               02920 ;
               02930 ;        Entry to seek next record of a file
               02940 ;
1421 CD6815    02950 @SEEKSC  CALL    CKOPEN@         ;Link to FCB & ck if open
1424 CD9215    02960          CALL    CKEOF1          ;Ensure not > EOF
1427 CCCB15    02970          CALL    Z,IOREC         ;Get track/sector data
142A C0        02980          RET     NZ              ;Back on I/O error
142B CDD019    02990          CALL    @SEEK           ;Issue seek to drive
142E AF        03000          XOR     A               ;Ignore seek errors here
142F C9        03010          RET
               03020 ;
               03030 ;        Entry to Skip record routine
               03040 ;
1430 CDB314    03050 @SKIP    CALL    @LOC            ;Locate next record
1433 03        03060          INC     BC              ;Step past it
               03070 ;
               03080 ;        Entry to Position to record routine
               03090 ;
1434 CD6815    03100 @POSN    CALL    CKOPEN@
```

File positioning subroutines

```
1437 DDCBØ1F6 Ø311Ø          SET     6,(IX+1)        ;Upd eof only if NRN>EOF
143B DDCBØ17E Ø312Ø          BIT     7,(IX+1)        ;Jump if sector i/o only
143F 281D     Ø313Ø          JR      Z,POSN1
1441 6Ø       Ø314Ø          LD      H,B             ;Record ptr to HL
1442 69       Ø315Ø          LD      L,C
1443 DDB6Ø9   Ø316Ø          OR      (IX+9)          ;P/u LRL
1446 2816     Ø317Ø          JR      Z,POSN1         ;Skip nxt if LRL=256
1448 CDC9Ø6   Ø318Ø          CALL    @MUL16          ;Calc sector & offset
144B 44       Ø319Ø          LD      B,H             ;Physical sector =>BC
144C 4D       Ø32ØØ          LD      C,L
144D DD77Ø5   Ø321Ø          LD      (IX+5),A        ;Set byte ptr
145Ø DDCBØ16E Ø322Ø          BIT     5,(IX+1)        ;Jump if buffer does not
1454 2ØØB     Ø323Ø          JR      NZ,POSN2        ;  contain current sector
1456 CDØC14   Ø324Ø          CALL    GETNRN          ;P/u the NRN
1459 37       Ø325Ø          SCF
145A ED42     Ø326Ø          SBC     HL,BC
145C 2812     Ø327Ø          JR      Z,$CKEOF        ;Pass on to CKEOF
145E DD77Ø5   Ø328Ø POSN1    LD      (IX+5),A        ;Offset in buffer
1461 C5       Ø329Ø POSN2    PUSH    BC
1462 CDA213   Ø33ØØ POSN2A   CALL    RWRIT@          ;Write current if needed
1465 C1       Ø331Ø          POP     BC              ;  before moving
1466 CØ       Ø332Ø          RET     NZ              ;Back on write error
1467 DD71ØA   Ø333Ø          LD      (IX+1Ø),C       ;NRN
146A DD7ØØB   Ø334Ø          LD      (IX+11),B
146D CD2A13   Ø335Ø          CALL    SET5            ;Show bfr does not
147Ø C39215   Ø336Ø $CKEOF   JP      CKEOF1          ;  contain current sector
              Ø337Ø ;
              Ø338Ø ;        Entry to force a physical read
              Ø339Ø ;
1473 CD6815   Ø34ØØ @RREAD   CALL    CKOPEN@
1476 ØEØ1     Ø341Ø          LD      C,1             ;Cause ADJUST to bump
              Ø342Ø ;                                ;  NRN when called
1478 CDØC14   Ø343Ø BKSP1    CALL    GETNRN          ;Get current record #
147B 7C       Ø344Ø          LD      A,H             ;If file is rewound,
147C B5       Ø345Ø          OR      L               ;  then ignore the req
147D 2815     Ø346Ø          JR      Z,BKSPØ         ;  & force OFFSET = Ø
147F 2B       Ø347Ø          DEC     HL              ;Back up by 1
148Ø CDBA15   Ø348Ø          CALL    ADJ2            ;RET if sector I/O only,
              Ø349Ø                                  ;  else bump fwd if RREAD
              Ø35ØØ                                  ;  then back up if bit 5=Ø
1483 E5       Ø351Ø          PUSH    HL              ;Will be popped into BC
1484 18DC     Ø352Ø          JR      POSN2A          ;Finish the job
              Ø353Ø ;
              Ø354Ø ;        Entry to backspace one logical record
              Ø355Ø ;
1486 CD6815   Ø356Ø @BKSP    CALL    CKOPEN@
1489 4F       Ø357Ø          LD      C,A             ;Keep ADJUST from bumping
148A DD46Ø9   Ø358Ø          LD      B,(IX+9)        ;P/u LRL
148D BØ       Ø359Ø          OR      B               ;Is it a Ø
148E 28E8     Ø36ØØ          JR      Z,BKSP1         ;Go if so
149Ø DD7EØ5   Ø361Ø          LD      A,(IX+5)        ;P/u next byte pointer
1493 9Ø       Ø362Ø          SUB     B               ;Sub one record length
1494 DD77Ø5   Ø363Ø BKSPØ    LD      (IX+5),A
1497 38DF     Ø364Ø          JR      C,BKSP1         ;Go if crossed sec bdry
1499 AF       Ø365Ø          XOR     A               ;  else all done
149A C9       Ø366Ø          RET
              Ø367Ø ;
              Ø368Ø ;        Entry to Rewind to beginning
              Ø369Ø ;
```

File positioning subroutines

```
149B CD6815   Ø37ØØ @REW    CALL    CKOPEN@
149E 47       Ø371Ø         LD      B,A             ;Zero NRN
149F 4F       Ø372Ø         LD      C,A
14AØ 18BC     Ø373Ø         JR      POSN1           ;Will also zero offset
              Ø374Ø ;
              Ø375Ø ;       Entry to Position to end-of-file
              Ø376Ø ;
14A2 CD6815   Ø377Ø @PEOF   CALL    CKOPEN@
14A5 DD4EØC   Ø378Ø         LD      C,(IX+12)       ;ERN to BC
14A8 DD46ØD   Ø379Ø         LD      B,(IX+13)
14AB DDB6Ø8   Ø38ØØ         OR      (IX+8)          ;P/u EOF byte
14AE 28AE     Ø381Ø         JR      Z,POSN1         ;Go if full sector
14BØ ØB       Ø382Ø         DEC     BC              ;Point to last rec
14B1 18AB     Ø383Ø         JR      POSN1           ;Use POSN to get end
              Ø384Ø ;
              Ø385Ø ;       Entry to Locate current record number
              Ø386Ø ;
14B3 CD6815   Ø387Ø @LOC    CALL    CKOPEN@
14B6 CDØC14   Ø388Ø         CALL    GETNRN          ;P/u NRN
14B9 CDB715   Ø389Ø         CALL    ADJUST          ;Get offset and adj NRN
14BC DD5EØ9   Ø39ØØ LOC1    LD      E,(IX+9)        ;P/u LRL
14BF 7B       Ø391Ø         LD      A,E             ;Test LRL for zero
14CØ B7       Ø392Ø         OR      A               ;If zero, then give NRN
14C1 2816     Ø393Ø         JR      Z,LOC3          ;LRL=Ø, NRN is correct
14C3 ØC       Ø394Ø         INC     C               ;If offset is zero,
14C4 ØD       Ø395Ø         DEC     C               ;  then it's at 256,
14C5 28Ø1     Ø396Ø         JR      Z,LOC2          ;  and we don't dec NRN
14C7 2B       Ø397Ø         DEC     HL
              Ø398Ø ;
              Ø399Ø ;       Divide the three byte pointer (HLC) by the LRL
              Ø4ØØØ ;
14C8 CDE3Ø6   Ø4Ø1Ø LOC2    CALL    @DIV16          ;Divide (NRN-1)/LRL
14CB 45       Ø4Ø2Ø         LD      B,L             ;Save high order result
14CC 54       Ø4Ø3Ø         LD      D,H             ;Save possible overflow
14CD 67       Ø4Ø4Ø         LD      H,A             ;Prepare 2nd dividend
14CE 69       Ø4Ø5Ø         LD      L,C             ;P/u low order dividend
14CF 7B       Ø4Ø6Ø         LD      A,E             ;P/u LRL divisor again
14DØ CDE3Ø6   Ø4Ø7Ø         CALL    @DIV16
14D3 6Ø       Ø4Ø8Ø         LD      H,B             ;Xfer high order result
14D4 B7       Ø4Ø9Ø         OR      A               ;If remainder, we have a
14D5 28Ø1     Ø41ØØ         JR      Z,$+3           ;  partial record to round
14D7 23       Ø411Ø         INC     HL              ;  up to next record #
14D8 7A       Ø412Ø         LD      A,D             ;Xfer possible overflow
14D9 C1       Ø413Ø LOC3    POP     BC              ;Pop RESTREG return adr
14DA E3       Ø414Ø         EX      (SP),HL         ;Exchange value with BC
14DB C5       Ø415Ø         PUSH    BC              ;Restore RESTREG
              Ø416Ø ;
              Ø417Ø         IF      @MOD4
14DC          Ø418Ø ORARET@ EQU     $
              Ø419Ø         ENDIF
14DC B7       Ø42ØØ         OR      A
14DD C9       Ø421Ø         RET
              Ø422Ø ;
              Ø423Ø ;       Entry to Locate the end-of-file record
              Ø424Ø ;
14DE CD6815   Ø425Ø @LOF    CALL    CKOPEN@
14E1 DD6EØC   Ø426Ø         LD      L,(IX+12)       ;P/u ERN
14E4 DD66ØD   Ø427Ø         LD      H,(IX+13)
14E7 DD4EØ8   Ø428Ø         LD      C,(IX+8)        ;EOF byte
```

File positioning subroutines

```
14EA 18DØ     Ø429Ø          JR      LOC1            ;Handle all LRLs
              Ø43ØØ ;
              Ø431Ø ;        Entry to Write an end-of-file mark
              Ø432Ø ;
14EC CD6815   Ø433Ø @WEOF    CALL    CKOPEN@
14EF CDA213   Ø434Ø          CALL    RWRIT@          ;Write buffer if needed
14F2 DD46Ø7   Ø435Ø WEOF1    LD      B,(IX+7)        ;P/u DEC of FPDE
14F5 DD4EØ6   Ø436Ø          LD      C,(IX+6)        ;P/u drive #
14F8 CDBB18   Ø437Ø          CALL    @DIRRD          ;Read file's dir record
14FB CØ       Ø438Ø          RET     NZ              ;Back if read error
14FC 2C       Ø439Ø          INC     L               ;Pt to ERN offset
14FD 2C       Ø44ØØ          INC     L
14FE 2C       Ø441Ø          INC     L
14FF DD7EØ8   Ø442Ø          LD      A,(IX+8)        ;P/u EOF offset
15Ø2 77       Ø443Ø          LD      (HL),A          ;Put in direc
15Ø3 111ØØ    Ø444Ø          LD      DE,17           ;Pt to EOF in dir
15Ø6 19       Ø445Ø          ADD     HL,DE
15Ø7 DD7EØC   Ø446Ø          LD      A,(IX+12)       ;P/u lo EOF
15ØA 77       Ø447Ø          LD      (HL),A          ;Put EOF in direc
15ØB 23       Ø448Ø          INC     HL
15ØC DD7EØD   Ø449Ø          LD      A,(IX+13)       ;P/u hi EOF
15ØF 77       Ø45ØØ          LD      (HL),A
151Ø C3Ø318   Ø451Ø          JP      @DIRWR          ;Write direc and return
              Ø452Ø ;
              Ø453Ø ;        Entry to Read a record
              Ø454Ø ;
1513 CD6815   Ø455Ø @READ    CALL    CKOPEN@
1516 E5       Ø456Ø          PUSH    HL
1517 CDA213   Ø457Ø          CALL    RWRIT@          ;Write buffer if needed
151A E1       Ø458Ø          POP     HL
151B CØ       Ø459Ø          RET     NZ              ;Back on write error
151C DD46Ø9   Ø46ØØ          LD      B,(IX+9)        ;P/u LRL
151F 78       Ø461Ø          LD      A,B             ;If LRL=256, just
152Ø B7       Ø462Ø          OR      A
1521 CA7513   Ø463Ø          JP      Z,NXTSECT       ;  get the next sector
1524 E5       Ø464Ø RDREC    PUSH    HL              ;Save buffer posn
1525 C5       Ø465Ø          PUSH    BC              ;Save LRL
1526 CD1213   Ø466Ø          CALL    RDCHAR          ;Read next byte
1529 C1       Ø467Ø          POP     BC
152A E1       Ø468Ø          POP     HL
152B CØ       Ø469Ø          RET     NZ              ;Back on read error
152C 77       Ø47ØØ          LD      (HL),A          ;Put char into buffer
152D 23       Ø471Ø          INC     HL              ;Bump buffer ptr
152E 1ØF4     Ø472Ø          DJNZ    RDREC           ;Loop for entire record
153Ø C9       Ø473Ø          RET
              Ø474Ø ;
              Ø475Ø ;        Entry to Write a record
              Ø476Ø ;
1531 CD6815   Ø477Ø @WRITE   CALL    CKOPEN@
1534 32EA13   Ø478Ø WRIT1    LD      (VEROP+1),A     ;Turn on/off verify
1537 DD46Ø9   Ø479Ø          LD      B,(IX+9)        ;P/u LRL
153A 78       Ø48ØØ          LD      A,B             ;Bypass if LRL=256
153B B7       Ø481Ø          OR      A
153C CABD13   Ø482Ø          JP      Z,RWRIT2
153F E5       Ø483Ø          PUSH    HL              ;Save some FCB values
154Ø DD66Ø5   Ø484Ø          LD      H,(IX+5)        ;P/u buffer offset loc
1543 DD6EØ8   Ø485Ø          LD      L,(IX+8)        ;P/U EOF offset byte
1546 E3       Ø486Ø          EX      (SP),HL         ;Put values on stack
              Ø487Ø                                  ;  and recover HL
```

File positioning subroutines

```
1547 7E       Ø488Ø WRREC    LD      A,(HL)          ;Pass the logical record
1548 23       Ø489Ø          INC     HL              ;  to the writing routine
1549 E5       Ø49ØØ          PUSH    HL              ;  byte by byte
154A C5       Ø491Ø          PUSH    BC
154B CD2F13   Ø492Ø          CALL    WRCHAR
154E C1       Ø493Ø          POP     BC
154F E1       Ø494Ø          POP     HL
155Ø 2ØØ5     Ø495Ø          JR      NZ,WRERROR      ;Exit and fix FCB
1552 1ØF3     Ø496Ø          DJNZ    WRREC           ;Loop for entire record
1554 E3       Ø497Ø          EX      (SP),HL         ;Remove stored FCB info
1555 E1       Ø498Ø          POP     HL              ;Recover HL
1556 C9       Ø499Ø          RET
1557 E3       Ø5ØØØ WRERROR  EX      (SP),HL         ;Get FCB Values
1558 DD74Ø5   Ø5Ø1Ø          LD      (IX+5),H        ;  and put them back
155B DD75Ø8   Ø5Ø2Ø          LD      (IX+8),L
155E E1       Ø5Ø3Ø          POP     HL              ;Restore HL
155F C9       Ø5Ø4Ø          RET                     ;Go back with error
              Ø5Ø5Ø ;
              Ø5Ø6Ø ;        Entry to Verify after write of a record
              Ø5Ø7Ø ;
156Ø CD6815   Ø5Ø8Ø @VER     CALL    CKOPEN@
1563 3C       Ø5Ø9Ø          INC     A               ;Set verify byte
1564 18CE     Ø51ØØ          JR      WRIT1
1566 37       Ø511Ø LNKFCB@  SCF                     ;Init to force file open
1567 D2       Ø512Ø          DB      ØD2H            ;  test by JP NC,aaaa
1568 1A       Ø513Ø CKOPEN@  LD      A,(DE)          ;Ignore if from LNKFCB
1569 Ø7       Ø514Ø          RLCA                    ;Test hi bit of FCB
156A E3       Ø515Ø          EX      (SP),HL
156B 2226ØØ   Ø516Ø          LD      (JRET$),HL      ;Save ret
156E ED532４ØØ Ø517Ø          LD      (JDCB$),DE      ;Save DCB
1572 E3       Ø518Ø          EX      (SP),HL
1573 3ØØF     Ø519Ø          JR      NC,NOTOPEN      ;Go if not an open FCB
1575 F1       Ø52ØØ          POP     AF              ;Get return
1576 D5       Ø521Ø          PUSH    DE              ;Dcb addr to IX
1577 DDE3     Ø522Ø          EX      (SP),IX
1579 E5       Ø523Ø          PUSH    HL              ;Save regs
157A D5       Ø524Ø          PUSH    DE
157B C5       Ø525Ø          PUSH    BC
157C E5       Ø526Ø          PUSH    HL              ;Estab ret
157D 218915   Ø527Ø          LD      HL,RESTREG      ;  to restore registers
158Ø E3       Ø528Ø          EX      (SP),HL
1581 F5       Ø529Ø          PUSH    AF              ;Put back ret
1582 AF       Ø53ØØ          XOR     A
1583 C9       Ø531Ø          RET                     ;Go back
              Ø532Ø ;
1584 F1       Ø533Ø NOTOPEN  POP     AF
1585 3E26     Ø534Ø          LD      A,26H           ;File not open
1587 B7       Ø535Ø          OR      A
1588 C9       Ø536Ø          RET
              Ø537Ø ;
1589 C1       Ø538Ø RESTREG  POP     BC              ;Pop back registers save
158A D1       Ø539Ø          POP     DE              ;  in CKOPEN@
158B E1       Ø54ØØ          POP     HL
158C DDE1     Ø541Ø          POP     IX
158E C9       Ø542Ø          RET
              Ø543Ø ;
              Ø544Ø ;        Entry to Check if at end-of-file
              Ø545Ø ;
158F CD6815   Ø546Ø @CKEOF   CALL    CKOPEN@
```

File positioning subroutines

```
1592 CDØC14    Ø547Ø CKEOF1  CALL    GETNRN          ;P/U NRN into HL
1595 E5        Ø548Ø         PUSH    HL              ;Save un-adjusted NRN
1596 CDB715    Ø549Ø         CALL    ADJUST          ;Adjust for special cases
1599 7C        Ø55ØØ         LD      A,H             ;Compare hi byte
159A DDBEØD    Ø551Ø         CP      (IX+13)
159D 2ØØE      Ø552Ø         JR      NZ,CKEOF2       ;Go if not equal
159F 7D        Ø553Ø         LD      A,L             ;Compare lo byte
15AØ DDBEØC    Ø554Ø         CP      (IX+12)
15A3 2ØØ8      Ø555Ø         JR      NZ,CKEOF2       ;Go if not equal
15A5 ØD        Ø556Ø         DEC     C               ;Adjust for ØØ=256
15A6 DD7EØ8    Ø557Ø         LD      A,(IX+8)        ;Compare offset byte
15A9 3D        Ø558Ø         DEC     A
15AA 91        Ø559Ø         SUB     C
15AB 3F        Ø56ØØ         CCF
15AC Ø3        Ø561Ø         INC     BC              ;Restore old C value
15AD E1        Ø562Ø CKEOF2  POP     HL              ;Restore unadjusted NRN
15AE 3E1D      Ø563Ø         LD      A,1DH           ;Rec # out of range code
15BØ 2ØØ2      Ø564Ø         JR      NZ,CKEOF3       ;Go if not at EOF
15B2 3D        Ø565Ø         DEC     A               ;X'1C'=EOF encountered
15B3 C9        Ø566Ø         RET                     ;Return with NZ flag
15B4 DØ        Ø567Ø CKEOF3  RET     NC              ;Return with error
15B5 AF        Ø568Ø         XOR     A               ;No error
15B6 C9        Ø569Ø         RET
               Ø57ØØ ;
               Ø571Ø ;       File positioning adjustment routines
               Ø572Ø ;
15B7           Ø573Ø ADJUST  EQU     $               ;Entry from @CKEOF & @LOC
15B7 DD4EØ5    Ø574Ø         LD      C,(IX+5)        ;Pick up offset
15BA           Ø575Ø ADJ2    EQU     $               ;Entry from @BKSP/@RREAD
15BA DDCBØ17E  Ø576Ø         BIT     7,(IX+1)        ;Sector I/O only?
15BE C8        Ø577Ø         RET     Z               ;No adjustment if so
15BF 79        Ø578Ø         LD      A,C             ;Offset =Ø? (or "RREAD?")
15CØ B7        Ø579Ø         OR      A
15C1 28Ø1      Ø58ØØ         JR      Z,$+3           ;Go if zero
15C3 23        Ø581Ø         INC     HL              ;Adjust
15C4 DDCBØ16E  Ø582Ø         BIT     5,(IX+1)        ;Check magic bit
15C8 CØ        Ø583Ø         RET     NZ              ;Go if set
15C9 2B        Ø584Ø         DEC     HL              ;Adjust
15CA C9        Ø585Ø         RET
               Ø586Ø ;
               Ø587Ø ;       Calculate the cylinder/sector of needed record
               Ø588Ø ;
15CB CDØC14    Ø589Ø IOREC   CALL    GETNRN          ;P/u record number
15CE CD261A    Ø59ØØ         CALL    @DCTBYT-5       ;Get # of sectors/gran
15D1 E61F      Ø591Ø         AND     1FH
15D3 3C        Ø592Ø         INC     A
15D4 CDE3Ø6    Ø593Ø         CALL    @DIV16          ;By # of sectors/gran
15D7 326Ø16    Ø594Ø         LD      (CALS5+1),A     ;Sv rmndr (sector offset)
15DA DDE5      Ø595Ø         PUSH    IX              ;Xfer fcb to HL
15DC E3        Ø596Ø         EX      (SP),HL
15DD Ø1ØEØØ    Ø597Ø         LD      BC,14           ;Pt to 1st extent info
15EØ Ø9        Ø598Ø         ADD     HL,BC
15E1 C1        Ø599Ø         POP     BC              ;Pop gran ptr HL into BC
15E2 3EØ5      Ø6ØØØ         LD      A,5             ;Init to ck 4 extents
15E4 11ØØØØ    Ø6Ø1Ø         LD      DE,Ø            ;  & extended FXDE ptr
15E7 F5        Ø6Ø2Ø GREC1   PUSH    AF
15E8 7E        Ø6Ø3Ø         LD      A,(HL)          ;P/u starting cyl byte
15E9 23        Ø6Ø4Ø         INC     HL              ;  & bypass if FF
15EA 3C        Ø6Ø5Ø         INC     A
```

File positioning subroutines

```
15EB 28ØB     Ø6Ø6Ø          JR      Z,GREC2
15ED E5       Ø6Ø7Ø          PUSH    HL              ;Xfer the # of grans up
15EE 62       Ø6Ø8Ø          LD      H,D             ;  to but not including
15EF 6B       Ø6Ø9Ø          LD      L,E             ;  this extent into HL
15FØ AF       Ø61ØØ          XOR     A               ;Sub gran pointer from
15F1 ED42     Ø611Ø          SBC     HL,BC           ;  cumulative figure & go
15F3 38ØE     Ø612Ø          JR      C,GREC3         ;  if not in previous ext
15F5 E1       Ø613Ø          POP     HL
15F6 2829     Ø614Ø          JR      Z,CALCSEC
15F8 23       Ø615Ø GREC2    INC     HL
15F9 F1       Ø616Ø          POP     AF
15FA 3D       Ø617Ø          DEC     A
15FB 2819     Ø618Ø          JR      Z,GREC4         ;Jump when all quads c'kd
15FD 5E       Ø619Ø          LD      E,(HL)          ;P/u cumulative # grans
15FE 23       Ø62ØØ          INC     HL              ;  up to but not
15FF 56       Ø621Ø          LD      D,(HL)          ;  including this extent
16ØØ 23       Ø622Ø          INC     HL
16Ø1 18E4     Ø623Ø          JR      GREC1
16Ø3 24       Ø624Ø GREC3    INC     H               ;Within 256 grans?
16Ø4 7D       Ø625Ø          LD      A,L             ;Xfer lo-order difference
16Ø5 E1       Ø626Ø          POP     HL              ;Rcvr # of contig grans
              Ø627Ø ;                                ;  in this extent
16Ø6 2ØFØ     Ø628Ø          JR      NZ,GREC2        ;Go if not within 256
16Ø8 D5       Ø629Ø          PUSH    DE              ;Save cumulative count
16Ø9 5F       Ø63ØØ          LD      E,A             ;Xfer gran dif (neg)
16ØA 7E       Ø631Ø          LD      A,(HL)          ;P/u # of grans
16ØB E61F     Ø632Ø          AND     1FH             ;  in this extent
16ØD 83       Ø633Ø          ADD     A,E             ;Add to negative diff
16ØE 7B       Ø634Ø          LD      A,E             ;Put neg diff into A
16ØF D1       Ø635Ø          POP     DE
161Ø 3ØE6     Ø636Ø          JR      NC,GREC2        ;Go if not in this extent
1612 ED44     Ø637Ø          NEG                     ;Is in this extent, make
1614 18ØB     Ø638Ø          JR      CALCSEC         ;  diff positive & use it
              Ø639Ø ;
              Ø64ØØ ;        All current quads checked - Need directory info
              Ø641Ø ;
              Ø642Ø GREC4
1616 CD6416   Ø643Ø          CALL    ALLOC           ;Get # of grans
1619 CØ       Ø644Ø          RET     NZ              ;  into the extent
161A 325Ø16   Ø645Ø          LD      (CALS4+1),A     ;  or error RET
161D 3Ø2A     Ø646Ø          JR      NC,CALS3        ;Jp if record in 1st ext
161F 181F     Ø647Ø          JR      CALS1           ;  else jp if in another
              Ø648Ø ;
              Ø649Ø ;        Calc sector in gran
              Ø65ØØ ;
1621 325Ø16   Ø651Ø CALCSEC  LD      (CALS4+1),A     ;Stuff # grans into
1624 46       Ø652Ø          LD      B,(HL)          ;  this extent
1625 2B       Ø653Ø          DEC     HL              ;P/u # contig grans &
1626 4E       Ø654Ø          LD      C,(HL)          ;  rel start & start cyl
1627 23       Ø655Ø          INC     HL
1628 F1       Ø656Ø          POP     AF              ;Rcvr # of quad
1629 2F       Ø657Ø          CPL
162A C6Ø4     Ø658Ø          ADD     A,4
162C 3Ø19     Ø659Ø          JR      NC,CALS2        ;Jump if 1st ext or quad
162E 3C       Ø66ØØ          INC     A               ;If not 1st, set up to move
162F Ø7       Ø661Ø          RLCA                    ;  matching quad to the
163Ø Ø7       Ø662Ø          RLCA                    ;  first position by
1631 C5       Ø663Ø          PUSH    BC              ;  shuffling the others up
1632 D5       Ø664Ø          PUSH    DE
```

File positioning subroutines

```
1633 4F       06650          LD      C,A             ;Get bytes to move
1634 0600     06660          LD      B,0
1636 EB       06670          EX      DE,HL           ;DE = top of last quad
1637 21FCFF   06680          LD      HL,-4
163A 19       06690          ADD     HL,DE           ;HL = top of next lower
163B EDB8     06700          LDDR                    ;Do the shuffle
163D EB       06710          EX      DE,HL
163E D1       06720          POP     DE
163F C1       06730          POP     BC
1640 70       06740 CALS1    LD      (HL),B          ;Move info on matching quad
1641 2B       06750          DEC     HL              ;  into position
1642 71       06760          LD      (HL),C
1643 2B       06770          DEC     HL
1644 72       06780          LD      (HL),D
1645 2B       06790          DEC     HL
1646 73       06800          LD      (HL),E
1647 60       06810 CALS2    LD      H,B             ;Xfer start & contig gran
1648 69       06820          LD      L,C             ;Xfer start cylinder
1649 7C       06830 CALS3    LD      A,H
164A 07       06840          RLCA                    ;P/u start gran on track
164B 07       06850          RLCA
164C 07       06860          RLCA
164D E607     06870          AND     7
164F C600     06880 CALS4    ADD     A,0             ;P/u # grans into extent
1651 CD1919   06890          CALL    RELCYL          ;Calc 1st relative cyl
1654 85       06900          ADD     A,L             ;Add starting cylinder
1655 57       06910          LD      D,A
1656 78       06920          LD      A,B             ;Rcvr # sectors/gran
1657 E61F     06930          AND     1FH
1659 3C       06940          INC     A
165A D5       06950          PUSH    DE              ;Calculate sector offset
165B CD0A19   06960          CALL    @MUL8           ;  into desired cylinder
165E D1       06970          POP     DE              ;  for desired granule
165F C600     06980 CALS5    ADD     A,0             ;P/u # of excess sectors
1661 5F       06990          LD      E,A             ;  over even gran & add
1662 AF       07000          XOR     A               ;  to granule sector
1663 C9       07010          RET
              07020 ;
              07030 ;        On entry, gran needed is in BC
              07040 ;
1664 CDAE16   07050 ALLOC    CALL    CYL_GRN         ;Find ext cntng gran
1667 C0       07060          RET     NZ              ;Ret on error
1668 E5       07070          PUSH    HL              ;Save starting cyl & gran
1669 60       07080          LD      H,B             ;Xfer granule needed to
166A 69       07090          LD      L,C             ;  HL then calculate how
166B AF       07100          XOR     A               ;  many grans into this
166C ED52     07110          SBC     HL,DE           ;  extent is the desired
166E 7D       07120          LD      A,L             ;  granule
166F 32A716   07130          LD      (ALL6+1),A      ;Stuff rel gran from
1672 E1       07140          POP     HL              ;  start of extent
1673 D5       07150          PUSH    DE              ;Save granule count
1674 DDE5     07160          PUSH    IX              ;  to extent
1676 E3       07170          EX      (SP),HL         ;FCB pointer to HL
1677 110E00   07180          LD      DE,14           ;Pt to 1st alloc in FCB
167A 19       07190          ADD     HL,DE
167B D1       07200          POP     DE              ;Pop starting cylinder
167C 0605     07210          LD      B,5             ;  to this extent
167E 7E       07220 ALL1     LD      A,(HL)          ;P/u a cyl
167F 23       07230          INC     HL              ;Does starting cyl of
```

File positioning subroutines

```
168Ø BB       Ø724Ø          CP      E               ;  needed gran alloc
1681 2ØØ6     Ø725Ø          JR      NZ,ALL2         ;  appear in this extent?
1683 7E       Ø726Ø          LD      A,(HL)          ;Now see if needed gran is
1684 AA       Ø727Ø          XOR     D               ;  in this extent field
1685 E6EØ     Ø728Ø          AND     ØEØH            ;  by checking its starting gran
1687 2819     Ø729Ø          JR      Z,ALL4
1689 Ø5       Ø73ØØ ALL2     DEC     B               ;Dec the count down loop
168A 28Ø5     Ø731Ø          JR      Z,ALL3          ;Done if no match
168C 23       Ø732Ø          INC     HL              ;Go to next extent
168D 23       Ø733Ø          INC     HL              ;  info in FCB
168E 23       Ø734Ø          INC     HL
168F 18ED     Ø735Ø          JR      ALL1
1691 D5       Ø736Ø ALL3     PUSH    DE              ;Save needed extent info
1692 EB       Ø737Ø          EX      DE,HL           ;Set up to shuffle extent
1693 21FCFF   Ø738Ø          LD      HL,-4           ;  info
1696 19       Ø739Ø          ADD     HL,DE
1697 Ø1ØCØØ   Ø74ØØ          LD      BC,12
169A EDB8     Ø741Ø          LDDR
169C EB       Ø742Ø          EX      DE,HL
169D C1       Ø743Ø          POP     BC
169E AF       Ø744Ø          XOR     A               ;Set Z, no error
169F 37       Ø745Ø          SCF                     ;Set CF, extent not found
16AØ 18Ø3     Ø746Ø          JR      ALL5
16A2 72       Ø747Ø ALL4     LD      (HL),D
16A3 EB       Ø748Ø          EX      DE,HL
16A4 AF       Ø749Ø          XOR     A               ;Set Z no error
16A5 D1       Ø75ØØ ALL5     POP     DE
16A6 3EØØ     Ø751Ø ALL6     LD      A,Ø             ;# of grans into this ext
16A8 C9       Ø752Ø          RET                     ;Where desired gran is
              Ø753Ø ;
              Ø754Ø ;        Extent is unused - need to allocate more space
              Ø755Ø ;
16A9 CDF216   Ø756Ø CGØ6     CALL    CGØ7            ;Try to allocate more
16AC C1       Ø757Ø          POP     BC              ;Get back desired gran
16AD CØ       Ø758Ø          RET     NZ              ;Return on error
              Ø759Ø                                  ;Look for gran again
              Ø76ØØ ;
              Ø761Ø ;        Find extent containing desired granule
              Ø762Ø ;
16AE C5       Ø763Ø CYL_GRN  PUSH    BC              ;Save desired gran #
16AF 11ØØØØ   Ø764Ø          LD      DE,Ø            ;Init gran counter
16B2 DD46Ø7   Ø765Ø          LD      B,(IX+7)        ;P/u DEC of file
16B5 78       Ø766Ø CGØ1     LD      A,B
16B6 32AC17   Ø767Ø          LD      (STUFDEC+1),A   ;Stuff
16B9 DD4EØ6   Ø768Ø          LD      C,(IX+6)        ;P/u drive for file
16BC CDBB18   Ø769Ø          CALL    @DIRRD          ;Read its directory
16BF Ø116ØØ   Ø77ØØ          LD      BC,22           ;Point to 1st extent
16C2 Ø9       Ø771Ø          ADD     HL,BC           ;  of its directory
16C3 EB       Ø772Ø          EX      DE,HL           ;Gran count to HL
16C4 C1       Ø773Ø          POP     BC              ;Restore desired gran
16C5 CØ       Ø774Ø          RET     NZ              ;Return on read error
16C6 1A       Ø775Ø CGØ2     LD      A,(DE)          ;Is this extent
16C7 FEFE     Ø776Ø          CP      ØFEH            ;  allocated?
16C9 3Ø1F     Ø777Ø          JR      NC,CGØ5         ;Jump if it is not
16CB 13       Ø778Ø          INC     DE              ;Point to allocation
16CC 1A       Ø779Ø          LD      A,(DE)          ;P/u relative gran & #
16CD E5       Ø78ØØ          PUSH    HL              ;  of contiguous grans
16CE E61F     Ø781Ø          AND     1FH             ;Keep contiguous grans
16DØ 3C       Ø782Ø          INC     A               ;  & bump for Ø offset
```

File positioning subroutines

```
16D1 85       Ø783Ø          ADD     A,L             ;Add to count in HL
16D2 6F       Ø784Ø          LD      L,A
16D3 3ØØ1     Ø785Ø          JR      NC,CGØ3
16D5 24       Ø786Ø          INC     H               ;Bump hi order
16D6 E5       Ø787Ø CGØ3     PUSH    HL              ;Save gran count to
16D7 2B       Ø788Ø          DEC     HL              ;  end of extent
16D8 AF       Ø789Ø          XOR     A               ;Test if EOF is in this
16D9 ED42     Ø79ØØ          SBC     HL,BC           ;  allocation
16DB E1       Ø791Ø          POP     HL
16DC 3ØØ4     Ø792Ø          JR      NC,CGØ4         ;EOF not > this alloc
16DE 13       Ø793Ø          INC     DE              ;Get rid of old
16DF F1       Ø794Ø          POP     AF              ;  current quantity
16EØ 18E4     Ø795Ø          JR      CGØ2            ;Check next extent
              Ø796Ø ;
              Ø797Ø ;        The EOF is within this allocation. Recover
              Ø798Ø ;        the allocation data and exit
              Ø799Ø ;
16E2 E1       Ø8ØØØ CGØ4     POP     HL              ;P/u gran count to extent
16E3 EB       Ø8Ø1Ø          EX      DE,HL           ;Gran count to DE
16E4 7E       Ø8Ø2Ø          LD      A,(HL)          ;P/u granule data
16E5 2B       Ø8Ø3Ø          DEC     HL
16E6 6E       Ø8Ø4Ø          LD      L,(HL)          ;P/u starting cylinder
16E7 67       Ø8Ø5Ø          LD      H,A
16E8 AF       Ø8Ø6Ø          XOR     A
16E9 C9       Ø8Ø7Ø          RET
              Ø8Ø8Ø ;
              Ø8Ø9Ø ;        This extent is 1) unused, or 2) FXDE pointer
              Ø81ØØ ;        and the needed gran has not been found yet
              Ø811Ø ;
16EA C5       Ø812Ø CGØ5     PUSH    BC              ;Gran count to DE &
16EB EB       Ø813Ø          EX      DE,HL           ;DIR ptr to HL
16EC 2ØBB     Ø814Ø          JR      NZ,CGØ6         ;Jump if unused
16EE 23       Ø815Ø          INC     HL              ;Point to DEC of FXDE
16EF 46       Ø816Ø          LD      B,(HL)          ;P/u the DEC
16FØ 18C3     Ø817Ø          JR      CGØ1            ;  & loop
              Ø818Ø ;
              Ø819Ø ;        See if the drive has enough free space left
              Ø82ØØ ;
16F2 C5       Ø821Ø CGØ7     PUSH    BC              ;Save needed gran
16F3 DD4EØ6   Ø822Ø          LD      C,(IX+6)        ;P/u file's drive
16F6 CD7418   Ø823Ø          CALL    @GATRD          ;Get GAT
16F9 C1       Ø824Ø          POP     BC              ;Rcvr needed gran
16FA CØ       Ø825Ø          RET     NZ              ;Return if GAT error
16FB E5       Ø826Ø          PUSH    HL
16FC 6Ø       Ø827Ø          LD      H,B             ;Xfer the requested
16FD 69       Ø828Ø          LD      L,C             ;  gran to HL &
16FE AF       Ø829Ø          XOR     A               ;  subtract current gran
16FF ED52     Ø83ØØ          SBC     HL,DE           ;Count to calculate how
17Ø1 44       Ø831Ø          LD      B,H             ;  many excess grans
17Ø2 4D       Ø832Ø          LD      C,L             ;  are needed
17Ø3 Ø3       Ø833Ø          INC     BC
17Ø4 D1       Ø834Ø          POP     DE              ;Rcvr dir byte ptr
17Ø5 13       Ø835Ø          INC     DE              ;Pt to next DIR byte
17Ø6 2623     Ø836Ø          LD      H,DIRBUF$<-8    ;Start looking at TRK #1
17Ø8 3A6AØØ   Ø837Ø          LD      A,(AFLAG$)      ;P/u Search start CYL
17ØB 6F       Ø838Ø          LD      L,A             ;  and put it in L
17ØC C5       Ø839Ø          PUSH    BC              ;Save excess grans needed
17ØD 7B       Ø84ØØ          LD      A,E             ;Is this extent the 1st?
17ØE E61E     Ø841Ø          AND     1EH             ;Jump if so, else we can
```

File positioning subroutines

```
171Ø FE16     Ø842Ø         CP      16H             ;  use it for allocation
1712 2842     Ø843Ø         JR      Z,CG14
1714 1D       Ø844Ø         DEC     E               ;Backup to previous
1715 1D       Ø845Ø         DEC     E               ;  extent
1716 1A       Ø846Ø CG12    LD      A,(DE)          ;P/u # of contig grans to
1717 E61F     Ø847Ø         AND     1FH             ;  see if the last gran
1719 3C       Ø848Ø         INC     A               ;  used can be extended
171A 4F       Ø849Ø         LD      C,A             ;Is current # the max
171B FE2Ø     Ø85ØØ         CP      2ØH             ;  an extent can hold?
171D 282Ø     Ø851Ø         JR      Z,CG13          ;Jump if a full extent
171F 1A       Ø852Ø         LD      A,(DE)          ;  (32 grans max) - else
172Ø E6EØ     Ø853Ø         AND     ØEØH            ;  p/u the relative
1722 Ø7       Ø854Ø         RLCA                    ;  granule offset
1723 Ø7       Ø855Ø         RLCA
1724 Ø7       Ø856Ø         RLCA
1725 81       Ø857Ø         ADD     A,C             ;Add the # of contiguous
1726 D5       Ø858Ø         PUSH    DE              ;  granules
1727 CD1919   Ø859Ø         CALL    RELCYL          ;Calc relative cyl needed
172A 47       Ø86ØØ         LD      B,A             ;Save offset
172B 4B       Ø861Ø         LD      C,E
172C D1       Ø862Ø         POP     DE
172D 1B       Ø863Ø         DEC     DE              ;Backup to starting cyl
172E 1A       Ø864Ø         LD      A,(DE)
172F 13       Ø865Ø         INC     DE              ;  & repoint to alloc byte
173Ø 8Ø       Ø866Ø         ADD     A,B             ;Add cyls used to
1731 6F       Ø867Ø         LD      L,A             ;  starting cyl
1732 2623     Ø868Ø         LD      H,DIRBUF$<-8    ;Is it less than max?
1734 FECB     Ø869Ø         CP      ØCBH
1736 3ØØ7     Ø87ØØ         JR      NC,CG13         ;Jump if too big
1738 79       Ø871Ø         LD      A,C
1739 46       Ø872Ø         LD      B,(HL)          ;P/u the cyl's GAT
173A CD5B18   Ø873Ø         CALL    TSTBIT          ;Test if gran is free
173D 284B     Ø874Ø         JR      Z,CG21          ;Bypass if free gran
              Ø875Ø ;
              Ø876Ø ;       The next gran cannot be used - get another extent
              Ø877Ø ;
173F 1C       Ø878Ø CG13    INC     E               ;Else point to next
174Ø 1C       Ø879Ø         INC     E               ;  extent field
1741 7B       Ø88ØØ         LD      A,E
1742 E61E     Ø881Ø         AND     1EH             ;Jump if not on the FXDE
1744 FE1E     Ø882Ø         CP      1EH             ;  field, else we have to
1746 2ØØE     Ø883Ø         JR      NZ,CG14         ;  obtain an FXDE record
              Ø884Ø ;
              Ø885Ø ;       Last extent used up, get new dir rec for FXDE
              Ø886Ø ;
1748 CDA417   Ø887Ø         CALL    CG23            ;Write curent GAT & HIT
174B C1       Ø888Ø         POP     BC
174C CØ       Ø889Ø         RET     NZ              ;Ret if GAT/HIT error
174D C5       Ø89ØØ         PUSH    BC
174E CDAF17   Ø891Ø         CALL    NEWHIT          ;Get new HIT for FXDE
1751 C1       Ø892Ø         POP     BC
1752 CØ       Ø893Ø         RET     NZ              ;Loop to process
1753 C3AE16   Ø894Ø         JP      CYL_GRN         ;  new extent
              Ø895Ø ;
              Ø896Ø ;       Extent is vacant - use it & get new allocation
              Ø897Ø ;
1756 CDFE18   Ø898Ø CG14    CALL    MAXCYL          ;Get highest # cyl
1759 326Ø17   Ø899Ø         LD      (CG17+1),A      ;Stuff highest cyl
175C Ø6Ø2     Ø9ØØØ         LD      B,2
```

File positioning subroutines

```
175E 7D       Ø9Ø1Ø CG16    LD      A,L             ;Test last cyl used
175F FEØØ     Ø9Ø2Ø CG17    CP      Ø               ;P/u max cyl
1761 3ØØ7     Ø9Ø3Ø         JR      NC,CG18
1763 7E       Ø9Ø4Ø         LD      A,(HL)          ;P/u a GAT byte
1764 3C       Ø9Ø5Ø         INC     A
1765 2Ø1Ø     Ø9Ø6Ø         JR      NZ,CG19         ;Go if space in this cyl
1767 2C       Ø9Ø7Ø         INC     L               ;  else bump to next one
1768 18F4     Ø9Ø8Ø         JR      CG16            ;  & loop
176A 2EØØ     Ø9Ø9Ø CG18    LD      L,Ø             ;Now start from begin
176C 1ØFØ     Ø91ØØ         DJNZ    CG16            ;  of disk & recheck
176E C1       Ø911Ø         POP     BC
176F CDA417   Ø912Ø         CALL    CG23            ;Write out GAT & HIT
1772 CØ       Ø913Ø         RET     NZ
1773 3E1B     Ø914Ø         LD      A,1BH           ;"disk space full"
1775 B7       Ø915Ø         OR      A
1776 C9       Ø916Ø         RET
              Ø917Ø ;
              Ø918Ø ;       Found available space in cylinder
              Ø919Ø ;
1777 3EFF     Ø92ØØ CG19    LD      A,ØFFH          ;Set DIR extent to FF
1779 12       Ø921Ø         LD      (DE),A
177A ØEØØ     Ø922Ø         LD      C,Ø
177C 46       Ø923Ø         LD      B,(HL)          ;P/u current GAT alloc
177D 79       Ø924Ø CG2Ø    LD      A,C
177E CD5B18   Ø925Ø         CALL    TSTBIT          ;Find a free gran
1781 28Ø7     Ø926Ø         JR      Z,CG21          ;  & jump when found
1783 1A       Ø927Ø         LD      A,(DE)          ;  else advance starting
1784 C62Ø     Ø928Ø         ADD     A,2ØH           ;  rel gran value
1786 12       Ø929Ø         LD      (DE),A
1787 ØC       Ø93ØØ         INC     C               ;Bump pointer to test
1788 18F3     Ø931Ø         JR      CG2Ø            ;  next gran
              Ø932Ø ;
              Ø933Ø ;       Next gran in line is free - allocate it
              Ø934Ø ;
178A 79       Ø935Ø CG21    LD      A,C
178B CD6818   Ø936Ø         CALL    SETBIT          ;Show it allocated
178E B6       Ø937Ø         OR      (HL)
178F 77       Ø938Ø         LD      (HL),A
179Ø 1D       Ø939Ø         DEC     E               ;Backup to starting cyl
1791 1A       Ø94ØØ         LD      A,(DE)          ;Bump by one to see if
1792 3C       Ø941Ø         INC     A               ;  this alloc is the 1st
1793 2ØØ2     Ø942Ø         JR      NZ,CG22         ;  one for the extent &
1795 7D       Ø943Ø         LD      A,L             ;  we have to set the
              Ø944Ø                                 ;  starting cylinder
1796 12       Ø945Ø         LD      (DE),A          ;Stuff starting cyl
1797 1C       Ø946Ø CG22    INC     E
1798 1A       Ø947Ø         LD      A,(DE)          ;Add 1 to # of contiguous
1799 3C       Ø948Ø         INC     A               ;  granules
179A 12       Ø949Ø         LD      (DE),A
179B C1       Ø95ØØ         POP     BC              ;Decrement needed gran
179C ØB       Ø951Ø         DEC     BC              ;  count since we just
179D C5       Ø952Ø         PUSH    BC              ;  allocated one
179E 78       Ø953Ø         LD      A,B             ;Loop if we need more
179F B1       Ø954Ø         OR      C               ;  space allocated
17AØ C21617   Ø955Ø         JP      NZ,CG12
17A3 C1       Ø956Ø         POP     BC
17A4 DD4EØ6   Ø957Ø CG23    LD      C,(IX+6)        ;Else p/u the drive #
17A7 CD7518   Ø958Ø         CALL    @GATWR          ;  & write out the GAT
17AA CØ       Ø959Ø         RET     NZ
```

File positioning subroutines

```
17AB Ø6ØØ     Ø96ØØ STUFDEC LD      B,Ø             ;P/u DEC of FPDE
17AD 1854      Ø961Ø         JR      @DIRWR
               Ø962Ø ;
               Ø963Ø ;       Get new HIT for FXDE
               Ø964Ø ;
17AF DD4EØ6    Ø965Ø NEWHIT  LD      C,(IX+6)        ;P/u drive #
17B2 CD9718    Ø966Ø         CALL    @HITRD          ;Read the HIT
17B5 CØ        Ø967Ø         RET     NZ
17B6 DD7EØ7    Ø968Ø         LD      A,(IX+7)        ;P/u FPDE DEC so 1st ck
17B9 E61F      Ø969Ø         AND     1FH             ;  will be for next
17BB CD1F18    Ø97ØØ         CALL    NHIT4           ;  in line
17BE 3E1E      Ø971Ø         LD      A,1EH           ;Init "full directory...
17CØ CØ        Ø972Ø         RET     NZ              ;Ret if no space
17C1 45        Ø973Ø         LD      B,L             ;Set DEC for
17C2 7D        Ø974Ø         LD      A,L             ;  directory read
17C3 32Ø218    Ø975Ø         LD      (NHIT3+1),A     ;Stuff new DEC from HIT
17C6 54        Ø976Ø         LD      D,H
17C7 DD5EØ7    Ø977Ø         LD      E,(IX+7)        ;P/u current DEC
17CA 1A        Ø978Ø         LD      A,(DE)          ;Copy filespec hash code
17CB 77        Ø979Ø         LD      (HL),A          ;  to new DEC
17CC CD9818    Ø98ØØ         CALL    @HITWR
17CF CCBB18    Ø981Ø         CALL    Z,@DIRRD
17D2 CØ        Ø982Ø         RET     NZ
17D3 369Ø      Ø983Ø         LD      (HL),9ØH        ;Show dir rec in use as
17D5 2C        Ø984Ø         INC     L               ;  FXDE record
17D6 C5        Ø985Ø         PUSH    BC              ;P/u DEC of FPDE &
17D7 3AAC17    Ø986Ø         LD      A,(STUFDEC+1)   ;  stuff it into FXDE's
17DA 77        Ø987Ø         LD      (HL),A          ;  DIR+1 to link back
17DB 2C        Ø988Ø         INC     L
17DC Ø614      Ø989Ø         LD      B,2Ø            ;Zero out 2Ø bytes
17DE 36ØØ      Ø99ØØ NHIT1   LD      (HL),Ø          ;  in the FXDE
17EØ 2C        Ø991Ø         INC     L
17E1 1ØFB      Ø992Ø         DJNZ    NHIT1
17E3 E5        Ø993Ø         PUSH    HL              ;Save ptr to 1st extent
17E4 Ø6ØA      Ø994Ø         LD      B,1Ø            ;Init to X'FF' 1Ø bytes
17E6 36FF      Ø995Ø NHIT2   LD      (HL),ØFFH       ;  or 5 extents
17E8 2C        Ø996Ø         INC     L
17E9 1ØFB      Ø997Ø         DJNZ    NHIT2
17EB D1        Ø998Ø         POP     DE              ;Rcvr ptr to 1st extent
17EC 13        Ø999Ø         INC     DE              ;Pt to allocation byte
17ED C1        1ØØØØ         POP     BC
17EE CDØ318    1ØØ1Ø         CALL    @DIRWR          ;Write FXDE back to disk
17F1 CØ        1ØØ2Ø         RET     NZ              ;Return if error
17F2 3AAC17    1ØØ3Ø         LD      A,(STUFDEC+1)   ;  else p/u DEC of FPDE
17F5 47        1ØØ4Ø         LD      B,A
17F6 CDBB18    1ØØ5Ø         CALL    @DIRRD          ;Read its directory
17F9 CØ        1ØØ6Ø         RET     NZ              ;  & return if error
17FA 7D        1ØØ7Ø         LD      A,L
17FB C61E      1ØØ8Ø         ADD     A,1EH           ;Point to FXDE posn
17FD 6F        1ØØ9Ø         LD      L,A             ;  in FPDE
17FE 36FE      1Ø1ØØ         LD      (HL),ØFEH       ;Show link to FXDE
18ØØ 2C        1Ø11Ø         INC     L
18Ø1 36ØØ      1Ø12Ø NHIT3   LD      (HL),Ø          ;Show what's the FXDE DEC
               1Ø13Ø                                 ;  & write the DIR back
               1Ø14Ø ;
               1Ø15Ø ;       Routine to write a directory sector
               1Ø16Ø ;       B => DEC of FPDE, C => logical drive number
               1Ø17Ø ;       HL <= points to directory record in SBUFF$
               1Ø18Ø ;
```

File positioning subroutines

```
18Ø3 CDØ718   1Ø19Ø @DIRWR  CALL    DIRWR           ;Permit two attempts
18Ø6 C8       1Ø2ØØ        RET     Z
18Ø7 D5       1Ø21Ø DIRWR  PUSH    DE              ;Save the reg
18Ø8 CDCA18   1Ø22Ø        CALL    CALCDIR         ;Calc dir cyl
18ØB 2EØØ     1Ø23Ø        LD      L,Ø             ;Set buffer to start
18ØD CDEC19   1Ø24Ø        CALL    @WRSSC          ;Write the sector
1811Ø CCDC19  1Ø25Ø        CALL    Z,@VRSEC        ;Verify on no error
1813 D6Ø6     1Ø26Ø        SUB     6
1815 D1       1Ø27Ø        POP     DE
1816 C8       1Ø28Ø        RET     Z               ;Back on system sector
1817 FEØ9     1Ø29Ø        CP      ØFH-6           ;WP error?
1819 3E12     1Ø3ØØ        LD      A,18            ;Set dir write error
181B CØ       1Ø31Ø        RET     NZ              ;  if not WP
181C D6Ø3     1Ø32Ø        SUB     3
181E C9       1Ø33Ø        RET
              1Ø34Ø ;
              1Ø35Ø ;      Find a spare HIT entry
              1Ø36Ø ;
181F F5       1Ø37Ø NHIT4  PUSH    AF
182Ø 3EØ7     1Ø38Ø        LD      A,7             ;Get highest # sector
1822 CD2B1A   1Ø39Ø        CALL    @DCTBYT         ;  on a cylinder
1825 D5       1Ø4ØØ        PUSH    DE              ;  into register E
1826 57       1Ø41Ø        LD      D,A
1827 E61F     1Ø42Ø        AND     1FH
1829 5F       1Ø43Ø        LD      E,A
182A 1C       1Ø44Ø        INC     E               ;& get number of heads
182B AA       1Ø45Ø        XOR     D               ;  into register A
182C Ø7       1Ø46Ø        RLCA
182D Ø7       1Ø47Ø        RLCA
182E Ø7       1Ø48Ø        RLCA
182F 3C       1Ø49Ø        INC     A
183Ø CDØA19   1Ø5ØØ        CALL    @MUL8           ;To calc sectors/cylinder
1833 CD3B19   1Ø51Ø        CALL    CKDBLBIT        ;Double if necessary
1836 D1       1Ø52Ø        POP     DE              ;Total sectors per cyl
1837 D6Ø2     1Ø53Ø        SUB     2               ;Reduce for GAT & HIT
1839 324918   1Ø54Ø        LD      (NHIT7+1),A     ;# of directory sectors
183C F1       1Ø55Ø        POP     AF              ;Get DEC init entry
183D 6F       1Ø56Ø        LD      L,A
183E CD4518   1Ø57Ø        CALL    NHIT6           ;Ck if HIT slot is spare
1841 C8       1Ø58Ø        RET     Z               ;Return if it is spare
1842 2E3F     1Ø59Ø        LD      L,3FH
1844 2C       1Ø6ØØ NHIT5  INC     L
1845 7D       1Ø61Ø NHIT6  LD      A,L
1846 E61F     1Ø62Ø        AND     1FH
1848 FEØØ     1Ø63Ø NHIT7  CP      Ø               ;Does value exceed
184A 3ØØD     1Ø64Ø        JR      NC,NHIT9        ;  sectors/cylinder?
184C 7E       1Ø65Ø        LD      A,(HL)
184D B7       1Ø66Ø        OR      A
184E C8       1Ø67Ø        RET     Z
184F 7D       1Ø68Ø NHIT8  LD      A,L
185Ø C62Ø     1Ø69Ø        ADD     A,2ØH
1852 6F       1Ø7ØØ        LD      L,A
1853 3ØFØ     1Ø71Ø        JR      NC,NHIT6
1855 FE1F     1Ø72Ø        CP      1FH             ;Else go to next sector
1857 2ØEB     1Ø73Ø        JR      NZ,NHIT5        ;  column
1859 B7       1Ø74Ø NHIT9  OR      A
185A C9       1Ø75Ø        RET
              1Ø76Ø ;
              1Ø77Ø ;      Test if gran is free in GAT
```

File positioning subroutines

```
                10780 ;
185B E607       10790 TSTBIT  AND     7               ;Get Ø to 7
185D Ø7         10800         RLCA                    ;Shift to match BIT n,
185E Ø7         10810         RLCA                    ;  opcode
185F Ø7         10820         RLCA
1860 F640       10830         OR      40H
1862 326618     10840         LD      (TBIT1+1),A     ;Modify BIT instruction
1865 CB40       10850 TBIT1   BIT     Ø,B
1867 C9         10860         RET
                10870 ;
                10880 ;       Set gran to allocated in GAT
                10890 ;
1868 Ø7         10900 SETBIT  RLCA                    ;Shift to create opcode
1869 Ø7         10910         RLCA                    ;  to match current bit
186A Ø7         10920         RLCA
186B F6C7       10930         OR      ØC7H
186D 327218     10940         LD      (SBIT1+1),A     ;Create SET n, opcode
1870 AF         10950         XOR     A
1871 CBC7       10960 SBIT1   SET     Ø,A
1873 C9         10970         RET
                10980 ;
                10990 ;       Routine reads/writes the Granule Allocation Table
                11000 ;
1874 F6         11010 @GATRD  DB      ØF6H            ;Set NZ for test
1875 AF         11020 @GATWR  XOR     A               ;Set Z for test
1876 D5         11030         PUSH    DE
1877 E5         11040         PUSH    HL
1878 F5         11050         PUSH    AF              ;Save flag for test
1879 CDF718     11060         CALL    @DIRCYL
187C 210023     11070         LD      HL,DIRBUF$
187F 5D         11080         LD      E,L             ;Set E to Ø
1880 F1         11090         POP     AF              ;Rcvr flag for R/W
1881 2807       11100         JR      Z,GATRW1        ;Go if @GATWR
1883 CDD818     11110         CALL    @RDSSC
1886 3E14       11120         LD      A,14H           ;Init "GAT read error"
1888 180A       11130         JR      GATRW2
188A CDEC19     11140 GATRW1  CALL    @WRSSC          ;Protected sector write
188D CCDC19     11150         CALL    Z,@VRSEC        ;Verify if OK
1890 FE06       11160         CP      6               ;Protected sector?
1892 3E15       11170         LD      A,15H           ;Init "GAT write error"
1894 E1         11180 GATRW2  POP     HL
1895 D1         11190         POP     DE
1896 C9         11200         RET
                11210 ;
                11220 ;       Read or write the hash index table
                11230 ;
1897 F6         11240 @HITRD  DB      ØF6H            ;Set NZ for test
1898 AF         11250 @HITWR  XOR     A               ;Set Z for test
1899 C5         11260         PUSH    BC
189A D5         11270         PUSH    DE
189B F5         11280         PUSH    AF              ;Save flag for test
189C CDF718     11290         CALL    @DIRCYL         ;D => directory cylinder
189F 1E01       11300         LD      E,1             ;E => HIT sector
18A1 21001D     11310         LD      HL,SBUFF$       ;HL => HIT buffer area
18A4 F1         11320         POP     AF              ;Rcvr flag for RD/WR
18A5 2807       11330         JR      Z,HITRW1        ;Go if @HITWR
18A7 CDD818     11340         CALL    @RDSSC          ;Read cyl D, sector E
18AA 3E16       11350         LD      A,22            ;Init "HIT read error"
18AC 180A       11360         JR      HITRW2
```

File positioning subroutines

```
18AE CDEC19   11370 HITRW1  CALL    @WRSSC          ;Protected sector write
18B1 CCDC19   11380         CALL    Z,@VRSEC        ;Verify the write
18B4 FEØ6     11390         CP      6               ;Protected sector?
18B6 3E17     11400         LD      A,23            ;"HIT write error"
18B8 D1       11410 HITRW2  POP     DE              ;Message for other than
18B9 C1       11420         POP     BC              ;  attempt protected sector
18BA C9       11430         RET
              11440 ;
              11450 ;       Routine to read a directory sector
              11460 ;       B => DEC of FPDE, C => logical drive number
              11470 ;       HL <= points to directory record in SBUFF$
              11480 ;
18BB D5       11490 @DIRRD  PUSH    DE
18BC CDCA18   11500         CALL    CALCDIR         ;Set HL to SBUFF$
18BF E5       11510         PUSH    HL
18CØ 2EØØ     11520         LD      L,Ø             ;Start of bfr
18C2 CDD818   11530         CALL    @RDSSC          ;Read it
18C5 E1       11540         POP     HL
18C6 3E11     11550         LD      A,17            ;Init to dir read err
18C8 D1       11560         POP     DE
18C9 C9       11570         RET
              11580 ;
              11590 ;       Routine to get directory access data
              11600 ;       B => DEC
              11610 ;       DE <= cylinder and sector needed
              11620 ;       HL <= pointer to directory record in SBUFF$
              11630 ;
18CA CDF718   11640 CALCDIR CALL    @DIRCYL         ;Get directory cyl in D
18CD 78       11650         LD      A,B             ;Calculate record start
18CE E6EØ     11660         AND     ØEØH            ;  from the DEC
18DØ 6F       11670         LD      L,A
18D1 261D     11680         LD      H,SBUFF$<-8     ;Point to buffer start
18D3 A8       11690         XOR     B               ;Calculate directory
18D4 C6Ø2     11700         ADD     A,2             ;  sector needed
18D6 5F       11710         LD      E,A
18D7 C9       11720         RET
              11730 ;
              11740 ;       Read system sector, D=Track, E=Sector, HL=Buffer
              11750 ;
18D8 CDF118   11760 @RDSSC  CALL    READIR
18DB C8       11770         RET     Z
18DC D5       11780         PUSH    DE
18DD 11Ø1ØØ   11790         LD      DE,1            ;Pt to tk Ø, sec 1
18EØ CDF419   11800         CALL    @RDSEC          ;Read to find dir cyl
18E3 D1       11810         POP     DE
18E4 CØ       11820         RET     NZ
18E5 E5       11830         PUSH    HL
18E6 23       11840         INC     HL              ;Pt to dir tk #
18E7 23       11850         INC     HL
18E8 56       11860         LD      D,(HL)          ;P/u dir tk fm boot
18E9 26Ø9     11870         LD      H,9             ;Update memory table
18EB CD341A   11880         CALL    DCTFLD@
18EE 6F       11890         LD      L,A
18EF 72       11900         LD      (HL),D
18FØ E1       11910         POP     HL
18F1 CDF419   11920 READIR  CALL    @RDSEC          ;Retry dir read
18F4 D6Ø6     11930         SUB     6               ;Test protected
18F6 C9       11940         RET
              11950 ;
```

File positioning subroutines

```
18F7 3EØ9      11960 @DIRCYL LD      A,9
18F9 CD2B1A    11970         CALL    @DCTBYT         ;Get the dir cylinder
18FC 57        11980         LD      D,A
18FD C9        11990         RET
               12000 ;
18FE 3EØ6      12010 MAXCYL  LD      A,6
19ØØ C5        12020         PUSH    BC
19Ø1 DD4EØ6    12030         LD      C,(IX+6)
19Ø4 CD2B1A    12040         CALL    @DCTBYT         ;Get highest # cyl
19Ø7 3C        12050         INC     A               ;Adjust for zero offset
19Ø8 C1        12060         POP     BC
19Ø9 C9        12070         RET
               12080 ;
               12090 ;       Multiply register E by register A
               12100 ;
190A C5        12110 @MUL8   PUSH    BC              ;Mult A x E
190B 57        12120         LD      D,A
190C AF        12130         XOR     A
190D Ø6Ø8      12140         LD      B,8
190F 87        12150 MEA1    ADD     A,A
1910 CB23      12160         SLA     E
1912 3ØØ1      12170         JR      NC,MEA2
1914 82        12180         ADD     A,D
1915 1ØF8      12190 MEA2    DJNZ    MEA1
1917 C1        12200         POP     BC
1918 C9        12210         RET
               12220 ;
               12230 ;       Calculate relative cylinder for granule needed
               12240 ;
1919 5F        12250 RELCYL  LD      E,A
191A CD261A    12260         CALL    @DCTBYT-5       ;Get # of grans/track
191D 47        12270         LD      B,A             ;Hang on to this
191E Ø7        12280         RLCA
191F Ø7        12290         RLCA
1920 Ø7        12300         RLCA
1921 E6Ø7      12310         AND     7
1923 3C        12320         INC     A               ;Adj for Ø offset
1924 CD3B19    12330         CALL    CKDBLBIT
               12340 ;
               12350 ;       Divide register E by register A
               12360 ;
1927 C5        12370 @DIV8   PUSH    BC
1928 4F        12380         LD      C,A
1929 Ø6Ø8      12390         LD      B,8
192B AF        12400         XOR     A
192C CB23      12410 DEA1    SLA     E
192E 17        12420         RLA
192F B9        12430         CP      C
1930 38Ø2      12440         JR      C,DEA2
1932 91        12450         SUB     C
1933 1C        12460         INC     E
1934 1ØF6      12470 DEA2    DJNZ    DEA1
1936 4F        12480         LD      C,A
1937 7B        12490         LD      A,E
1938 59        12500         LD      E,C
1939 C1        12510         POP     BC
193A C9        12520         RET
               12530 ;
               12540 ;       Routine to double the A register if DBL bit is set
```

File positioning subroutines

```
                12550 ;
                12560 CKDBLBIT
193B 57         12570         LD      D,A             ;Adjust for 2-sided &
193C 3E04       12580         LD      A,4             ;  calculate # of cyls
193E CD2B1A     12590         CALL    @DCTBYT
1941 CB6F       12600         BIT     5,A             ;Test if 2-sided
1943 7A         12610         LD      A,D
1944 2801       12620         JR      Z,$+3           ;Double the grans if 2
1946 87         12630         ADD     A,A             ;  & fall thru to DIV8
1947 C9         12640         RET
```

File positioning subroutines

```
1948           03360          PAGE
1948           03370 CORE$    DEFL    $
F80D           03380          ORG     CRTBGN$+13
F80D 4C        03390          DB      'LS-DOS 06.02.00'
     53 2D 44 4F 53 20 30 36
     2E 30 32 2E 30 30
               03400          IF      @USA
F81C 20        03410          DB      ' '
               03420          ENDIF
               03430          IF      @GERMAN
               03440          DB      'D'
               03450          ENDIF
               03460          IF      @FRENCH
               03470          DB      'F'
               03480          ENDIF
F81D 2D        03490          DB      '- Copyright 1984 '
     20 43 6F 70 79 72 69 67
     68 74 20 31 39 38 34 20
F82E 4C        03500          DB      'Logical Systems Inc.'
     6F 67 69 63 61 6C 20 53
     79 73 74 65 6D 73 20 49
     6E 63 2E
F85E           03510          ORG     CRTBGN$+80+14
F85E 41        03520          DB      'All Rights Reserved. '
     6C 6C 20 52 69 67 68 74
     73 20 52 65 73 65 72 76
     65 64 2E 20
F873 4C        03530          DB      'Licensed to
     69 63 65 6E 73 65 64 20
     74 6F 20 20 20 20 20 20
     20 20 20 20 20 20 20 20
     20 20 20 20 20
1948           03540          ORG     CORE$
               03550 ;
               03560 ;        get the system loader
               03570 ;
1948           03580          SUBTTL  '<System Loader and associated routines>'
```

System Loader and associated routines

```
1948            Ø36ØØ *GET    LOADER:3
                12650 ;LOADER/ASM - LS-DOS 6.2
1948            12660 CORE$   DEFL    $
Ø1ØØ            12670         ORG     SVCTAB$
                12680 ;
                12690 ;       Supervisor Call table - Page 5
                12700 ;
Ø1ØØ F21B       12710         DW      @IPL,@KEY,@DSP,@GET             ;Ø-3
     28Ø6 42Ø6 38Ø6
Ø1Ø8 45Ø6       12720         DW      @PUT,@CTL,@PRT,@WHERE           ;4-7
     23Ø6 3DØ6 7919
Ø11Ø 35Ø6       12730         DW      @KBD,@KEYIN,@DSPLY,@LOGER       ;8-11
     85Ø5 2DØ5 Ø3Ø5
Ø118 ØØØ5       12740         DW      @LOGOT,@MSG,@PRINT,@VDCTL       ;12-15
     3ØØ5 28Ø5 99ØB
Ø12Ø 82Ø3       12750         DW      @PAUSE,@PARAM,@DATE,@TIME       ;16-19
     8719 A8Ø7 8DØ7
Ø128 89Ø6       12760         DW      @CHNIO,@ABORT,@EXIT,SVCERR      ;2Ø-23
     Ø81B ØB1B F41A
Ø13Ø 7E19       12770         DW      @CMNDI,@CMNDR,@ERROR,@DEBUG     ;24-27
     7B19 ØF1B AØ19
Ø138 F51C       12780         DW      @CKTSK,@ADTSK,@RMTSK,@RPTSK     ;28-31
     DA1C D71C EB1C
Ø14Ø DØ1C       12790         DW      @KLTSK,@CKDRV,@DODIR,@RAMDIR    ;32-35
     9319 AF19 AC19
Ø148 F41A       12800         DW      SVCERR,SVCERR,SVCERR,SVCERR     ;36-39
     F41A F41A F41A
Ø15Ø B519       12810         DW      @DCSTAT,@SLCT,@DCINIT,@DCRES    ;4Ø-43
     BC19 CØ19 C419
Ø158 C819       12820         DW      @RSTOR,@STEPI,@SEEK,@RSLCT      ;44-47
     CC19 DØ19 D419
Ø16Ø D819       12830         DW      @RDHDR,@RDSEC,@VRSEC,@RDTRK     ;48-51
     F419 DC19 EØ19
Ø168 E419       12840         DW      @HDFMT,@WRSEC,@WRSSC,@WRTRK     ;52-55
     E819 EC19 FØ19
Ø17Ø 9619       12850         DW      @RENAME,@REMOVE,@INIT,@OPEN     ;56-59
     A619 8D19 8A19
Ø178 9919       12860         DW      @CLOSE,@BKSP,@CKEOF,@LOC        ;6Ø-63
     8614 8F15 B314
Ø18Ø DE14       12870         DW      @LOF,@PEOF,@POSN,@READ          ;64-67
     A214 3414 1315
Ø188 9B14       12880         DW      @REW,@RREAD,@RWRIT,@SEEKSC      ;68-71
     7314 AD13 2114
Ø19Ø 3Ø14       12890         DW      @SKIP,@VER,@WEOF,@WRITE         ;72-75
     6Ø15 EC14 3115
Ø198 381B       12900         DW      @LOAD,@RUN,@FSPEC,@FEXT         ;76-79
     1D1B 8119 8419
Ø1AØ 9C19       12910         DW      @FNAME,@GTDCT,@GTDCB,@GTMOD     ;8Ø-83
     1E1A 9Ø19 B219
Ø1A8 F41A       12920         DW      SVCERR,@RDSSC,@GATRD,@DIRRD     ;84-87
     D818 7418 BB18
Ø1BØ Ø318       12930         DW      @DIRWR,@GATWR,@MUL8,@MUL16      ;88-91
     7518 ØA19 C9Ø6
Ø1B8 F41A       12940         DW      SVCERR,@DIV8,@DIV16,SVCERR      ;92-95
     2719 E3Ø6 F41A
Ø1CØ E1Ø3       12950         DW      @DECHEX,@HEXDEC,@HEX8,@HEX16    ;96-99
     F6Ø6 C2Ø7 BDØ7
Ø1C8 4819       12960         DW      @HIGH$,@FLAGS,@BANK,@BREAK      ;1ØØ-1Ø3
     6A19 77Ø8 6F19
Ø1DØ 92Ø3       12970         DW      @SOUND,@CLS,@CKBRKC,SVCERR      ;1Ø4-1Ø7
```

System Loader and associated routines

```
     45Ø5 53Ø5 F41A
Ø1D8 F41A     1298Ø          DW      SVCERR,SVCERR,SVCERR,SVCERR     ;1Ø8-111
     F41A F41A F41A
Ø1EØ F41A     1299Ø          DW      SVCERR,SVCERR,SVCERR,SVCERR     ;112-115
     F41A F41A F41A
Ø1E8 F41A     13ØØØ          DW      SVCERR,SVCERR,SVCERR,SVCERR     ;116-119
     F41A F41A F41A
Ø1FØ F41A     13Ø1Ø          DW      SVCERR,SVCERR,SVCERR,SVCERR     ;12Ø-123
     F41A F41A F41A
Ø1F8 F41A     13Ø2Ø          DW      SVCERR,SVCERR,SVCERR,SVCERR     ;124-127
     F41A F41A F41A
1948          13Ø3Ø          ORG     CORE$
              13Ø4Ø ;
              13Ø5Ø ;        Routine to set or retrieve HIGH$/LOW$
              13Ø6Ø ;
1948 7C       13Ø7Ø @HIGH$   LD      A,H             ;Test if put or get
1949 B5       13Ø8Ø          OR      L
194A 2812     13Ø9Ø          JR      Z,GETHILO       ;Go if get
194C 3A6CØØ   131ØØ          LD      A,(CFLAG$)      ;Is HIGH$ changeable?
194F ØF       1311Ø          RRCA
195Ø 3E2B     1312Ø          LD      A,43            ;Init SVC parm error
1952 D8       1313Ø          RET     C               ;Back with NZ
1953 Ø4       1314Ø          INC     B               ;Test for HIGH$/LOW$
1954 Ø5       1315Ø          DEC     B
1955 2ØØE     1316Ø          JR      NZ,PUTLO        ;Go if LOW$
1957 22ØEØ4   1317Ø          LD      (HIGH$),HL      ;Set new HIGH$
195A 2AØEØ4   1318Ø GETHI    LD      HL,(HIGH$)      ;P/u the value &
195D C9       1319Ø          RET                     ;  ret with Z-flag
195E Ø4       132ØØ GETHILO  INC     B               ;Test for HIGH$/LOW$
195F Ø5       1321Ø          DEC     B
196Ø 28F8     1322Ø          JR      Z,GETHI
1962 2A1EØØ   1323Ø          LD      HL,(LOW$)       ;P/u LOW$
1965 221EØØ   1324Ø PUTLO    LD      (LOW$),HL       ;Get LOW$
1968 AF       1325Ø          XOR     A               ;Set Z-flag
1969 C9       1326Ø          RET
              1327Ø ;
196A FD216AØØ 1328Ø @FLAGS   LD      IY,FLGTAB$
196E C9       1329Ø          RET
              133ØØ ;
196F E5       1331Ø @BREAK   PUSH    HL              ;Save user vector
197Ø 2A881C   1332Ø          LD      HL,(BRKVEC$)    ;P/u current vector
1973 E3       1333Ø          EX      (SP),HL         ;Save current & get user
1974 22881C   1334Ø          LD      (BRKVEC$),HL    ;Stuff new vector
1977 E1       1335Ø          POP     HL              ;Recover old vector
1978 C9       1336Ø          RET
              1337Ø ;
1979 E1       1338Ø @WHERE   POP     HL
197A E9       1339Ø          JP      (HL)
              134ØØ ;
              1341Ø ;        Code for these SVCs is in system overlays
              1342Ø ;
197B 3EA3     1343Ø @CMNDR   LD      A,ØA3H          ;Interpret command & RET
197D EF       1344Ø          RST     4Ø
197E 3EB3     1345Ø @CMNDI   LD      A,ØB3H          ;Interpret a command
198Ø EF       1346Ø          RST     4Ø
1981 3EC3     1347Ø @FSPEC   LD      A,ØC3H          ;Parse a filespec
1983 EF       1348Ø          RST     4Ø
1984 3ED3     1349Ø @FEXT    LD      A,ØD3H          ;Optional default EXT
1986 EF       135ØØ          RST     4Ø
```

System Loader and associated routines

```
1987 3EE3     13510 @PARAM  LD    A,0E3H        ;Parameter scanner
1989 EF       13520         RST   40
198A 3E94     13530 @OPEN   LD    A,94H         ;Open a file
198C EF       13540         RST   40
198D 3EA4     13550 @INIT   LD    A,0A4H        ;Initialize a file
198F EF       13560         RST   40
1990 3EB4     13570 @GTDCB  LD    A,0B4H        ;Get a DCB vector
1992 EF       13580         RST   40
1993 3EC4     13590 @CKDRV  LD    A,0C4H        ;Drive available?
1995 EF       13600         RST   40
1996 3EF4     13610 @RENAME LD    A,0F4H        ;Rename a file
1998 EF       13620         RST   40
1999 3E95     13630 @CLOSE  LD    A,95H         ;Close a file
199B EF       13640         RST   40
199C 3EA5     13650 @FNAME  LD    A,0A5H        ;Recover filespec
199E EF       13660         RST   40
199F C9       13670 @DBGHK  RET                 ;Init DEBUG off (NOP=on)
19A0 F5       13680 @DEBUG  PUSH  AF
19A1 3E97     13690         LD    A,97H         ;Enter system Debugger
19A3 EF       13700         RST   40
19A4 DC14     13710 EXTDBG$ DW    ORARET@       ;Hook for extended DEBUG
19A6 3E9C     13720 @REMOVE LD    A,9CH         ;Remove a file/device
19A8 EF       13730         RST   40
19A9 3ECD     13740 @DOKEY  LD    A,0CDH        ;DO execution
19AB EF       13750         RST   40
19AC 3E9E     13760 @RAMDIR LD    A,09EH        ;Directory data
19AE EF       13770         RST   40
19AF 3EAE     13780 @DODIR  LD    A,0AEH        ;Directory data
19B1 EF       13790         RST   40
19B2 3EBE     13800 @GTMOD  LD    A,0BEH        ;Get module address
19B4 EF       13810         RST   40
              13820 ;
              13830 ;       These SVCs handle the disk primitive requests
              13840 ;
19B5 AF       13850 @DCSTAT XOR   A               ;FDC status
19B6 183E     13860         JR    IOFUNC
19B8 3A2300   13870 TAPDRV  LD    A,(LDRV$)     ;P/u drive #
19BB 4F       13880         LD    C,A
19BC 3E01     13890 @SLCT   LD    A,1           ;Select drive
19BE 1836     13900         JR    IOFUNC
19C0 3E02     13910 @DCINIT LD    A,2           ;FDC init
19C2 1832     13920         JR    IOFUNC
19C4 3E03     13930 @DCRES  LD    A,3           ;FDC reset
19C6 182E     13940         JR    IOFUNC
19C8 3E04     13950 @RSTOR  LD    A,4           ;Restore to cyl 0
19CA 182A     13960         JR    IOFUNC
19CC 3E05     13970 @STEPI  LD    A,5           ;Step in 1 cyl
19CE 1826     13980         JR    IOFUNC
19D0 3E06     13990 @SEEK   LD    A,6           ;Seek a track/sector
19D2 1822     14000         JR    IOFUNC
19D4 3E07     14010 @RSLCT  LD    A,7           ;Re-select drive
19D6 181E     14020         JR    IOFUNC
19D8 3E08     14030 @RDHDR  LD    A,8
19DA 181A     14040         JR    IOFUNC
19DC 3E0A     14050 @VRSEC  LD    A,10          ;Verify a sector
19DE 1816     14060         JR    IOFUNC
19E0 3E0B     14070 @RDTRK  LD    A,11
19E2 1812     14080         JR    IOFUNC
19E4 3E0C     14090 @HDFMT  LD    A,12
```

System Loader and associated routines

```
19E6 18ØE     14100          JR      IOFUNC
19E8 3EØD     14110 @WRSEC   LD      A,13            ;Write standard sector
19EA 18ØA     14120          JR      IOFUNC
19EC 3EØE     14130 @WRSSC   LD      A,14            ;Write a system sector
19EE 18Ø6     14140          JR      IOFUNC
19FØ 3EØF     14150 @WRTRK   LD      A,15            ;Write a track
19F2 18Ø2     14160          JR      IOFUNC
19F4 3EØ9     14170 @RDSEC   LD      A,9             ;Read a sector
              14180 ;
19F6 C5       14190 IOFUNC   PUSH    BC              ;Save reg pair
19F7 47       14200          LD      B,A             ;Xfer the function code
              14210 ;
              14220 ;        Bring up bank Ø
              14230 ;
19F8 C5       14240          PUSH    BC
19F9 AF       14250          XOR     A
19FA 47       14260          LD      B,A             ;Set bank function Ø,
19FB 4F       14270          LD      C,A             ;  bank number Ø
19FC CD77Ø8   14280          CALL    @BANK           ;Bring up bank
19FF F1       14290          POP     AF              ;Perform EX (SP),BC
1AØØ C5       14300          PUSH    BC
1AØ1 F5       14310          PUSH    AF
1AØ2 C1       14320          POP     BC
              14330 ;
              14340 ;        Continue disk I/O setup
              14350 ;
1AØ3 79       14360          LD      A,C             ;Xfer the drive code
1AØ4 3223ØØ   14370          LD      (LDRV$),A
1AØ7 FDE5     14380          PUSH    IY
1AØ9 CD1E1A   14390          CALL    @GTDCT          ;Get DCT address in IY
1AØC 3E2Ø     14400          LD      A,2ØH           ;Set illegal drive #
1AØE B7       14410          OR      A               ;  if drive disabled
1AØF CD1C1A   14420          CALL    GODOIO
1A12 FDE1     14430          POP     IY
              14440 ;
              14450 ;        Bring back the old bank
              14460 ;
1A14 C1       14470          POP     BC
1A15 F5       14480          PUSH    AF              ;Save disk I/O retcod
1A16 3E66     14490          LD      A,1Ø2           ;Set for @BANK
1A18 EF       14500          RST     4Ø              ;No need to ck for error
              14510                                  ;  from @BANK
1A19 F1       14520          POP     AF
1A1A C1       14530          POP     BC
1A1B C9       14540          RET
              14550 ;
1A1C FDE9     14560 GODOIO   JP      (IY)
              14570 ;
1A1E E5       14580 @GTDCT   PUSH    HL              ;Get i/o routine addr
1A1F CD341A   14590          CALL    DCTFLD@         ;  into IY
1A22 E3       14600          EX      (SP),HL
1A23 FDE1     14610          POP     IY
1A25 C9       14620          RET
              14630 ;
              14640 ;        Entry to get DCT+8 of FCB (IX) drive spec
              14650 ;
1A26 DD4EØ6   14660 D@FBYT8  LD      C,(IX+6)        ;P/u drive
              14670 ;
              14680 ;        Entry to get DCT+8 of Reg C drive spec
```

System Loader and associated routines

```
                14690 ;
                14700 DCTBYT8@
1A29 3E08       14710           LD      A,8
                14720 ;
                14730 ;         Entry to get byte (Reg A) from DCT of Reg C drive
                14740 ;          C => logical drive specification
                14750 ;          A => relative byte requested from DCT
                14760 ;          A <= data at position requested
                14770 ;
1A2B E5         14780 @DCTBYT   PUSH    HL              ;Save the register pair
1A2C 67         14790           LD      H,A             ;Xfer relative position
1A2D CD341A     14800           CALL    DCTFLD@         ;Get HL pointing to
1A30 6F         14810           LD      L,A             ;  DCT position
1A31 7E         14820           LD      A,(HL)          ;Get the byte
1A32 E1         14830           POP     HL
1A33 C9         14840           RET
                14850 ;
                14860 ;         Entry to get HL pointing to DCT byte Reg C, Reg A
                14870 ;          C => logical drive number
                14880 ;          A => relative byte in DCT requested
                14890 ;         HL <= start of requested DCT for the drive
                14900 ;          A <= low order pointer to relative byte request
                14910 ;
1A34 79         14920 DCTFLD@   LD      A,C             ;Get drive spec &
1A35 E607       14930           AND     7               ;  strip excess data
1A37 87         14940           ADD     A,A             ;Times 2
1A38 6F         14950           LD      L,A             ;  & saved
1A39 87         14960           ADD     A,A             ;Times 4
1A3A 87         14970           ADD     A,A             ;Times 8
1A3B 85         14980           ADD     A,L             ;Times 10
1A3C C670       14990           ADD     A,70H           ;Add DCT offset from Ø
1A3E 6F         15000           LD      L,A             ;Point L to DCT low order
1A3F 84         15010           ADD     A,H             ;Add in rel pos desired
1A40 2604       15020           LD      H,DCT$<-8       ;Point H to DCT hi-order
1A42 C9         15030           RET
                15040 ;
                15050 ;         Process supervisory calls <Ø-127>
                15060 ;
1A43 FE1A       15070 SVCUSER   CP      26              ;Check for @ERROR
1A45 2808       15080           JR      Z,ERRSVC        ;Skip next if so
1A47 32ØDØØ     15090           LD      (LSVC$),A       ;Store SVC request
1A4A E3         15100           EX      (SP),HL         ;P/u RET address
1A4B 22ØBØØ     15110           LD      (SVCRET$),HL    ;  and save it
1A4E E3         15120           EX      (SP),HL         ;Restore RET address
1A4F E5         15130 ERRSVC    PUSH    HL              ;Save HL
1A50 07         15140           RLCA                    ;Multiply by two
1A51 2601       15150           LD      H,SVCTAB$<-8    ;Base of table
1A53 6F         15160           LD      L,A             ;Set up the low order
1A54 7E         15170           LD      A,(HL)          ;P/u table entry
1A55 2C         15180           INC     L
1A56 66         15190           LD      H,(HL)
1A57 6F         15200           LD      L,A
1A58 E3         15210           EX      (SP),HL         ;P/u HL & stuff vector
1A59 79         15220           LD      A,C             ;Xfer for PUT type ops
1A5A C9         15230           RET
                15240 ;
                15250 ;         RST 28 vector - System & user SVCs
                15260 ;
1A5B B7         15270 RST28     OR      A               ;Test if bit 7 set
```

System Loader and associated routines

```
1A5C F2431A   15280         JP      P,SVCUSER       ;Jump on user SVC attempt
1A5F E3       15290         EX      (SP),HL         ;Discard return addr &
1A6Ø F5       15300         PUSH    AF              ;  save HL, AF
1A61 219F19   15310         LD      HL,@DBGHK       ;Set up DEBUG linkage
1A64 7E       15320         LD      A,(HL)
1A65 32791A   15330         LD      (SET@EXEC),A
1A68 36C9     15340         LD      (HL),ØC9H
1A6A F1       15350         POP     AF              ;Restore AF, HL
1A6B E1       15360         POP     HL
1A6C CD7F1A   15370 HKRES$  CALL    CKMOD@          ;Get overlay if needed
1A6F 3EØØ     15380         LD      A,Ø             ;P/u new overlay #
1A7Ø          15390 OVRLYOLD        EQU     $-1
1A71 326ØØØ   15400         LD      (OVRLY$),A      ;  & update current
1A74 CDØØØØ   15410 TRANSFR CALL    Ø               ;Traadr of SYSx
1A77 F5       15420         PUSH    AF
1A78 3EØØ     15430         LD      A,Ø             ;Set to C9 if EXEC only
1A79          15440 SET@EXEC        EQU     $-1
1A7A 329F19   15450         LD      (@DBGHK),A
1A7D F1       15460         POP     AF
1A7E C9       15470         RET
              15480 ;
              15490 ;       DOS command overlay request
              15500 ;
1A7F E5       15510 CKMOD@  PUSH    HL
1A8Ø 67       15520         LD      H,A             ;Save command value
1A81 78       15530         LD      A,B
1A82 32D21A   15540         LD      (EXOVR2+1),A    ;Set overlay #
1A85 7C       15550         LD      A,H
1A86 F6Ø1     15560         OR      1               ;Set for SYS6 & SYS7
1A88 FE89     15570         CP      89H             ;Is it either?
1A8A 7C       15580         LD      A,H             ;Get back the correct #
1A8B 2813     15590         JR      Z,EXOVR         ;Sys6/7 req? Use ISAM!
1A8D FE8A     15600         CP      8AH             ;Sys8 also ISAM
1A8F 28ØF     15610         JR      Z,EXOVR
1A91 3A69ØØ   15620         LD      A,(OVRLY$)      ;P/u current overlay
1A94 AC       15630         XOR     H               ;Ck if it's the one
1A95 E6ØF     15640         AND     ØFH             ;  we need to execute
1A97 7C       15650         LD      A,H
1A98 327Ø1A   15660         LD      (OVRLYOLD),A    ;Update current tempy
1A9B 21ØØ1E   15670         LD      HL,OVERLAY      ;Init to SYSx entry
1A9E 283A     15680         JR      Z,EXOVR3        ;Go exec if resident
              15690 ;
              15700 ;       Execute a system overlay
              15710 ;
1AAØ D5       15720 EXOVR   PUSH    DE
1AA1 C5       15730         PUSH    BC
1AA2 E6ØF     15740         AND     ØFH             ;Get right nybble
1AA4 CB5F     15750         BIT     3,A             ;Check for SYSØ-7
1AA6 2802     15760         JR      Z,EXOVR1        ;  w/o changing carry
1AA8 C618     15770         ADD     A,18H           ;Adjust for sys8-15
1AAA 3293ØØ   15780 EXOVR1  LD      (SFCB$+7),A
1AAD 47       15790         LD      B,A             ;Set DEC for directory
1AAE 3E2Ø     15800         LD      A,2ØH           ;Set bit 5 of FCB+1
1ABØ 328DØØ   15810         LD      (SFCB$+1),A
1AB3 ED62     15820         SBC     HL,HL           ;Carry is clear here
1AB5 2296ØØ   15830         LD      (SFCB$+1Ø),HL   ;Zero NRN
1AB8 4C       15840         LD      C,H             ;Init for drive Ø
1AB9 CDBB18   15850         CALL    @DIRRD          ;Read dir entry
1ABC 2Ø1A     15860         JR      NZ,EXERR        ;Go if error
```

System Loader and associated routines

```
1ABE 7E       15870          LD      A,(HL)          ;Was overlay purged?
1ABF E650     15880          AND     50H             ;  or is it non-system?
1AC1 EE50     15890          XOR     50H
1AC3 3EØ7     15900          LD      A,7             ;Init "deleted error
1AC5 2011     15910          JR      NZ,EXERR
1AC7 7D       15920          LD      A,L
1AC8 C616     15930          ADD     A,22            ;Point to 1st extent
1ACA 6F       15940          LD      L,A
1ACB 119AØØ   15950          LD      DE,SFCB$+14     ;Extent field in FCB
1ACE CDE11A   15960          CALL    PAT1            ;Stuff 1st two extents
1AD1 0600     15970 EXOVR2   LD      B,Ø             ;P/u ISAM # or zero
1AD3 1E8C     15980          LD      E,SFCB$&ØFFH
1AD5 CD561B   15990          CALL    LOADER          ;Read system overlay
1AD8 C1       16000 EXERR    POP     BC
1AD9 D1       16010          POP     DE
1ADA 22751A   16020 EXOVR3   LD      (TRANSFR+1),HL  ;Stuff overlay entry pt
1ADD E1       16030          POP     HL
1ADE C8       16040          RET     Z
1ADF 1816     16050          JR      SYSERR          ;Go if I/O error on read
              16060 ;
              16070 ;        Routine to calculate 1st two extents of SYS file
              16080 ;
1AE1 CDEC1A   16090 PAT1     CALL    PAT1A           ;Move first extent
1AE4 E61F     16100          AND     1FH             ;Compute # of granules
1AE6 3C       16110          INC     A
1AE7 12       16120          LD      (DE),A          ;And store in FCB
1AE8 13       16130          INC     DE
1AE9 AF       16140          XOR     A
1AEA 12       16150          LD      (DE),A
1AEB 13       16160          INC     DE
1AEC CDEF1A   16170 PAT1A    CALL    PAT1B           ;Move second extent
1AEF 7E       16180 PAT1B    LD      A,(HL)
1AFØ 12       16190          LD      (DE),A
1AF1 23       16200          INC     HL
1AF2 13       16210          INC     DE
1AF3 C9       16220          RET
              16230 ;
              16240 ;        System error display routine
              16250 ;        The NOP is provided so an intercept routine vector
              16260 ;         may be patched in during program development
              16270 ;
1AF4 3E2B     16280 SVCERR   LD      A,43            ;SVC error
1AF6 ØØ       16290          NOP
1AF7 E63F     16300 SYSERR   AND     3FH             ;Strip excess bits
1AF9 21191B   16310          LD      HL,ERRNUM       ;Pack error number
1AFC CDC207   16320          CALL    @HEX8           ;  into message
1AFF 21131B   16330          LD      HL,SYSERR$
1BØ2 CDØØØ5   16340          CALL    @LOGOT          ;Log the error & ABORT
1BØ5 3180Ø3   16350          LD      SP,STACK$       ;reset stack
1BØ8 21FFFF   16360 @ABORT   LD      HL,-1
1BØB 3E93     16370 @EXIT    LD      A,93H           ;Exit to DOS
1BØD EF       16380          RST     40
              16390 ;
1BØE E1       16400 POPERR   POP     HL              ;Pop extended error
1BØF F5       16410 @ERROR   PUSH    AF              ;Save the error code
1B10 3E96     16420          LD      A,96H           ;Display the error number
1B12 EF       16430          RST     40
              16440 ;
1B13 45       16450 SYSERR$  DM      'Error '
```

System Loader and associated routines

```
     72 72 6F 72 2Ø
1B19 78       16460 ERRNUM  DM      'xxH',CR
     78 48 ØD
              16470 ;
              16480 ;       Routine to RUN a program
              16490 ;
1B1D E5       16500 @RUN    PUSH    HL              ;Save register pair
1B1E 217CØØ   16510         LD      HL,SFLAG$
1B21 CBD6     16520         SET     2,(HL)          ;Turn on RUN flag bit
1B23 CD381B   16530         CALL    @LOAD           ;Load the program module
1B26 E3       16540         EX      (SP),HL         ;Put traadr on the stack
              16550 ;
              16560 ;       Note: The error code is set to NOT abort. Errors
              16570 ;        will be passed back to the calling module after
              16580 ;        @ERROR. Note that HL will contain the error #.
              16590 ;
1B27 2ØE5     16600         JR      NZ,POPERR
              16610 ;
              16620 ;       Place the INBUF$ pointer in register pair BC
              16630 ;
1B29 Ø12ØØ4   16640         LD      BC,INBUF$       ;Reflect buffer pointer
              16650 ;
              16660 ;       Get TRAADR then test if we need to go to DEBUG
              16670 ;
1B2C 3A7CØØ   16680         LD      A,(SFLAG$)
1B2F CB4F     16690         BIT     1,A             ;Go to the program if
1B31 CØ       16700         RET     NZ              ;  its EXEC only access
1B32 CB7F     16710         BIT     7,A             ;  else test if DEBUG
1B34 C23ØØØ   16720         JP      NZ,@RST3Ø       ;  is on & go to it
1B37 C9       16730         RET                     ;  else go to program
              16740 ;
              16750 ;       This routine LOADs a Load Module Format file
              16760 ;
1B38 Ø6ØØ     16770 @LOAD   LD      B,Ø             ;LRL=256
1B3A 217CØØ   16780         LD      HL,SFLAG$
1B3D CBC6     16790         SET     Ø,(HL)          ;Don't set "file open"
1B3F 21ØØ1D   16800         LD      HL,SBUFF$       ;Set buffer to system
1B42 CD8A19   16810         CALL    @OPEN           ;Open the file
1B45 D5       16820         PUSH    DE              ;Save FCB pointer
1B46 CC561B   16830         CALL    Z,LOADER        ;Load if no OPEN error
1B49 D1       16840         POP     DE              ;Restore FCB pointer
1B4A C8       16850         RET     Z               ;Back if no error
1B4B 6F       16860         LD      L,A             ;Xfer the error code
1B4C 26ØØ     16870         LD      H,Ø
1B4E F6CØ     16880         OR      ØCØH            ;Set RETurn & abbrev
1B5Ø FED8     16890         CP      ØD8H            ;Change "file not in dir"
1B52 CØ       16900         RET     NZ              ;  to "program not found"
1B53 C6Ø7     16910         ADD     A,7
1B55 C9       16920         RET
              16930 ;
              16940 ;       System command file loader
              16950 ;
1B56 78       16960 LOADER  LD      A,B             ;Set overlay # (Ø on non
1B57 32B31B   16970         LD      (LDR14+1),A     ;  SYStem file)
1B5A D5       16980         PUSH    DE              ;Save IX & xfer FCB to IX
1B5B DDE3     16990         EX      (SP),IX
1B5D 11FF1D   17000         LD      DE,SBUFF$+255   ;Init to end of buffer
1B6Ø CD6F1B   17010         CALL    LDRØ1           ;Do the load
1B63 DDE1     17020         POP     IX              ;Recover IX
```

System Loader and associated routines

```
1B65 C9       17Ø3Ø         RET
              17Ø4Ø ;
              17Ø5Ø ;       Routine to ignore the LMF record
              17Ø6Ø ;
1B66 CDD61B   17Ø7Ø LDRØ5   CALL    LDR15           ;Get length of "comment"
1B69 47       17Ø8Ø         LD      B,A
1B6A CDD61B   17Ø9Ø LDRØ6   CALL    LDR15           ;Read & ignore that many
1B6D 1ØFB     171ØØ         DJNZ    LDRØ6           ;  bytes then fall thru
              1711Ø ;
              1712Ø ;       Routine to parse LMF record types
              1713Ø ;
1B6F CDD61B   1714Ø LDRØ1   CALL    LDR15           ;Get record type
1B72 FEØ1     1715Ø LDRØ2   CP      1               ;Start of block?
1B74 281F     1716Ø         JR      Z,LDRØ8
1B76 FEØ2     1717Ø         CP      2               ;Start of TRAADR?
1B78 2814     1718Ø LDRØ3   JR      Z,LDRØ7
1B7A FEØ4     1719Ø         CP      4               ;End of LIB member?
1B7C 282A     172ØØ         JR      Z,LDR12
1B7E FEØ8     1721Ø         CP      8               ;Begin ISAM table entry?
1B8Ø 2828     1722Ø         JR      Z,LDR13
1B82 FEØA     1723Ø         CP      1Ø              ;End of ISAM map?
1B84 28Ø4     1724Ø         JR      Z,LDRØ4
1B86 FE2Ø     1725Ø         CP      2ØH             ;Ignore all other control
1B88 38DC     1726Ø         JR      C,LDRØ5
1B8A 3E22     1727Ø LDRØ4   LD      A,22H           ;Load file format err
1B8C B7       1728Ø         OR      A
1B8D C9       1729Ø         RET
              173ØØ ;
              1731Ø ;       Grab transfer address
              1732Ø ;
1B8E CDD61B   1733Ø LDRØ7   CALL    LDR15           ;Bypass 2nd X'Ø2'
1B91 CDE81B   1734Ø         CALL    GETADR          ;P/u transfer address
1B94 C9       1735Ø         RET                     ;Ret Z or NZ
              1736Ø ;
              1737Ø ;       Grab load block
              1738Ø ;
1B95 CDD61B   1739Ø LDRØ8   CALL    LDR15           ;P/u block len
1B98 47       174ØØ         LD      B,A
1B99 CDE81B   1741Ø         CALL    GETADR          ;P/u load address
1B9C CØ       1742Ø         RET     NZ
1B9D Ø5       1743Ø         DEC     B               ;Adj length for adr
1B9E Ø5       1744Ø         DEC     B
1B9F CDD61B   1745Ø LDRØ9   CALL    LDR15           ;P/u block byte
1BA2 77       1746Ø         LD      (HL),A
1BA3 23       1747Ø         INC     HL
1BA4 1ØF9     1748Ø         DJNZ    LDRØ9           ;Loop until block end
1BA6 18C7     1749Ø         JR      LDRØ1
              175ØØ ;
1BA8 E1       1751Ø LDR12   POP     HL
1BA9 C9       1752Ø         RET
              1753Ø ;
              1754Ø ;       Routine to check ISAM table match
              1755Ø ;
1BAA CDD61B   1756Ø LDR13   CALL    LDR15           ;Get record length
1BAD 47       1757Ø         LD      B,A
1BAE CDD61B   1758Ø         CALL    LDR15           ;Get ISAM number
1BB1 Ø5       1759Ø         DEC     B               ;  & decrement counter
1BB2 FEØØ     176ØØ LDR14   CP      Ø               ;Either ISAM# or Ø
1BB4 2ØB4     1761Ø         JR      NZ,LDRØ6        ;Go if not a match
```

System Loader and associated routines

```
1BB6 CDE81B   17620          CALL    GETADR          ;  else get the TRAADR
1BB9 E5       17630          PUSH    HL              ;  & save it
1BBA CCE81B   17640          CALL    Z,GETADR        ;Get the NRN for member
1BBD 2Ø27     17650          JR      NZ,LODERR
1BBF CDD61B   17660          CALL    LDR15           ;Get the sector offset
1BC2 5F       17670          LD      E,A             ;Update pointer offset
1BC3 C5       17680          PUSH    BC
1BC4 44       17690          LD      B,H             ;Xfer NRN position needed
1BC5 4D       17700          LD      C,L
1BC6 D5       17710          PUSH    DE              ;Save buffer ptr offset
1BC7 DDE5     17720          PUSH    IX
1BC9 D1       17730          POP     DE              ;P/u FCB into DE
1BCA CD3414   17740          CALL    @POSN           ;Position to ISAM rec
1BCD D1       17750          POP     DE              ;Rcvr buffer ptr offset
1BCE C1       17760          POP     BC
1BCF 2Ø15     17770          JR      NZ,LODERR
1BD1 CDDB1B   17780          CALL    LDR17           ;Read the sector
1BD4 189C     17790          JR      LDRØ2           ;Now go read the member
              17800 ;
              17810 ;        Routine to get the next file byte
              17820 ;
1BD6 1C       17830 LDR15    INC     E               ;Bump buf pointer
1BD7 28Ø2     17840          JR      Z,LDR17         ;Read sector if needed
1BD9 1A       17850 LDR16    LD      A,(DE)          ;P/U byte from buffer
1BDA C9       17860          RET
1BDB E5       17870 LDR17    PUSH    HL              ;Save regs
1BDC D5       17880          PUSH    DE
1BDD C5       17890          PUSH    BC
1BDE CD7513   17900          CALL    NXTSECT         ;Read next record
1BE1 C1       17910          POP     BC              ;Restore regs
1BE2 D1       17920          POP     DE
1BE3 E1       17930          POP     HL
1BE4 28F3     17940          JR      Z,LDR16         ;Bypass if no error
1BE6 C1       17950 LODERR   POP     BC              ;Pop return address
1BE7 C9       17960          RET
              17970 ;
              17980 ;        Routine to get an address field
              17990 ;
1BE8 CDD61B   18000 GETADR   CALL    LDR15           ;Get low order byte
1BEB 6F       18010          LD      L,A
1BEC CDD61B   18020          CALL    LDR15           ;Get hi order byte
1BEF 67       18030          LD      H,A
1BFØ BF       18040          CP      A
1BF1 C9       18050          RET
              18060 ;
              18070 ;        BOOT code brings back the ROM
              18080 ;
4300          18090 MOD3BUF
1BF2 21FBØ3   18100 @IPL     LD      HL,BOOTCOD      ;Code to toggle in ROM
1BF5 114043   18110          LD      DE,MOD3BUF      ;Buffer used by ROM
1BF8 D5       18120          PUSH    DE              ;This is return address
1BF9 Ø1Ø5ØØ   18130          LD      BC,BOOTLEN
1BFC EDBØ     18140          LDIR                    ;Transfer boot code and
1BFE C9       18150          RET                     ;  jump to it
              18160 ;
              18170 ;        End of loader module
              18180 ;
1BFF          Ø3610          SUBTTL  '<System front end & task processor>'
```

System front end & task processor

```
1BFF            03630 *GET    TASKER:3
                18190 ;TASKER/ASM - LS-DOS 6.2
                18200 ;
                18210 ;       Interrupt task table, IM 1
                18220 ;
1BFF            18230 CORE$   DEFL    $
004E            18240         ORG     TCB$
004E E91C       18250         DW      NOTASK,NOTASK,NOTASK,NOTASK
     E91C E91C E91C
0056 E91C       18260         DW      NOTASK,NOTASK,NOTASK,NOTASK
     E91C E91C E91C
005E E91C       18270         DW      NOTASK,NOTASK,TYPTSK$,NOTASK
     E91C 260B E91C
1BFF            18280         ORG     CORE$
                18290 ;
                18300 ;       Model IV task processor
                18310 ;
                18320 RST38@
1BFF E3         18330         EX      (SP),HL
1C00 22AF07     18340         LD      (PCSAVE$),HL    ;Save for TRACE
1C03 E3         18350         EX      (SP),HL
1C04 E5         18360         PUSH    HL              ;Save HL for now
1C05 F5         18370         PUSH    AF              ;Save AF for now
1C06 217700     18380         LD      HL,NFLAG$       ;Show the system we
1C09 CBF6       18390         SET     6,(HL)          ;  are in the TASKER
1C0B 210202     18400         LD      HL,LBANK$       ;P/U & save the current
1C0E 7E         18410         LD      A,(HL)          ;  logical bank #
1C0F 3600       18420         LD      (HL),0
1C11 F5         18430         PUSH    AF
1C12 217800     18440         LD      HL,OPREG$       ;Get current memory
1C15 7E         18450         LD      A,(HL)
1C16 F5         18460         PUSH    AF              ;  config & save
1C17 E68C       18470         AND     8CH             ;Strip bits 0, 1, 4-6
1C19 F603       18480         OR      3               ;Bring up regular 64K
1C1B 77         18490         LD      (HL),A
1C1C D384       18500
00E0            18510 INTLAT  EQU     0E0H
1C1E DBE0       18520                                 ;Get interrupt latch
1C20 2F         18530         CPL                     ;Mod IV is reverse
1C21 213C00     18540         LD      HL,INTIM$       ;Store state of int
1C24 77         18550         LD      (HL),A
1C25 2C         18560         INC     L               ;Advance to int mask
1C26 A6         18570         AND     (HL)            ;Mask the latch bits
1C27 2808       18580         JR      Z,TSTBRK        ;Go if nothing interptd
1C29 2C         18590 NXTVCT  INC     L               ;Ck on INTVC$
1C2A 1F         18600         RRA                     ;Ck if device interrupted
1C2B 381C       18610         JR      C,ACTVTSK
1C2D 2C         18620 NXTMSK  INC     L               ;Ck all 8 bits of mask
1C2E B7         18630         OR      A               ;When fin, ck overhead
1C2F 20F8       18640         JR      NZ,NXTVCT       ;  task routine
                18650 ;
1C31 CDD607     18660 TSTBRK  CALL    KCK@            ;Test <BREAK>, <SHIFT>
1C34 202A       18670         JR      NZ,BREAK?       ;Go if break
1C36 F1         18680 TSKEXIT POP     AF              ;Get previous mem config
1C37 327800     18690         LD      (OPREG$),A      ;  & restore to it
1C3A D384       18700
1C3C F1         18710         POP     AF
1C3D 320202     18720         LD      (LBANK$),A
1C40 217700     18730         LD      HL,NFLAG$       ;Now leaving the TASKER
1C43 CBB6       18740         RES     6,(HL)          ; show the system
```

System front end & task processor

```
1C45 F1       18750          POP     AF              ;Restore previous regs
1C46 E1       18760          POP     HL
1C47 FB       18770          EI
1C48 C9       18780 RETINST  RET
              18790 ;
              18800 ;
              18810 ;        Found active INTVC$
              18820 ;
1C49 F5       18830 ACTVTSK  PUSH    AF              ;Save the regs
1C4A C5       18840          PUSH    BC
1C4B D5       18850          PUSH    DE
1C4C E5       18860          PUSH    HL
1C4D DDE5     18870          PUSH    IX
1C4F 11581C   18880          LD      DE,POPREGS      ;Stack return vector
1C52 D5       18890          PUSH    DE
1C53 5E       18900          LD      E,(HL)          ;P/u INTVC pointer vector
1C54 2C       18910          INC     L
1C55 56       18920          LD      D,(HL)
1C56 EB       18930          EX      DE,HL           ;Shift it to HL
1C57 E9       18940          JP      (HL)            ;Go to service routine
              18950 ;
              18960 ;        Register restoral after service routine
              18970 ;
1C58 DDE1     18980 POPREGS  POP     IX
1C5A E1       18990          POP     HL
1C5B D1       19000          POP     DE
1C5C C1       19010          POP     BC
1C5D F1       19020          POP     AF
1C5E 18CD     19030          JR      NXTMSK          ;Loop to next mask bit
              19040 ;
              19050 ;        BREAK key detected
              19060 ;
1C60 3008     19070 BREAK?   JR      NC,GOTBRK       ;Go if <BREAK> only
1C62 C5       19080          PUSH    BC              ;Was <SHIFT-BREAK>
1C63 F3       19090          DI
1C64 CDB819   19100          CALL    TAPDRV          ;Reselect drive
1C67 C1       19110          POP     BC
1C68 18CC     19120          JR      TSKEXIT
              19130 ;
              19140 ;        BREAK during tasking - enter DEBUG? - user BREAK?
              19150 ;
1C6A 3A7C00   19160 GOTBRK   LD      A,(SFLAG$)      ;Check if BREAK key is
1C6D E610     19170          AND     10H             ;  disabled to inhibit
1C6F 20C5     19180          JR      NZ,TSKEXIT      ;  DEBUG or BREAK vector
1C71 219F19   19190          LD      HL,@DBGHK       ;Merge DEBUG flag &
1C74 B6       19200          OR      (HL)            ;  hook (X'00' or X'C9')
1C75 36C9     19210          LD      (HL),0C9H       ;Turn off DEBUG
1C77 23       19220          INC     HL              ;Point to @DEBUG vector &
1C78 2814     19230          JR      Z,EXITBRK       ;  go if DEBUG is active
              19240 ;
1C7A 3AB007   19250          LD      A,(PCSAVE$+1)   ;Don't allow vectored break
1C7D FE24     19260          CP      MAXCOR$<-8      ; if old PC is in SYSRES
1C7F 38B5     19270          JR      C,TSKEXIT
1C81 210F04   19280          LD      HL,HIGH$+1      ; or if old PC is
1C84 BE       19290          CP      (HL)            ; above HIGH$
1C85 30AF     19300          JR      NC,TSKEXIT
1C87 210000   19310          LD      HL,0            ;  else ck if BREAK is
1C88          19320 BRKVEC$  EQU     $-2
1C8A 7C       19330          LD      A,H             ;  to be trapped by user
```

System front end & task processor

```
1C8B B5       19340          OR      L
1C8C 28A8     19350          JR      Z,TSKEXIT
1C8E F1       19360 EXITBRK  POP     AF              ;Discard old mem config
1C8F F1       19370          POP     AF              ;Restore reg AF
1C90 F1       19380          POP     AF
1C91 E3       19390          EX      (SP),HL         ;P/u HL & stack vector
1C92 FB       19400          EI
1C93 C9       19410          RET                     ;To DEBUG or BREAK vector
              19420 ;
              19430 ;        Real Time Clock interrupt processor
              19440 ;
1C94          19450 RTCPROC  EQU     $
1C94 DBEC     19460                                  ;Clear the RTC interrupt
1C96 3E0B     19470          LD      A,11            ;Task 11 executes every
1C98 CDBB1C   19480          CALL    RTCTASK         ;  RTC interrupt
1C9B 212B00   19490          LD      HL,TIMSL$
1C9E CB06     19500          RLC     (HL)            ;Ck on time slice
1CA0 D0       19510          RET     NC              ;Ignore if nothing
1CA1 111307   19520          LD      DE,TIMTSK$      ;  on this interrupt
1CA4 D5       19530          PUSH    DE              ;  else init for clocker
1CA5 3E08     19540          LD      A,8             ;Task 8 at INT/2 if fast
1CA7 CDBB1C   19550          CALL    RTCTASK
1CAA 3E09     19560          LD      A,9             ;Task 9 at INT/2 if fast
1CAC CDBB1C   19570          CALL    RTCTASK
1CAF 3E0A     19580          LD      A,10            ;Task 10 at INT/2 if fast
1CB1 CDBB1C   19590          CALL    RTCTASK
1CB4 212C00   19600          LD      HL,TIMER$       ;Bump the timer at INT/2
1CB7 34       19610          INC     (HL)
1CB8 7E       19620          LD      A,(HL)          ;P/u the heart beat
1CB9 E607     19630          AND     7               ;For this interrupt,
1CBB 07       19640 RTCTASK  RLCA                    ;  consider 0-7 only
1CBC C64E     19650          ADD     A,TCB$&0FFH     ;Add offset to table
1CBE 6F       19660          LD      L,A
1CBF 2600     19670          LD      H,TCB$<-8
1CC1 22EC1C   19680          LD      (@RPTSK+1),HL
1CC4 5E       19690          LD      E,(HL)          ;P/u task vector addr
1CC5 2C       19700          INC     L
1CC6 56       19710          LD      D,(HL)
1CC7 D5       19720          PUSH    DE
1CC8 DDE1     19730          POP     IX              ;Also to IX
1CCA EB       19740          EX      DE,HL
1CCB 5E       19750          LD      E,(HL)          ;P/u task entry point
1CCC 23       19760          INC     HL
1CCD 56       19770          LD      D,(HL)
1CCE EB       19780          EX      DE,HL
1CCF E9       19790          JP      (HL)            ;Go to task
              19800 ;
1CD0 D1       19810 @KLTSK   POP     DE              ;Remove ret
1CD1 3AEC1C   19820          LD      A,(@RPTSK+1)    ;Pt to task tbl entry
1CD4 D64E     19830          SUB     TCB$&0FFH
1CD6 0F       19840          RRCA                    ;  of last task
              19850 ;
1CD7 11E91C   19860 @RMTSK   LD      DE,NOTASK       ;Remove entry
              19870 ;
1CDA FE0C     19880 @ADTSK   CP      12              ;Too large a task?
1CDC D0       19890          RET     NC              ;Ret if too big else
1CDD 07       19900          RLCA                    ;  add to task table
1CDE C64E     19910          ADD     A,TCB$&0FFH     ;Add the offset
1CE0 6F       19920          LD      L,A             ;Estab ptr to vector
```

System front end & task processor

```
1CE1 26ØØ     19930          LD      H,TCB$<-8
1CE3 F3       19940 CHGTASK  DI
1CE4 73       19950          LD      (HL),E          ;Vector adr to ptr tbl
1CE5 2C       19960          INC     L
1CE6 72       19970          LD      (HL),D
1CE7 FB       19980          EI
1CE8 C9       19990          RET
              20000 ;
1CE9 E81C     20010 NOTASK   DW      $-1             ;Current task vector
              20020 ;
1CEB 210000   20030 @RPTSK   LD      HL,Ø            ;P/u last task done
1CEE 5E       20040          LD      E,(HL)          ;P/u task vector addr
1CEF 23       20050          INC     HL
1CFØ 56       20060          LD      D,(HL)
1CF1 EB       20070          EX      DE,HL
1CF2 D1       20080          POP     DE              ;Pop ret addr
1CF3 18EE     20090          JR      CHGTASK
              20100 ;
              20110 ;        Routine to check if task slot active
              20120 ;
1CF5 Ø7       20130 @CKTSK   RLCA                    ;Task number * 2
1CF6 C64F     20140          ADD     A,TCB$&ØFFH+1   ;Index into task table
1CF8 6F       20150          LD      L,A
1CF9 26ØØ     20160          LD      H,TCB$<-8
1CFB 3E1C     20170          LD      A,NOTASK<-8     ;Check match of high
1CFD BE       20180          CP      (HL)            ;  order only
1CFE C9       20190          RET                     ; Z or NZ result
              Ø3640          IFGT    $,1DØØH+START$
              Ø3650          ERR     'SYSRES memory overflow
              Ø3660          ENDIF
1CFF          Ø3670 CORE$    DEFL    $
1CFF ØØ       Ø3680          DC      1DØØH-CORE$,Ø
1CFF          Ø3690          ORG     CORE$
1DØØ          Ø3700          ORG     1DØØH+START$
1DØØ          Ø3710 SBUFF$   EQU     $
Ø1ØØ          Ø3720          DS      256             ;Page disk I/O buffer
23ØØ          Ø3730 DIRBUF$  EQU     MAXCOR$-256     ;Another file buffer
              Ø3740 ;
              Ø3750 ;        get the system initialization module
              Ø3760 ;
1EØØ          Ø3770 OVERLAY  EQU     $
1EØØ          Ø3780          SUBTTL  '<System initialization routines>'
```

System initialization routines

```
1EØØ            Ø38ØØ *GET    SYSINIT4:3
                2Ø2ØØ ;SYSINIT4/ASM - LS-DOS 6.2
                2Ø21Ø ;
                2Ø22Ø ;       This is the initialization part of SYSRES
                2Ø23Ø ;
ØØF1            2Ø24Ø TRKREG                          ;FDC track register
F4Ø1            2Ø25Ø KB1                             ;Keyboard row 1
F46Ø            2Ø26Ø KB67                            ;Keyboard rows 6&7
F44Ø            2Ø27Ø KB7                             ;Keyboard row 7
ØØ1D            2Ø28Ø BOL                             ;Beginning of line
                2Ø29Ø ;
1EØØ            2Ø3ØØ         ORG     1EØØH+START$
                2Ø31Ø ;
1EØØ F3         2Ø32Ø         DI
1EØ1 21E9ØF     2Ø33Ø         LD      HL,@RSTNMI      ;Reset NMI vector to
1EØ4 2267ØØ     2Ø34Ø         LD      (@NMI+1),HL     ;  SYSRES's needs
1EØ7 211ØØ4     2Ø35Ø         LD      HL,PAKNAM$      ;Pt to pack name
1EØA 11BEF8     2Ø36Ø         LD      DE,2*8Ø+CRTBGN$+3Ø
1EØD Ø1Ø8ØØ     2Ø37Ø         LD      BC,8
1E1Ø EDBØ       2Ø38Ø         LDIR                    ;move pack name to crt
1E12 ØEØ8       2Ø39Ø         LD      C,8             ;B contains Ø already
1E14 13         2Ø4ØØ         INC     DE              ;Leave 2 spaces
1E15 13         2Ø41Ø         INC     DE
1E16 EDBØ       2Ø42Ø         LDIR                    ;Move pack date to crt
                2Ø43Ø ;
                2Ø44Ø ;       Initialization routines
                2Ø45Ø ;
1E18 AF         2Ø46Ø         XOR     A               ;Clear out stack area
1E19 2181Ø3     2Ø47Ø         LD      HL,STACK$+1     ;Stack start +1
1E1C 2D         2Ø48Ø CLRLOOP DEC     L               ;Move down a byte
1E1D 77         2Ø49Ø         LD      (HL),A          ;Now loop and fill
1E1E 2ØFC       2Ø5ØØ         JR      NZ,CLRLOOP      ;  and fill with Ø's
                2Ø51Ø ;
1E2Ø ED56       2Ø52Ø         IM      1
1E22 318ØØ3     2Ø53Ø         LD      SP,STACK$       ;Set the stack area
1E25 AF         2Ø54Ø         XOR     A
1E26 32Ø2Ø2     2Ø55Ø         LD      (LBANK$),A      ;Set logical bank #
1E29 D3E4       2Ø56Ø                                 ;Disable INTRQ & DRQ
                2Ø57Ø ;
1E2B 2138Ø2     2Ø58Ø         LD      HL,S1DCB$
1E2E 77         2Ø59Ø ZERDCB  LD      (HL),A          ;Zero spare dcb area
1E2F 2C         2Ø6ØØ         INC     L
1E3Ø 2ØFC       2Ø61Ø         JR      NZ,ZERDCB
                2Ø62Ø ;
1E32 3A76ØØ     2Ø63Ø         LD      A,(MODOUT$)     ;Set hi-speed
1E35 D3EC       2Ø64Ø                                 ;  and external bus
1E37 3A8ØØØ     2Ø65Ø         LD      A,(WRINT$)
1E3A D3EØ       2Ø66Ø                                 ;Enable RTC interrupts
1E3C 3A78ØØ     2Ø67Ø         LD      A,(OPREG$)      ;Set memory configuration
1E3F 47         2Ø68Ø         LD      B,A
1E4Ø 3EA7       2Ø69Ø         LD      A,ØA7H          ;Value for AUX/RAM
1E42 ØE84       2Ø7ØØ         LD      C,@OPREG        ;Set the memory mgt port
1E44 ED41       2Ø71Ø                                 ;Bring up reg RAM
1E46 21FFFF     2Ø72Ø         LD      HL,-1           ;Ck for extended RAM
1E49 22ØEØ4     2Ø73Ø         LD      (HIGH$),HL
1E4C 221CØØ     2Ø74Ø         LD      (PHIGH$),HL
                2Ø75Ø ;       Check the BANKS
1E4F 56         2Ø76Ø         LD      D,(HL)          ;Save what's in RAM
1E5Ø 3655       2Ø77Ø         LD      (HL),55H        ;Stuff in reg RAM
1E52 ED79       2Ø78Ø                                 ;Switch in alt RAM
```

System initialization routines

```
1E54 5E       2Ø79Ø          LD      E,(HL)          ;Save the byte there
1E55 77       2Ø8ØØ          LD      (HL),A          ;Stuff alt RAM
1E56 ED41     2Ø81Ø                                  ;Switch to reg RAM
1E58 BE       2Ø82Ø          CP      (HL)            ;See what's there now
1E59 72       2Ø83Ø          LD      (HL),D          ;Restore original value
1E5A ED79     2Ø84Ø                                  ;Back to alt RAM
1E5C 73       2Ø85Ø          LD      (HL),E          ;Restore original byte
1E5D ED41     2Ø86Ø                                  ;Back to reg RAM
1E5F 3EFE     2Ø87Ø          LD      A,ØFEH          ;Init BAR$ for bank-Ø
1E61 28Ø2     2Ø88Ø          JR      Z,$+4           ;Bypass if only 64K
1E63 3EF8     2Ø89Ø          LD      A,ØF8H          ;Init BAR$ for bank Ø-2
1E65 32Ø1Ø2   2Ø9ØØ          LD      (BAR$),A        ;Load Bank Avail RAM
1E68 32ØØØ2   2Ø91Ø          LD      (BUR$),A        ;Load Bank Used RAM
1E6B 3A6FØØ   2Ø92Ø          LD      A,(FEMSK$)      ;P/u port FE mask
1E6E D3FE     2Ø93Ø                                  ;  & set it
1E7Ø ØØ       2Ø94Ø          DC      3,Ø             ;Space for a JUMP
     ØØ ØØ
              2Ø95Ø ;
              2Ø96Ø ;        Update DCT$ info for SYSTEM drive
              2Ø97Ø ;
1E73 3A9D43   2Ø98Ø          LD      A,(BOOTST$)     ;P/u Boot Step rate
1E76 E6Ø3     2Ø99Ø          AND     3               ;Strip all but it
1E78 47       21ØØØ          LD      B,A             ;Save tempy
1E79 2173Ø4   21Ø1Ø          LD      HL,DCT$+3       ;Pt to DCT step
1E7C 7E       21Ø2Ø          LD      A,(HL)          ;P/u DCT Step
1E7D E6FC     21Ø3Ø          AND     ØFCH            ;Strip step rate
1E7F BØ       21Ø4Ø          OR      B               ;Merge in Boot step
1E8Ø 77       21Ø5Ø          LD      (HL),A          ;Update DCT
1E81 DBF1     21Ø6Ø          IN      A,(TRKREG)      ;Update DCT with current
1E83 3275Ø4   21Ø7Ø          LD      (DCT$+5),A      ;  track posn of head
              21Ø8Ø ;
1E86 11Ø8Ø2   21Ø9Ø          LD      DE,KIDCB$       ;Flush type,init ptrs.
1E89 3EØ3     211ØØ          LD      A,3
1E8B CD23Ø6   2111Ø          CALL    @CTL
1E8E FB       2112Ø          EI                      ;Interrupts on
              2113Ø ;
              2114Ø ;        P/u CONFIG status & set ZERO byte
              2115Ø ;
1E8F 21Ø1Ø4   2116Ø          LD      HL,ZERO$
1E92 7E       2117Ø          LD      A,(HL)          ;set to NOP if SYSGEN'd
1E93 36ØØ     2118Ø          LD      (HL),Ø          ;Make always zero byte
1E95 F5       2119Ø          PUSH    AF              ;save SYSGEN flag
              212ØØ ;
              2121Ø ;        Check if date prompt is to be suppressed
              2122Ø ;
1E96 3AC2Ø4   2123Ø          LD      A,(DTPMT$)      ;No prompt for date?
1E99 B7       2124Ø          OR      A
              2125Ø ;
              2126Ø ;        Check on currency of date
              2127Ø ;
1E9A 213ØØ    2128Ø          LD      HL,DATE$        ;Point to Year
1E9D 4E       2129Ø          LD      C,(HL)          ;  & save in reg C
1E9E 36ØØ     213ØØ          LD      (HL),Ø          ;  while resetting to zero
1EAØ 23       2131Ø          INC     HL              ;Bump to day
1EA1 46       2132Ø          LD      B,(HL)          ;  & save in reg B
1EA2 36ØØ     2133Ø          LD      (HL),Ø          ;  while resetting to zero
1EA4 23       2134Ø          INC     HL              ;Bump to Month
1EA5 7E       2135Ø          LD      A,(HL)          ;  & save in Reg A
1EA6 36ØØ     2136Ø          LD      (HL),Ø          ;  while resetting to zero
```

System initialization routines

```
1EA8 C2991F   21370         JP      NZ,TIMIN        ;Ck time if DATE=OFF
1EAB 2EFF     21380         LD      L,CFGFCB$+31&ØFFH       ;Reset pointer
              21390 ;
              21400         IF      @INTL
              21410         LD      (HL),B          ;Stuff day
              21420         DEC     HL
              21430         LD      (HL),A          ;Stuff month
              21440         ELSE
1EAD 77       21450         LD      (HL),A          ;Stuff month
1EAE 2B       21460         DEC     HL
1EAF 7Ø       21470         LD      (HL),B          ;Stuff day
              21480         ENDIF
              21490 ;
1EBØ 2B       21500         DEC     HL
1EB1 71       21510         LD      (HL),C          ;Stuff Year
1EB2 EB       21520         EX      DE,HL           ;  & point DE to CFGFCB$+29
1EB3 3D       21530         DEC     A               ;Check for month range <1-12>
1EB4 FEØC     21540         CP      12              ;OK if Ø-11 now
1EB6 38ØE     21550         JR      C,DATIN1
              21560 ;
1EB8 211B15   21570 DATIN   LD      HL,21<8!27      ;Set video row,col
1EBB 111E21   21580         LD      DE,DATEPR       ;DATE? question
1EBE Ø13ØØ8   21590         LD      BC,8<+8!'Ø'     ;Set buf len & char
1EC1 CDAE2Ø   21600         CALL    GETPARM         ;Get response
1EC4 3ØF2     21610         JR      NC,DATIN        ;Jump on format error
1EC6 1A       21620 DATIN1  LD      A,(DE)          ;Is year a leap year?
1EC7 4F       21630         LD      C,A             ;Save year for later
1EC8 D65Ø     21640         SUB     8Ø              ;Reduce for range test
1ECA FEØ8     21650         CP      8
1ECC 3ØEA     21660         JR      NC,DATIN
1ECE E6Ø3     21670         AND     3
1EDØ 3E1C     21680         LD      A,28            ;Init February
1ED2 2ØØ6     21690         JR      NZ,NOTLEAP
1ED4 2137ØØ   21700         LD      HL,DATE$+3+1    ;Set leap flag
1ED7 CBFE     21710         SET     7,(HL)
1ED9 3C       21720         INC     A               ;Feb to 29 days
1EDA 21Ø3Ø4   21730 NOTLEAP LD      HL,MAXDAY$+2    ;Set Feb max day #
1EDD 77       21740         LD      (HL),A
              21750 ;
              21760         IF      @INTL
              21770         NOP                     ;Keep same length
              21780         ELSE
1EDE 13       21790         INC     DE              ;Bump to DAY
              21800         ENDIF
1EDF 13       21810         INC     DE              ;Bump to month & get it
1EEØ 1A       21820         LD      A,(DE)
1EE1 47       21830         LD      B,A             ;Save month in reg B
1EE2 3D       21840         DEC     A               ;Range check
1EE3 FEØC     21850         CP      12
1EE5 3ØD1     21860         JR      NC,DATIN        ;Go if Ø or >12
1EE7 2B       21870         DEC     HL              ;Point to Jan entry
1EE8 85       21880         ADD     A,L             ;Index the month
1EE9 6F       21890         LD      L,A
              21900 ;
              21910         IF      @INTL
              21920         INC     DE              ;Point to day
              21930         ELSE
1EEA 1B       21940         DEC     DE              ;Point to day
              21950         ENDIF
```

System initialization routines

```
                21960 ;
1EEB 1A         21970           LD      A,(DE)          ;P/u day entry
1EEC 3D         21980           DEC     A               ;Reduce for test (Ø->FF)
1EED BE         21990           CP      (HL)
1EEE 30C8       22000           JR      NC,DATIN        ;Go if too large (or Ø)
                22010 ;
                22020 ;         Range checks OK - move into DATE$
                22030 ;
1EFØ 213500     22040           LD      HL,DATE$+2
1EF3 3C         22050           INC     A               ;Compensate for DEC A
1EF4 70         22060           LD      (HL),B          ;Stuff month
1EF5 2D         22070           DEC     L
1EF6 77         22080           LD      (HL),A          ;Stuff day
1EF7 2D         22090           DEC     L
1EF8 71         22100           LD      (HL),C          ;Stuff year
                22110 ;
                22120 ;         Date is in DATE$ - display it
                22130 ;
1EF9 79         22140           LD      A,C
1EFA F5         22150           PUSH    AF              ;  & save it for later
1EFB E603       22160           AND     3               ;Check on leap year
1EFD 2103Ø4     22170           LD      HL,MAXDAY$+2    ;Init and adjust Feb
1FØØ 361C       22180           LD      (HL),28         ;  as required
1FØ2 2ØØ1       22190           JR      NZ,$+3
1FØ4 34         22200           INC     (HL)            ;Bump to 29
1FØ5 3A35ØØ     22210           LD      A,(DATE$+2)     ;P/u month & xfer to B
1FØ8 47         22220           LD      B,A
1FØ9 3A34ØØ     22230           LD      A,(DATE$+1)     ;P/u day of month
                22240 ;
                22250 ;         Compute day of year and day of week
                22260 ;
1FØC 6F         22270           LD      L,A             ;Start off with days
1FØD 26ØØ       22280           LD      H,Ø             ;  in this month
1FØF 11Ø1Ø4     22290           LD      DE,MAXDAY$
1F12 1A         22300 DAYLP     LD      A,(DE)
1F13 85         22310           ADD     A,L             ;8 bit add to 16 bit
1F14 6F         22320           LD      L,A
1F15 8C         22330           ADC     A,H             ;Add in hi order & carry
1F16 95         22340           SUB     L               ;Subtract off lo order
1F17 67         22350           LD      H,A             ;Update hi order
1F18 13         22360           INC     DE
1F19 1ØF7       22370           DJNZ    DAYLP
1F1B EB         22380           EX      DE,HL           ;Move day of year to DE
1F1C 213600     22390           LD      HL,DATE$+3      ;  and store
1F1F 73         22400           LD      (HL),E
1F20 23         22410           INC     HL
1F21 7A         22420           LD      A,D             ;Get bit "8"
1F22 B6         22430           OR      (HL)            ;  and OR it in
1F23 77         22440           LD      (HL),A          ;Then put it back
1F24 EB         22450           EX      DE,HL           ;Get DOY back to HL
1F25 F1         22460           POP     AF              ;Pop the year & mask
1F26 E607       22470           AND     7               ;Compute day of week
1F28 5F         22480           LD      E,A             ;  offset
1F29 C603       22490           ADD     A,3
1F2B ØF         22500           RRCA
1F2C ØF         22510           RRCA
1F2D E603       22520           AND     3
1F2F 83         22530           ADD     A,E
1F30 5F         22540           LD      E,A             ;And add it in
```

System initialization routines

```
1F31 16ØØ      22550          LD      D,Ø             ;Add into HL
1F33 19        22560          ADD     HL,DE
1F34 23        22570          INC     HL              ;To start in right place
1F35 ØE07      22580          LD      C,7             ;Now divide by 7 (B=Ø)
1F37 ED42      22590 DIV7     SBC     HL,BC           ;Subtract weeks (7-days)
1F39 3ØFC      22600          JR      NC,DIV7         ;  until under flow
1F3B 7D        22610          LD      A,L
1F3C C608      22620          ADD     A,8             ;Add back to get 1-7
1F3E 47        22630          LD      B,A             ;Save in reg B
1F3F Ø7        22640          RLCA                    ;Shift to bits 1-3
1F4Ø 4F        22650          LD      C,A             ;Save tempy
1F41 213700    22660          LD      HL,DATE$+3+1
1F44 7E        22670          LD      A,(HL)          ;Pack into field
1F45 E6F1      22680          AND     ØF1H
1F47 B1        22690          OR      C
1F48 77        22700          LD      (HL),A
1F49 C5        22710          PUSH    BC
1F4A 211B15    22720          LD      HL,21<8!27      ;Set video row,col
1F4D Ø603      22730          LD      B,3             ;Set function code 3
1F4F CD99ØB    22740          CALL    @VDCTL          ;  to position cursor
1F52 C1        22750          POP     BC
1F53 21C704    22760          LD      HL,DAYTBL$
1F56 CDFF2Ø    22770          CALL    SPACE4          ;Write out the DAY
1F59 3E2C      22780          LD      A,','
1F5B CD4206    22790          CALL    @DSP
1F5E 3E2Ø      22800          LD      A,' '
1F6Ø CD4206    22810          CALL    @DSP
1F63 3A35ØØ    22820          LD      A,(DATE$+2)     ;P/u month number
1F66 47        22830          LD      B,A
1F67 2EDC      22840          LD      L,MONTBL$&ØFFH  ;Reset HL for month table
1F69 CDØ721    22850          CALL    DSPMDY          ;Write out the month name
1F6C 3E2Ø      22860          LD      A,' '
1F6E CD4206    22870          CALL    @DSP
1F71 3A34ØØ    22880          LD      A,(DATE$+1)     ;P/u day
1F74 Ø5        22890          DEC     B               ;From Ø to X'FF'
1F75 Ø4        22900 DIV1Ø    INC     B               ;Divide by 1Ø
1F76 D6ØA      22910          SUB     1Ø              ;  with quotient in B
1F78 3ØFB      22920          JR      NC,DIV1Ø
1F7A F5        22930          PUSH    AF              ;Save remainder (-1Ø)
1F7B 78        22940          LD      A,B             ;P/u quotient
1F7C C63Ø      22950          ADD     A,'Ø'           ;Change to ASCII
1F7E FE3Ø      22960          CP      'Ø'             ;Zero?
1F8Ø C44206    22970          CALL    NZ,@DSP         ;Display if not
1F83 F1        22980          POP     AF              ;Get back remainder
1F84 C63A      22990          ADD     A,3AH           ;Change to ASCII
1F86 CD4206    23000          CALL    @DSP
1F89 211721    23010          LD      HL,PARTYR       ;Part of year
1F8C CD2DØ5    23020          CALL    @DSPLY
1F8F 3A33ØØ    23030          LD      A,(DATE$)       ;Form last year digit
1F92 E607      23040          AND     7
1F94 C63Ø      23050          ADD     A,'Ø'
1F96 CD4206    23060          CALL    @DSP            ;  and display it
               23070 ;
               23080 ;        Prompt for time
               23090 ;
1F99 3AC3Ø4    23100 TIMIN    LD      A,(TMPMT$)      ;Time to be prompted
1F9C B7        23110          OR      A
1F9D 2Ø28      23120          JR      NZ,SELDCT       ;Skip if not
1F9F 211B16    23130 TIMINØ   LD      HL,22<8!27
```

System initialization routines

```
1FA2 113Ø21   2314Ø          LD      DE,TIMEPR       ;Set prompt message
1FA5 Ø13ØØ8   2315Ø          LD      BC,8<+8!'Ø'     ;Set len & separ char
1FA8 CDAE2Ø   2316Ø          CALL    GETPARM
1FAB 3ØF2     2317Ø          JR      NC,TIMINØ       ;Loop on format error
1FAD 21FFØØ   2318Ø          LD      HL,CFGFCB$+31
1FBØ 3E17     2319Ø          LD      A,23
1FB2 BE       2320Ø          CP      (HL)            ;Test hour range
1FB3 38EA     2321Ø          JR      C,TIMINØ
1FB5 2B       2322Ø          DEC     HL
1FB6 3E3B     2323Ø          LD      A,59
1FB8 BE       2324Ø          CP      (HL)            ;Test minute range
1FB9 38E4     2325Ø          JR      C,TIMINØ
1FBB 2B       2326Ø          DEC     HL
1FBC BE       2327Ø          CP      (HL)            ;Test the second range
1FBD 38EØ     2328Ø          JR      C,TIMINØ
1FBF 112DØØ   2329Ø          LD      DE,TIME$        ;Move the time value
1FC2 Ø13ØØØ   2330Ø          LD      BC,3            ;  into the TIME$ field
1FC5 EDBØ     2331Ø          LDIR
              2332Ø ;
              2333Ø ;        Check on any AUTO command
              2334Ø ;
1FC7 212ØØ4   2335Ø SELDCT   LD      HL,INBUF$
1FCA 7E       2336Ø          LD      A,(HL)          ;Pt to 1st byte of AUTO
1FCB FE2A     2337Ø          CP      '*'             ;BREAK disable?
1FCD 2ØØF     2338Ø          JR      NZ,CKDCR
1FCF 23       2339Ø          INC     HL
1FDØ 3EE6     2340Ø          LD      A,ØE6H          ;Set BREAK bit in flag by
1FD2 325A2Ø   2341Ø          LD      (STUB1+1),A     ;  changing RES 4,(SFLAG$)
              2342Ø                                  ;  to SET 4,(SFLAG$)
1FD5 181A     2343Ø          JR      AUTO?
1FD7 CD17Ø8   2344Ø GETKB17  CALL    ENADIS_DO_RAM
1FDA 3A41F4   2345Ø          LD      A,(KB1!KB7)     ;scan row 1 & 7
1FDD C9       2346Ø          RET
1FDE CDD71F   2347Ø CKDCR    CALL    GETKB17         ;Strobe keyboard
1FE1 CB67     2348Ø          BIT     4,A             ;Is 'D' depressed?
1FE3 E5       2349Ø          PUSH    HL              ;Save auto command pt
1FE4 21Ø81B   2350Ø          LD      HL,@ABORT       ;P/u abort address
1FE7 E3       2351Ø          EX      (SP),HL         ;Swap them around
1FE8 C2A019   2352Ø          JP      NZ,@DEBUG       ;DEBUG on <D>
1FEB D1       2353Ø          POP     DE              ;Stack integrity
1FEC 2F       2354Ø          CPL
1FED E6Ø1     2355Ø          AND     1               ;No AUTO if <ENTER>
1FEF 28Ø3     2356Ø          JR      Z,NOAUT1
1FF1 7E       2357Ø AUTO?    LD      A,(HL)          ;Any AUTO command?
1FF2 FEØD     2358Ø          CP      CR              ;None if equal
1FF4 D1       2359Ø NOAUT1   POP     DE              ;Get back SYSGEN flag
1FF5 7A       2360Ø          LD      A,D             ;  & move into reg A
1FF6 11ØB1B   2361Ø          LD      DE,@EXIT        ;Where to go after boot
1FF9 Ø1ØØØØ   2362Ø          LD      BC,Ø            ;Init BC(HL)=Ø for @EXIT
1FFC 28ØF     2363Ø          JR      Z,NOAUT         ;Go if no AUTO
1FFE E5       2364Ø          PUSH    HL              ;Save buffer pointer
1FFF 218420   2365Ø          LD      HL,CURSET       ;Point to cusor setting
2ØØ2 34       2366Ø          INC     (HL)            ;Bump it down a line
2ØØ3 E1       2367Ø          POP     HL              ;Recover INBUF$ pointer
2ØØ4 117E19   2368Ø          LD      DE,@CMNDI       ;Lo order of @CMNDI
2ØØ7 D5       2369Ø          PUSH    DE              ;Put on stack for RET
2ØØ8 44       2370Ø          LD      B,H             ;Put INBUF$ pointer on
2ØØ9 4D       2371Ø          LD      C,L             ;  stack for @CMNDI
2ØØA 112DØ5   2372Ø          LD      DE,@DSPLY       ;But do this first
```

System initialization routines

```
200D D5        23730 NOAUT   PUSH    DE              ;Put on stack for RET
200E C5        23740         PUSH    BC              ;Either INBUF$ or Ø
200F 21562Ø    23750         LD      HL,STUB
2012 115Ø43    23760         LD      DE,MOD3BUF+8Ø   ;Must move out of way
2015 Ø158ØØ    23770         LD      BC,STUBLEN      ;  amount to move
2018 D5        23780         PUSH    DE              ;Add ret vector to stack
2019 EDBØ      23790         LDIR                    ;Move stub up
201B CD4C2Ø    23800         CALL    GETKB67
201E 117ØØ4    23810         LD      DE,DCT$         ;Set up to move DCT's
2021 21ØØ43    23820         LD      HL,MOD3BUF      ;  from configed area
2024 Ø15ØØØ    23830         LD      BC,8Ø           ;Count for DCTs (8*1Ø)
2027 D9        23840         EXX                     ;Keep in alternate set
2028 E682      23850         AND     82H             ;Load config if zero
202A CØ        23860         RET     NZ              ;No config > Go back
202B 21ØØ15    23870         LD      HL,21<8         ;Set to line 21
202E Ø6Ø3      23880         LD      B,3             ;Position cursor
2030 CD99ØB    23890         CALL    @VDCTL
2033 213F2Ø    23900         LD      HL,CONFIG$      ;Show sysgen message
2036 CD2DØ5    23910         CALL    @DSPLY
2039 11EØØØ    23920         LD      DE,CFGFCB$      ;Set up to load config
203C C3381B    23930         JP      @LOAD           ;Go to load config
               23940 ;
203F 2A        23950 CONFIG$ DB      '** SYSGEN **',Ø3       ; Config DSP
     2A 2Ø 53 59 53 47 45 4E
     2Ø 2A 2A Ø3
               23960 ;
204C 216ØF4    23970 GETKB67 LD      HL,KB67         ;Check <CLEAR> key
204F 4F        23980         LD      C,A
2050 CD1708    23990         CALL    ENADIS_DO_RAM
2053 79        24000         LD      A,C
2054 B6        24010         OR      (HL)            ;Key down OR not SYSGENed
2055 C9        24020         RET
               24030 ;
               24040 ;       Final initialization code
               24050 ;
2056 217CØØ    24060 STUB    LD      HL,SFLAG$
2059 CBA6      24070 STUB1   RES     4,(HL)          ;Test or SET Break bit
               24080                                 ;  without changing Z/NZ
205B 2ØØC      24090         JR      NZ,NOTSG        ;Go if no SYSGEN found
205D 217ØØØ    24100         LD      HL,MODOUT$      ;P/u ptr to port mask
2060 7E        24110         LD      A,(HL)          ;P/u mask byte
2061 D3EC      24120                                 ;Speed it up
2063 D9        24130         EXX                     ;Set to move DCT's
2064 EDBØ      24140         LDIR                    ;Move 'em
2066 CD86ØØ    24150         CALL    @ICNFG          ;Init config
               24160 NOTSG
2069 ØE07      24170         LD      C,7
               24180 SETCYLØ
206B CD1E1A    24190         CALL    @GTDCT
206E FDCBØ35E  24200         BIT     3,(IY+3)        ;If hard drive, don't stuff FF
2072 2ØØB      24210         JR      NZ,NOFF         ;  & don't restore
2074 FD36Ø5FF  24220         LD      (IY+5),ØFFH     ;Set in case no restore
2078 3AC4Ø4    24230         LD      A,(RSTOR$)      ;Do we restore the drives?
207B B7        24240         OR      A
207C CCC819    24250         CALL    Z,@RSTOR        ;Restore drives 1-7
207F ØD        24260 NOFF    DEC     C
2080 2ØE9      24270         JR      NZ,SETCYLØ
2082 21ØØ15    24280         LD      HL,21<8         ;Set cursor
2084           24290 CURSET  EQU     $-1
```

System initialization routines

```
2Ø85 Ø6Ø3     243ØØ          LD      B,3
2Ø87 CD99ØB   2431Ø          CALL    @VDCTL
              2432Ø ;
              2433Ø ;        Detect Model 4 or 4P and adjust TFLAG$
              2434Ø ;        Look at 'MODEL' at 4Ø18H. If so MOD-4P (5)
              2435Ø ;
              2436Ø ;
2Ø8A 114D4F   2437Ø          LD      DE,'OM'
2Ø8D 2A184Ø   2438Ø                                  ;P/u 4P rom leftover
2Ø9Ø ED52     2439Ø          SBC     HL,DE         ;Check if it's 'MO'
2Ø92 3EØ4     2440Ø          LD      A,4           ;Init for MOD 4 REG.
2Ø94 2ØØ2     2441Ø          JR      NZ,MOD4REG
2Ø96 3EØ5     2442Ø          LD      A,5           ;Change to MOD 4P
2Ø98 327DØØ   2443Ø MOD4REG  LD      (TFLAG$),A
              2444Ø ;
2Ø9B 2138ØØ   2445Ø          LD      HL,@RST38
2Ø9E 36C3     2446Ø          LD      (HL),ØC3H     ;Activate task processor
2ØAØ E1       2447Ø          POP     HL            ;Pop INBUF$
2ØA1 C9       2448Ø          RET                   ;To @CMD or @DSPLY,@CMNDI
2ØA2 ØØ       2449Ø          DC      12,Ø          ;Space for more code
     ØØ ØØ ØØ ØØ ØØ ØØ ØØ ØØ
     ØØ ØØ ØØ
2ØAE          2450Ø STUBEND  EQU     $
ØØ58          2451Ø STUBLEN  EQU     STUBEND-STUB
              2452Ø ;
              2453Ø ;        Date & Time prompting
              2454Ø ;
2ØAE C5       2455Ø GETPARM  PUSH    BC            ;Save separator char
2ØAF D5       2456Ø          PUSH    DE            ;Save message pointer
2ØBØ Ø6Ø3     2457Ø          LD      B,3
2ØB2 CD99ØB   2458Ø          CALL    @VDCTL        ;Position the cursor
2ØB5 E1       2459Ø          POP     HL            ;Recover message pointer
2ØB6 CD2DØ5   2460Ø          CALL    @DSPLY        ;  & display the message
2ØB9 21ØØ1E   2461Ø          LD      HL,OVERLAY    ;Buffer for reply
2ØBC C1       2462Ø          POP     BC
2ØBD C5       2463Ø          PUSH    BC
2ØBE CD85Ø5   2464Ø          CALL    @KEYIN        ;Get reply & wait a bit
2ØC1 AF       2465Ø          XOR     A             ;  disable test
2ØC2 BØ       2466Ø          OR      B
2ØC3 C1       2467Ø          POP     BC            ;  of key prior to AUTO
2ØC4 C8       2468Ø          RET     Z             ;Ret with NC if no entry
2ØC5 C5       2469Ø          PUSH    BC
2ØC6 Ø64Ø     2470Ø          LD      B,4ØH
2ØC8 CD82Ø3   2471Ø          CALL    @PAUSE        ;  to let finger off
2ØCB C1       2472Ø          POP     BC
              2473Ø ;
              2474Ø ;        Routine to parse DATE entry
              2475Ø ;
2ØCC 11FFØØ   2476Ø PARSDAT  LD      DE,CFGFCB$+31 ;Point to buf end
2ØCF Ø6Ø3     2477Ø          LD      B,3           ;Process 3 fields
2ØD1 D5       2478Ø PRSD1    PUSH    DE            ;Save pointer
              2479Ø ;
              2480Ø ;        Routine to parse a digit pair
              2481Ø ;
2ØD2 CDF82Ø   2482Ø          CALL    PRSD3         ;Get a digit
2ØD5 3ØØF     2483Ø          JR      NC,PRSD2      ;Jump if bad digit
2ØD7 5F       2484Ø          LD      E,A           ;Multiply by ten
2ØD8 Ø7       2485Ø          RLCA
2ØD9 Ø7       2486Ø          RLCA
```

System initialization routines

```
20DA 83          24870          ADD     A,E
20DB Ø7          24880          RLCA
20DC 5F          24890          LD      E,A
20DD CDF820      24900          CALL    PRSD3           ;Get another digit
20EØ 3ØØ4        24910          JR      NC,PRSD2        ;Jump on bad digit
20E2 83          24920          ADD     A,E             ;Accumulate new digit
20E3 5F          24930          LD      E,A             ;Save 2-digit value
20E4 37          24940          SCF                     ;Show valid
20E5 7B          24950          LD      A,E             ;Xfer field value
20E6 D1          24960 PRSD2    POP     DE              ;Recover pointer
20E7 DØ          24970          RET     NC              ;Ret if bad digit pair
20E8 12          24980          LD      (DE),A          ;Else stuff the value
20E9 Ø5          24990          DEC     B               ;Loop countdown
20EA 37          25000          SCF
20EB C8          25010          RET     Z               ;Ret when through
20EC 1B          25020          DEC     DE              ;Backup the pointer
20ED 7E          25030          LD      A,(HL)          ;Ck for valid separator
20EE 23          25040          INC     HL              ;Bump pointer
20EF FE3A        25050          CP      ':'             ;Check for colon ':'
20F1 28DE        25060          JR      Z,PRSD1         ;  loop if match
20F3 B9          25070          CP      C               ;Separator char required
20F4 3ØØ6        25080          JR      NC,PRSD4        ;Exit if bad char
20F6 18D9        25090          JR      PRSD1           ;  else loop now
20F8 7E          25100 PRSD3    LD      A,(HL)          ;P/u a digit &
20F9 23          25110          INC     HL              ;  convert to binary
20FA D63Ø        25120          SUB     3ØH
20FC FEØA        25130 PRSD4    CP      1Ø
20FE C9          25140          RET
                 25150 ;
                 25160 ;        Routine to display month or day of week
                 25170 ;
20FF E5          25180 SPACE4   PUSH    HL              ;Print 4 SPACES
2100 214221      25190          LD      HL,SPACE4$      ;  point to string
2103 CD2DØ5      25200          CALL    @DSPLY
2106 E1          25210          POP     HL
2107 Ø5          25220 DSPMDY   DEC     B               ;Point to Bth entry
2108 7D          25230          LD      A,L             ;  in table
2109 8Ø          25240          ADD     A,B
210A 8Ø          25250          ADD     A,B
210B 8Ø          25260          ADD     A,B
210C 6F          25270          LD      L,A
210D Ø6Ø3        25280          LD      B,3             ;Print 3 characters
210F 7E          25290 DSPM1    LD      A,(HL)
2110 23          25300          INC     HL
2111 CD42Ø6      25310          CALL    @DSP
2114 1ØF9        25320          DJNZ    DSPM1
2116 C9          25330          RET
2117 2C          25340 PARTYR   DB      ', 198',3Ø,3
     2Ø 31 39 38 1E Ø3
                 25350 ;
                 25360          IF      @INTL
                 25370 DATEPR   DB      3Ø,'Date DD/MM/YY ? ',3
                 25380          ELSE
211E 1E          25390 DATEPR   DB      3Ø,'Date MM/DD/YY ? ',3
     44 61 74 65 2Ø 4D 4D 2F
     44 44 2F 59 59 2Ø 3F 2Ø
     Ø3
                 25400          ENDIF
                 25410 ;
```

System initialization routines

```
213Ø 1E       2542Ø TIMEPR  DB      3Ø,'Time HH:MM:SS ? ',3
     54 69 6D 65 2Ø 48 48 3A
     4D 4D 3A 53 53 2Ø 3F 2Ø
     Ø3
2142 2Ø       2543Ø SPACE4$ DB      '   ',Ø3,Ø3     ;3 or 4 space string
     2Ø 2Ø Ø3 Ø3
2147 ØØ       2544Ø         DC      32,ØØ           ;Space for message, or??
     ØØ ØØ ØØ ØØ ØØ ØØ ØØ ØØ
     ØØ ØØ ØØ ØØ ØØ ØØ ØØ ØØ
     ØØ ØØ ØØ ØØ ØØ ØØ ØØ ØØ
     ØØ ØØ ØØ ØØ ØØ ØØ ØØ
2167          Ø381Ø         SUBTTL  '<Misc. lowcore routines>'
```

Misc. lowcore routines

```
2167            Ø383Ø *GET    SOUND:3
                2545Ø ;SOUND/ASM - LS-DOS 6.2
                2546Ø ;
                2547Ø ;       Contains IPL, PAUSE, SOUND, and DECHEX routines
                2548Ø ;       Will be loaded into lowcore area along with SYSRES
                2549Ø ;
                2550Ø *MOD
ØØ9Ø            2551Ø SNDPORT
Ø38Ø            2552Ø         ORG     STACK$
Ø38Ø ØØØØ       2553Ø         DW      ØØ              ;Stack gaurd
                2554Ø ;
                2555Ø ;       Pause routine
                2556Ø ;
Ø382 C5         2557Ø @PAUSE  PUSH    BC              ;Save the count
                2558Ø ;       SRL     B               ;Adjust for WAIT STATES
                2559Ø ;       RR      C
Ø383 3A7CØØ     2560Ø         LD      A,(SFLAG$)      ;If system (FAST)
Ø386 CB5F       2561Ø         BIT     3,A             ;  then double it
Ø388 C48CØ3     2562Ø         CALL    NZ,CDLOOP       ;Call if fast
Ø38B C1         2563Ø         POP     BC              ;Restore the count
Ø38C ØB         2564Ø CDLOOP  DEC     BC              ;Count down routine
Ø38D 78         2565Ø         LD      A,B
Ø38E B1         2566Ø         OR      C
Ø38F 2ØFB       2567Ø         JR      NZ,CDLOOP
Ø391 C9         2568Ø         RET
                2569Ø ;
                2570Ø ;       @SOUND SVC-1Ø4 - Operates sound generator
                2571Ø ;       B => sound function
                2572Ø ;       Bits Ø-2 <Ø-7> = note # (Ø highest)
                2573Ø ;       Bits 3-7 <Ø-31> = relative sound duration
                2574Ø ;       All regs except A left unchanged
                2575Ø ;       Z-flag set on exit
                2576Ø ;       Note that interrupts disabled during duration
                2577Ø ;
Ø392 C5         2578Ø @SOUND  PUSH    BC              ;Save registers
Ø393 E5         2579Ø         PUSH    HL
Ø394 78         2580Ø         LD      A,B             ;P/u sound data
Ø395 E6Ø7       2581Ø         AND     7               ;  & strip off duration
Ø397 Ø7         2582Ø         RLCA                    ;Adj for 2-byte fields
Ø398 21D1Ø3     2583Ø         LD      HL,SNDTAB
Ø39B 4F         2584Ø         LD      C,A
Ø39C 78         2585Ø         LD      A,B             ;Pick up duration data
Ø39D Ø6ØØ       2586Ø         LD      B,Ø             ;Index into tone table
Ø39F Ø9         2587Ø         ADD     HL,BC           ;  to get note-on/off
Ø3AØ 4E         2588Ø         LD      C,(HL)          ;P/u note-on/off data
Ø3A1 23         2589Ø         INC     HL
Ø3A2 6E         2590Ø         LD      L,(HL)          ;P/u note duration
Ø3A3 ØF         2591Ø         RRCA                    ;Rotate sound duration
Ø3A4 ØF         2592Ø         RRCA                    ;  into bits Ø-4
Ø3A5 ØF         2593Ø         RRCA
Ø3A6 E61F       2594Ø         AND     1FH             ;Strip off sound #
Ø3A8 3C         2595Ø         INC     A               ;Adjust for offset
Ø3A9 67         2596Ø         LD      H,A             ;Set sound counter
Ø3AA 3A7CØØ     2597Ø         LD      A,(SFLAG$)      ;If fast, double values
Ø3AD E6Ø8       2598Ø         AND     8H
Ø3AF 28Ø6       2599Ø         JR      Z,$A1
Ø3B1 CB24       2600Ø         SLA     H
Ø3B3 CB25       2601Ø         SLA     L
Ø3B5 CB21       2602Ø         SLA     C
Ø3B7 F3         2603Ø $A1     DI                      ;Can't interrupt timing
```

Misc. lowcore routines

```
Ø3B8 E5       26Ø4Ø $A2     PUSH    HL              ;Save note duration
Ø3B9 41       26Ø5Ø $A3     LD      B,C             ;Play tone
Ø3BA 3EØ1     26Ø6Ø         LD      A,1             ;Hold output high
Ø3BC D39Ø     26Ø7Ø         OUT     (SNDPORT),A     ;  for count of (B)
Ø3BE 1ØFE     26Ø8Ø         DJNZ    $
Ø3CØ 41       26Ø9Ø         LD      B,C             ;Hold output low for
Ø3C1 3C       261ØØ         INC     A               ;  for count of (B)
Ø3C2 D39Ø     2611Ø         OUT     (SNDPORT),A
Ø3C4 1ØFE     2612Ø         DJNZ    $
Ø3C6 2D       2613Ø         DEC     L               ;Dec the duration
Ø3C7 2ØFØ     2614Ø         JR      NZ,$A3
Ø3C9 E1       2615Ø         POP     HL              ;Get sound/note durations
Ø3CA 25       2616Ø         DEC     H               ;Count down the sound
Ø3CB 2ØEB     2617Ø         JR      NZ,$A2          ;  duration counter
Ø3CD FB       2618Ø         EI                      ;Restore interrupts
Ø3CE E1       2619Ø         POP     HL
Ø3CF C1       262ØØ         POP     BC
Ø3DØ C9       2621Ø         RET
              2622Ø ;
              2623Ø ;       Note table
              2624Ø ;
ØØB4          2625Ø SNDOFF  EQU     18Ø             ;Sound duration offset
ØØ1C          2626Ø TONER   EQU     28
Ø3D1 5Ø       2627Ø SNDTAB  DB      1Ø8-TONER       ;Note Ø (highest)
Ø3D2 4C       2628Ø         DB      Ø-SNDOFF
Ø3D3 56       2629Ø         DB      114-TONER
Ø3D4 48       263ØØ         DB      252-SNDOFF
Ø3D5 5C       2631Ø         DB      12Ø-TONER
Ø3D6 44       2632Ø         DB      248-SNDOFF
Ø3D7 62       2633Ø         DB      126-TONER
Ø3D8 4Ø       2634Ø         DB      244-SNDOFF
Ø3D9 6B       2635Ø         DB      135-TONER
Ø3DA 3C       2636Ø         DB      24Ø-SNDOFF
Ø3DB 72       2637Ø         DB      142-TONER
Ø3DC 38       2638Ø         DB      236-SNDOFF
Ø3DD 79       2639Ø         DB      149-TONER
Ø3DE 34       264ØØ         DB      232-SNDOFF
Ø3DF 8Ø       2641Ø         DB      156-TONER       ;Note 7 (lowest)
Ø3EØ 3Ø       2642Ø         DB      228-SNDOFF
ØØ4F          2643Ø SNDLEN  EQU     $-@SOUND
              2644Ø ;
              2645Ø ;       Process decimal assignment
              2646Ø ;
Ø3E1 Ø1ØØØØ   2647Ø @DECHEX LD      BC,Ø            ;Init value to zero
Ø3E4 7E       2648Ø DEC1    LD      A,(HL)          ;P/u a char
Ø3E5 D63Ø     2649Ø         SUB     3ØH             ;Cvrt to binary
Ø3E7 D8       265ØØ         RET     C               ;Return if < "Ø"
Ø3E8 FEØA     2651Ø         CP      1Ø              ;Ck for bad decimal
Ø3EA DØ       2652Ø         RET     NC              ;Ret if not Ø-9
Ø3EB C5       2653Ø         PUSH    BC              ;Exchange BC & HL
Ø3EC E3       2654Ø         EX      (SP),HL         ;  & save HL on stack
Ø3ED 29       2655Ø         ADD     HL,HL           ;Multiply by 1Ø
Ø3EE 29       2656Ø         ADD     HL,HL
Ø3EF Ø9       2657Ø         ADD     HL,BC
Ø3FØ 29       2658Ø         ADD     HL,HL
Ø3F1 Ø6ØØ     2659Ø         LD      B,Ø             ;Merge in new digit
Ø3F3 4F       266ØØ         LD      C,A             ;New digit to C
Ø3F4 Ø9       2661Ø         ADD     HL,BC           ;  & add it in
Ø3F5 44       2662Ø         LD      B,H             ;Current value to BC
```

Misc. lowcore routines

```
Ø3F6 4D       26630          LD      C,L
Ø3F7 E1       26640          POP     HL              ;Recover HL pointer
Ø3F8 23       26650          INC     HL
Ø3F9 18E9     26660          JR      DEC1            ;Loop
              26670 ;
              26680 ;        Special Boot code to be moved to 43ØØh by IPL
              26690 ;
Ø3FB F3       26700 BOOTCOD  DI                      ;Boot stub for @IPL to
Ø3FC AF       26710          XOR     A               ;  to move to 43ØØh
Ø3FD D384     26720          OUT     (@OPREG),A
Ø3FF C7       26730          RST     Ø
ØØØ5          26740 BOOTLEN  EQU     $-BOOTCOD
              26750 ;
Ø4ØØ          Ø3840          SUBTTL  '<Sign-on LOGO display>'
Ø4ØØ          Ø3850 *GET     LOGO:3
              26760 ;RSLOGOB/ASM  3-D RS LOGO used on 6.2.Ø - 1/2Ø/84
              26770 *LIST    OFF
              27540 *LIST    ON
              Ø3860 ;
1EØØ          Ø3870          END     OVERLAY
```

Origin	Symbolic Label	Value	Line#	Usage	Line#'s of References
LDOS60	@$SYS	08F0	0040		
$MAIN	+@@1	0000	0387		
LDOS60	+@@1	0000	0050		
$MAIN	+@@2	0000	0387		
LDOS60	+@@2	0000	0060		
$MAIN	+@@3	0000	0387		
LDOS60	+@@3	0000	0070		
$MAIN	+@@4	0000	0387		
LDOS60	+@@4	0000	0080		
LOADER	@ABORT	1B08	1636	LOADER	1276
				SYSINIT4	2350
TASKER	@ADTSK	1CDA	1988	LOADER	1278
LDOS60	@BANK	0877	0090	LOADER	1296 1428
FILPOSN	@BKSP	1486	0356	LOADER	1286
LOADER	@BREAK	196F	1331	LOADER	1296
LDOS60	@BYTEIO	1300	0100	$MAIN	0329
LDOS60	@CHNIO	0689	0110	LOADER	1276
LDOS60	@CKBRKC	0553	0120	LOADER	1297
LOADER	@CKDRV	1993	1359	LOADER	1279
FILPOSN	@CKEOF	158F	0546	LOADER	1286
TASKER	@CKTSK	1CF5	2013	LOADER	1278
LOADER	@CLOSE	1999	1363	LOADER	1286
LDOS60	@CLS	0545	0130	LOADER	1297
LOADER	@CMNDI	197E	1345	LOADER	1277
				SYSINIT4	2368
LOADER	@CMNDR	197B	1343	LOADER	1277
LDOS60	@CTL	0623	0140	LOADER	1272
				SYSINIT4	2111
LDOS60	@DATE	07A8	0150	LOADER	1275
LOADER	@DBGHK	199F	1367	LOADER	1531 1545
				TASKER	1919
LOADER	@DCINIT	19C0	1391	LOADER	1281
LOADER	@DCRES	19C4	1393	LOADER	1281
LOADER	@DCSTAT	19B5	1385	LOADER	1281
LOADER	@DCTBYT	1A2B	1478	FILPOSN	0590 1039 1197 1204 1226 1259
LOADER	@DEBUG	19A0	1368	$MAIN	0057
				LOADER	1277
				SYSINIT4	2352
SOUND	@DECHEX	03E1	2647	LOADER	1295
FILPOSN	@DIRCYL	18F7	1196	FILPOSN	1106 1129 1164
FILPOSN	@DIRRD	18BB	1149	FILPOSN	0437 0769 0981 1005
				LOADER	1292 1585
FILPOSN	@DIRWR	1803	1019	FILPOSN	0451 0961 1001
				LOADER	1293
LDOS60	@DIV16	06E3	0016	FILPOSN	0401 0407 0593
				LOADER	1294
FILPOSN	@DIV8	1927	1237	LOADER	1294
LOADER	@DODIR	19AF	1378	LOADER	1279
LOADER	@DOKEY	19A9	1374		
LDOS60	@DSP	0642	0017	LOADER	1271
				SYSINIT4	2279 2281 2287 2297 2300 2306
				SYSINIT4	2531
LDOS60	@DSPLY	052D	0018	LOADER	1273
				SYSINIT4	2302 2372 2391 2460 2520
LOADER	@ERROR	1B0F	1641	LOADER	1277
LOADER	@EXIT	1B0B	1637	LOADER	1276
				SYSINIT4	2361
LOADER	@FEXT	1984	1349	LOADER	1290
LOADER	@FLAGS	196A	1328	LOADER	1296

Origin	Symbolic Label	Value	Line#	Usage	Line#'s of References
LOADER	@FNAME	199C	1365Ø	LOADER	1291Ø
LDOS6Ø	@FRENCH	ØØØØ	ØØ19Ø	$MAIN	Ø148Ø Ø346Ø
LOADER	@FSPEC	1981	1347Ø	LOADER	129ØØ
FILPOSN	@GATRD	1874	11Ø1Ø	FILPOSN	Ø823Ø
				LOADER	1292Ø
FILPOSN	@GATWR	1875	1102Ø	FILPOSN	Ø958Ø
				LOADER	1293Ø
LDOS6Ø	@GERMAN	ØØØØ	ØØ2ØØ	$MAIN	Ø151Ø Ø343Ø
LDOS6Ø	@GET	Ø638	ØØ21Ø	LOADER	1271Ø
LOADER	@GTDCB	199Ø	1357Ø	LOADER	1291Ø
LOADER	@GTDCT	1A1E	1458Ø	LOADER	1291Ø 1439Ø
				SYSINIT4	2419Ø
LOADER	@GTMOD	19B2	138ØØ	LOADER	1291Ø
LOADER	@HDFMT	19E4	149Ø	LOADER	1284Ø
LDOS6Ø	@HEX16	Ø7BD	ØØ22Ø	LOADER	1295Ø
LDOS6Ø	@HEX8	Ø7C2	ØØ23Ø	LOADER	1295Ø 1632Ø
LDOS6Ø	@HEXDEC	Ø6F6	ØØ24Ø	LOADER	1295Ø
LOADER	@HIGH$	1948	13Ø7Ø	LOADER	1296Ø
FILPOSN	@HITRD	1897	1124Ø	FILPOSN	Ø966Ø
FILPOSN	@HITWR	1898	1125Ø	FILPOSN	Ø98ØØ
LDOS6Ø	@HZ5Ø	ØØØØ	ØØ25Ø		
$MAIN	@ICNFG	ØØ86	Ø287Ø	SYSINIT4	2415Ø
LOADER	@INIT	198D	1355Ø	LOADER	1285Ø
LDOS6Ø	@INTL	ØØØØ	ØØ26Ø	$MAIN	Ø189Ø
				SYSINIT4	214ØØ 2176Ø 2191Ø 2536Ø
LOADER	@IPL	1BF2	181ØØ	LOADER	1271Ø
LDOS6Ø	@JCL	Ø63Ø	ØØ27Ø		
LDOS6Ø	@KBD	Ø635	ØØ28Ø	LOADER	1273Ø
LDOS6Ø	@KEY	Ø628	ØØ29Ø	LOADER	1271Ø
LDOS6Ø	@KEYIN	Ø585	ØØ3ØØ	LOADER	1273Ø
				SYSINIT4	2464Ø
$MAIN	@KITSK	ØØ89	Ø292Ø		
LDOS6Ø	@KITSK	ØØ89	ØØ31Ø		
TASKER	@KLTSK	1CDØ	1981Ø	LOADER	1279Ø
LOADER	@LOAD	1B38	1677Ø	LOADER	129ØØ 1653Ø
				SYSINIT4	2393Ø
FILPOSN	@LOC	14B3	Ø387Ø	FILPOSN	Ø3Ø5Ø
				LOADER	1286Ø
FILPOSN	@LOF	14DE	Ø425Ø	LOADER	1287Ø
LDOS6Ø	@LOGER	Ø5Ø3	ØØ32Ø	LOADER	1273Ø
LDOS6Ø	@LOGOT	Ø5ØØ	ØØ33Ø	LOADER	1274Ø 1634Ø
LDOS6Ø	@MOD2	ØØØØ	ØØ34Ø		
LDOS6Ø	@MOD4	FFFF	ØØ35Ø	$MAIN	Ø249Ø
				FILPOSN	Ø417Ø
LDOS6Ø	@MSG	Ø53Ø	ØØ36Ø	LOADER	1274Ø
LDOS6Ø	@MUL16	Ø6C9	ØØ37Ø	FILPOSN	Ø318Ø
				LOADER	1293Ø
FILPOSN	@MUL8	19ØA	1211Ø	FILPOSN	Ø696Ø 1Ø5ØØ
				LOADER	1293Ø
$MAIN	@NMI	ØØ66	ØØ85Ø	SYSINIT4	2Ø34Ø
LOADER	@OPEN	198A	1353Ø	LOADER	1285Ø 1681Ø
LDOS6Ø	@OPREG	ØØ84	ØØ38Ø	TASKER	185ØØ 187ØØ
				SYSINIT4	2Ø7ØØ
				SOUND	2672Ø
LOADER	@PARAM	1987	1351Ø	LOADER	1275Ø
SOUND	@PAUSE	Ø382	2557Ø	LOADER	1275Ø
				SYSINIT4	2471Ø
FILPOSN	@PEOF	14A2	Ø377Ø	LOADER	1287Ø
FILPOSN	@POSN	1434	Ø31ØØ	LOADER	1287Ø 1774Ø

Origin	Symbolic Label	Value	Line#	Usage	Line#'s of References
LDOS60	@PRINT	0528	0039	LOADER	1274
LDOS60	@PRT	063D	0040	LOADER	1272
LDOS60	@PUT	0645	0041	LOADER	1272
LOADER	@RAMDIR	19AC	1376	LOADER	1279
LOADER	@RDHDR	19D8	1403	LOADER	1283
LOADER	@RDSEC	19F4	1417	FILPOSN	0213 1180 1192
				LOADER	1283
FILPOSN	@RDSSC	18D8	1176	FILPOSN	1111 1134 1153
				LOADER	1292
LOADER	@RDTRK	19E0	1407	LOADER	1283
FILPOSN	@READ	1513	0455	LOADER	1287
LOADER	@REMOVE	19A6	1372	LOADER	1285
LOADER	@RENAME	1996	1361	LOADER	1285
FILPOSN	@REW	149B	0370	LOADER	1288
TASKER	@RMTSK	1CD7	1986	LOADER	1278
TASKER	@RPTSK	1CEB	2003	LOADER	1278
				TASKER	1968 1982
FILPOSN	@RREAD	1473	0340	LOADER	1288
LOADER	@RSLCT	19D4	1401	LOADER	1282
$MAIN	@RST00	0000	0030		
$MAIN	@RST08	0008	0034		
$MAIN	@RST10	0010	0040		
$MAIN	@RST18	0018	0043		
$MAIN	@RST20	0020	0048		
$MAIN	@RST28	0028	0053		
$MAIN	@RST30	0030	0057	LOADER	1672
$MAIN	@RST38	0038	0059	SYSINIT4	2445
LDOS60	@RSTNMI	0FE9	0042	SYSINIT4	2033
LOADER	@RSTOR	19C8	1395	LOADER	1282
				SYSINIT4	2425
LDOS60	@RSTREG	0680	0043		
LOADER	@RUN	1B1D	1650	LOADER	1290
FILPOSN	@RWRIT	13AD	0230	LOADER	1288
LOADER	@SEEK	19D0	1399	FILPOSN	0299
				LOADER	1282
FILPOSN	@SEEKSC	1421	0295	LOADER	1288
FILPOSN	@SKIP	1430	0305	LOADER	1289
LOADER	@SLCT	19BC	1389	LOADER	1281
SOUND	@SOUND	0392	2578	LOADER	1297
				SOUND	2643
LOADER	@STEPI	19CC	1397	LOADER	1282
LDOS60	@TIME	078D	0044	LOADER	1275
LDOS60	@USA	FFFF	0045	$MAIN	0154 0340
LDOS60	@VDCTL	0B99	0046	LOADER	1274
				SYSINIT4	2274 2389 2431 2458
LDOS60	@VDCTL3	0D38	0047		
FILPOSN	@VER	1560	0508	LOADER	1289
LOADER	@VRSEC	19DC	1405	FILPOSN	0263 1025 1115 1138
				LOADER	1283
FILPOSN	@WEOF	14EC	0433	LOADER	1289
LOADER	@WHERE	1979	1338	LOADER	1272
FILPOSN	@WRITE	1531	0477	LOADER	1289
LOADER	@WRSEC	19E8	1411	FILPOSN	0259
				LOADER	1284
LOADER	@WRSSC	19EC	1413	FILPOSN	1024 1114 1137
				LOADER	1284
LOADER	@WRTRK	19F0	1415	LOADER	1284
LDOS60	@_VDCTL	0D42	0048		
TASKER	ACTVTSK	1C49	1883	TASKER	1861

Origin	Symbolic Label	Value	Line#	Usage	Line#'s of References
LDOS60	ADDR_2_ROWCOL	0DF1	00490		
FILPOSN	ADJ2	15BA	05750	FILPOSN	03480
FILPOSN	ADJUST	15B7	05730	FILPOSN	03890 05490
$MAIN	AFLAG$	006A	00990	FILPOSN	08370
FILPOSN	ALL1	167E	07220	FILPOSN	07350
FILPOSN	ALL2	1689	07300	FILPOSN	07250
FILPOSN	ALL3	1691	07360	FILPOSN	07310
FILPOSN	ALL4	16A2	07470	FILPOSN	07290
FILPOSN	ALL5	16A5	07500	FILPOSN	07460
FILPOSN	ALL6	16A6	07510	FILPOSN	07130
FILPOSN	ALLOC	1664	07050	FILPOSN	06430
FILPOSN	ATEOFW	135E	01880	FILPOSN	01850
SYSINIT4	AUTO?	1FF1	23570	SYSINIT4	23430
LDOS60	BAR$	0201	00500	SYSINIT4	20900
FILPOSN	BFRPOS	1413	02850	FILPOSN	01550 01770
FILPOSN	BKSP0	1494	03630	FILPOSN	03460
FILPOSN	BKSP1	1478	03430	FILPOSN	03600 03640
SYSINIT4	BOL	001D	20280		
SOUND	BOOTCOD	03FB	26700	LOADER	18100
				SOUND	26740
SOUND	BOOTLEN	0005	26740	LOADER	18130
LDOS60	BOOTST$	439D	00510	SYSINIT4	20980
TASKER	BREAK?	1C60	19070	TASKER	18670
TASKER	BRKVEC$	1C88	19320	LOADER	13320 13340
FILPOSN	BUMPNRN	1398	02170	FILPOSN	02140 02650
LDOS60	BUR$	0200	00520	SYSINIT4	20910
FILPOSN	BYTEIO	1300	01380		
FILPOSN	CALCDIR	18CA	11640	FILPOSN	10220 11500
FILPOSN	CALCSEC	1621	06510	FILPOSN	06140 06380
FILPOSN	CALS1	1640	06740	FILPOSN	06470
FILPOSN	CALS2	1647	06810	FILPOSN	06590
FILPOSN	CALS3	1649	06830	FILPOSN	06460
FILPOSN	CALS4	164F	06880	FILPOSN	06450 06510
FILPOSN	CALS5	165F	06980	FILPOSN	05940
LDOS60	CASHK$	0A7B	00530		
SOUND	CDLOOP	038C	25640	SOUND	25620 25670
$MAIN	CFCB$	00E0	03140		
$MAIN	CFGFCB$	00E0	03150	SYSINIT4	21380 23180 23920 24760
LDOS60	CFLAG$	006C	00540		
$MAIN	CFLAG$	006C	01120	LOADER	13100
FILPOSN	CG01	16B5	07660	FILPOSN	08170
FILPOSN	CG02	16C6	07750	FILPOSN	07950
FILPOSN	CG03	16D6	07870	FILPOSN	07850
FILPOSN	CG04	16E2	08000	FILPOSN	07920
FILPOSN	CG05	16EA	08120	FILPOSN	07770
FILPOSN	CG06	16A9	07560	FILPOSN	08140
FILPOSN	CG07	16F2	08210	FILPOSN	07560
FILPOSN	CG12	1716	08460	FILPOSN	09550
FILPOSN	CG13	173F	08780	FILPOSN	08510 08700
FILPOSN	CG14	1756	08980	FILPOSN	08430 08830
FILPOSN	CG16	175E	09010	FILPOSN	09080 09100
FILPOSN	CG17	175F	09020	FILPOSN	08990
FILPOSN	CG18	176A	09090	FILPOSN	09030
FILPOSN	CG19	1777	09200	FILPOSN	09060
FILPOSN	CG20	177D	09240	FILPOSN	09310
FILPOSN	CG21	178A	09350	FILPOSN	08740 09260
FILPOSN	CG22	1797	09460	FILPOSN	09420
FILPOSN	CG23	17A4	09570	FILPOSN	08870 09120
TASKER	CHGTASK	1CE3	19940	TASKER	20090

Origin	Symbolic Label	Value	Line#	Usage	Line#'s of References
FILPOSN	CKDBLBIT	193B	12560	FILPOSN	10510 12330
SYSINIT4	CKDCR	1FDE	23470	SYSINIT4	23380
FILPOSN	CKEOF1	1592	05470	FILPOSN	01500 01840 02020 02510 02700 02960
				FILPOSN	03360
FILPOSN	CKEOF2	15AD	05620	FILPOSN	05520 05550
FILPOSN	CKEOF3	15B4	05670	FILPOSN	05640
LOADER	CKMOD@	1A7F	15510	LOADER	15370
FILPOSN	CKOPEN@	1568	05130	FILPOSN	01400 02300 02950 03100 03400 03560
				FILPOSN	03700 03770 03870 04250 04330 04550
				FILPOSN	04770 05080 05460
SYSINIT4	CLRLOOP	1E1C	20480	SYSINIT4	20500
SYSINIT4	CONFIG$	203F	23950	SYSINIT4	23900
LDOS60	+CORE$	1CFF	00550		
$MAIN	+CORE$	1BFF	03670		
TASKER	+CORE$	1948	18230		
LOADER	+CORE$	1948	12660	$MAIN	03540
				LOADER	13030
				TASKER	18280
				$MAIN	03680 03690
$MAIN	+CORE$	0300	03370		
$MAIN	CR	000D	00130	LOADER	16460
				SYSINIT4	23580
LDOS60	CRTBGN$	F800	00560	$MAIN	03380 03510
				SYSINIT4	20360
				LOGO	26780 26840 26900 26950 27000 27050
				LOGO	27090 27130 27160 27190 27230 27270
				LOGO	27320 27370 27420 27480
SYSINIT4	CURSET	2084	24290	SYSINIT4	23650
FILPOSN	CYL_GRN	16AE	07630	FILPOSN	07050 08940
LOADER	D@FBYT8	1A26	14660		
LDOS60	DATE$	0033	00570		
$MAIN	DATE$	0033	00580	SYSINIT4	21280 21700 22040 22210 22230 22390
				SYSINIT4	22660 22820 22880 23030
SYSINIT4	DATEPR	211E	25390	SYSINIT4	21580
SYSINIT4	DATIN	1EB8	21570	SYSINIT4	21610 21660 21860 22000
SYSINIT4	DATIN1	1EC6	21620	SYSINIT4	21550
SYSINIT4	DAYLP	1F12	22300	SYSINIT4	22370
LDOS60	DAYTBL$	04C7	00580	SYSINIT4	22760
$MAIN	DBGSV$	00A0	03040		
LDOS60	DCBKL$	0031	00590		
LDOS60	DCT$	047	00600	LOADER	15020
				SYSINIT4	21010 21070 23810
LOADER	DCTBYT8@	1A29	14700		
LOADER	DCTFLD@	1A34	14920	FILPOSN	11880
				LOADER	14590 14800
FILPOSN	DEA1	192C	12410	FILPOSN	12470
FILPOSN	DEA2	1934	12470	FILPOSN	12440
SOUND	DEC1	03E4	26480	SOUND	26660
LDOS60	DFLAG$	006D	00610		
$MAIN	DFLAG$	006D	01240		
$MAIN	DIRBUF$	2300	03730	FILPOSN	08360 08680 11070
FILPOSN	DIRWR	1807	10210	FILPOSN	10190
LDOS60	DIS_DO_RAM	0846	00620		
SYSINIT4	DIV10	1F75	22900	SYSINIT4	22920
SYSINIT4	DIV7	1F37	22590	SYSINIT4	22600
FILPOSN	DNTSET	1367	01910	FILPOSN	01870
LDOS60	DODATA$	0B94	00630		
LDOS60	DODCB$	0210	00640		
LDOS60	DO_CONTROL	0C44	00650		

Origin	Symbolic Label	Value	Line#	Usage	Line#'s of References
LDOS60	DO_DSPCHAR	0CB8	0660		
LDOS60	DO_INVERT_DIS	0C8C	0670		
LDOS60	DO_INVERT_ENA	0C89	0680		
LDOS60	DO_INVERT_OFF	0C9B	0690		
LDOS60	DO_MASK	0000	0700		
LDOS60	DO_RET	0BCB	0710		
LDOS60	DO_RET1	0BCC	0720		
LDOS60	DO_SCROLL	0CCE	0730		
LDOS60	DO_TABS	0BEA	0740		
LDOS60	DSKTYP$	04C0	0750		
SYSINIT4	DSPM1	210F	2529	SYSINIT4	2532
SYSINIT4	DSPMDY	2107	2522	SYSINIT4	2285
LDOS60	DTPMT$	04C2	0760	SYSINIT4	2123
LDOS60	DVREND$	0FF4	0770		
LDOS60	DVRHI$	0206	0780		
$MAIN	EFLAG$	006E	0133		
LDOS60	ENADIS_DO_RAM	0817	0790	SYSINIT4	2344 2399
LOADER	ERRNUM	1B19	1646	LOADER	1631
LOADER	ERRSVC	1A4F	1513	LOADER	1508
LOADER	EXERR	1AD8	1600	LOADER	1586 1591
TASKER	EXITBRK	1C8E	1936	TASKER	1923
LOADER	EXOVR	1AA0	1572	LOADER	1559 1561
LOADER	EXOVR1	1AAA	1578	LOADER	1576
LOADER	EXOVR2	1AD1	1597	LOADER	1554
LOADER	EXOVR3	1ADA	1602	LOADER	1568
LOADER	EXTDBG$	19A4	1371		
LDOS60	FDDINT$	000E	0800		
$MAIN	FDDINT$	000E	0380		
$MAIN	FEMSK$	006F	0134	SYSINIT4	2092
LDOS60	FLGTAB$	006A	0810	LOADER	1328
$MAIN	FLGTAB$	006A	0940		
FILPOSN	GATRW1	188A	1114	FILPOSN	1100
FILPOSN	GATRW2	1894	1118	FILPOSN	1130
LOADER	GETADR	1BE8	1800	LOADER	1734 1741 1762 1764
LOADER	GETHI	195A	1318	LOADER	1322
LOADER	GETHILO	195E	1320	LOADER	1309
SYSINIT4	GETKB17	1FD7	2344	SYSINIT4	2347
SYSINIT4	GETKB67	204C	2397	SYSINIT4	2380
FILPOSN	GETNRN	140C	0281	FILPOSN	0231 0324 0343 0388 0547 0589
SYSINIT4	GETPARM	20AE	2455	SYSINIT4	2160 2316
LDOS60	GET_@_ROWCOL	0DAE	0820		
LOADER	GODOIO	1A1C	1456	LOADER	1442
TASKER	GOTBRK	1C6A	1916	TASKER	1907
FILPOSN	GREC1	15E7	0602	FILPOSN	0623
FILPOSN	GREC2	15F8	0615	FILPOSN	0606 0628 0636
FILPOSN	GREC3	1603	0624	FILPOSN	0612
FILPOSN	GREC4	1616	0642	FILPOSN	0618
LDOS60	HERTZ$	075	0083		
LDOS60	HIGH$	040E	0084	LOADER	1317 1318
				TASKER	1928
				SYSINIT4	2073
FILPOSN	HITRW1	18AE	1137	FILPOSN	1133
FILPOSN	HITRW2	18B8	1141	FILPOSN	1136
LOADER	HKRES$	1A6C	1537		
LDOS60	IFLAG$	0072	0085		
$MAIN	IFLAG$	0072	0147		
LDOS60	INBUF$	0420	0086	LOADER	1664
				SYSINIT4	2335
$MAIN	INTIM$	003C	0064	TASKER	1854

Origin	Symbolic Label	Value	Line#	Usage	Line#'s of References
TASKER	INTLAT	00E0	1851	TASKER	1852
$MAIN	INTMSK$	003D	00700		
LDOS60	INTVC$	003E	00870		
$MAIN	INTVC$	003E	00750		
LOADER	IOFUNC	19F6	1419	LOADER	1386 1390 1392 1394 1396 1398
				LOADER	1400 1402 1404 1406 1408 1410
				LOADER	1412 1414 1416
FILPOSN	IOREC	15CB	0589	FILPOSN	0208 0253 0297
FILPOSN	IORETZ	136A	0193	FILPOSN	0146
LDOS60	JCLCB$	0203	00880		
$MAIN	JDCB$	0024	00510	FILPOSN	0517
$MAIN	JFCB$	00C0	03080		
LDOS60	JLDCB$	0230	00890		
$MAIN	JRET$	0026	00520	FILPOSN	0516
SYSINIT4	KB1	F401	2025	SYSINIT4	2345
SYSINIT4	KB67	F460	2026	SYSINIT4	2397
SYSINIT4	KB7	F440	2027	SYSINIT4	2345
LDOS60	KCK@	07D6	00900	TASKER	1866
$MAIN	KFLAG$	0074	01690		
LDOS60	KFLAG$	0074	00910		
LDOS60	KIDATA$	08FC	00920		
LDOS60	KIDCB$	0208	00930	SYSINIT4	2109
LDOS60	LBANK$	0202	00940	TASKER	1840 1872
				SYSINIT4	2055
LOADER	LDR01	1B6F	1714	LOADER	1701 1749
LOADER	LDR02	1B72	1715	LOADER	1779
LOADER	LDR03	1B78	1718		
LOADER	LDR04	1B8A	1727	LOADER	1724
LOADER	LDR05	1B66	1707	LOADER	1726
LOADER	LDR06	1B6A	1709	LOADER	1710 1761
LOADER	LDR07	1B8E	1733	LOADER	1718
LOADER	LDR08	1B95	1739	LOADER	1716
LOADER	LDR09	1B9F	1745	LOADER	1748
LOADER	LDR12	1BA8	1751	LOADER	1720
LOADER	LDR13	1BAA	1756	LOADER	1722
LOADER	LDR14	1BB2	1760	LOADER	1697
LOADER	LDR15	1BD6	1783	LOADER	1707 1709 1714 1733 1739 1745
				LOADER	1756 1758 1766 1800 1802
LOADER	LDR16	1BD9	1785	LOADER	1794
LOADER	LDR17	1BDB	1787	LOADER	1778 1784
$MAIN	LDRV$	0023	00500	LOADER	1387 1437
$MAIN	LF	000A	00120		
$MAIN	LFLAG$	0075	01770		
FILPOSN	LNKFCB@	1566	0511		
LOADER	LOADER	1B56	1696	LOADER	1599 1683
FILPOSN	LOC1	14BC	0390	FILPOSN	0429
FILPOSN	LOC2	14C8	0401	FILPOSN	0396
FILPOSN	LOC3	14D9	0413	FILPOSN	0393
LOADER	LODERR	1BE6	1795	LOADER	1765 1777
$MAIN	LOW$	001E	00470	LOADER	1323 1324
$MAIN	LSVC$	000D	00370	LOADER	1509
$MAIN	MAXCOR$	2400	0327	TASKER	1926
				$MAIN	0373
FILPOSN	MAXCYL	18FE	1201	FILPOSN	0898
LDOS60	MAXDAY$	0401	00950	SYSINIT4	2173 2217 2229
FILPOSN	MEA1	190F	1215	FILPOSN	1219
FILPOSN	MEA2	1915	1219	FILPOSN	1217
$MAIN	MINCOR$	3000	0328		
LOADER	MOD3BUF	4300	1809	LOADER	1811

Origin	Symbolic Label	Value	Line#	Usage	Line#'s of References
				SYSINIT4	2376Ø 2382Ø
SYSINIT4	MOD4REG	2Ø98	2443Ø	SYSINIT4	2441Ø
LDOS6Ø	MODOUT$	ØØ76	ØØ96Ø	SYSINIT4	2Ø63Ø 241ØØ
$MAIN	MODOUT$	ØØ76	Ø192Ø		
LDOS6Ø	MONTBL$	Ø4DC	ØØ97Ø	SYSINIT4	2284Ø
FILPOSN	NEWHIT	17AF	Ø965Ø	FILPOSN	Ø891Ø
LDOS6Ø	NFLAG$	ØØ77	ØØ98Ø	TASKER	1838Ø 1873Ø
FILPOSN	NHIT1	17DE	Ø99ØØ	FILPOSN	Ø992Ø
FILPOSN	NHIT2	17E6	Ø995Ø	FILPOSN	Ø997Ø
FILPOSN	NHIT3	18Ø1	1Ø12Ø	FILPOSN	Ø975Ø
FILPOSN	NHIT4	181F	1Ø37Ø	FILPOSN	Ø97ØØ
FILPOSN	NHIT5	1844	1Ø6ØØ	FILPOSN	1Ø73Ø
FILPOSN	NHIT6	1845	1Ø61Ø	FILPOSN	1Ø57Ø 1Ø71Ø
FILPOSN	NHIT7	1848	1Ø63Ø	FILPOSN	1Ø54Ø
FILPOSN	NHIT8	184F	1Ø68Ø		
FILPOSN	NHIT9	1859	1Ø74Ø	FILPOSN	1Ø64Ø
SYSINIT4	NOAUT	2ØØD	2373Ø	SYSINIT4	2363Ø
SYSINIT4	NOAUT1	1FF4	2359Ø	SYSINIT4	2356Ø
SYSINIT4	NOFF	2Ø7F	2426Ø	SYSINIT4	2421Ø
TASKER	NOTASK	1CE9	2ØØ1Ø	TASKER	1825Ø 1825Ø 1825Ø 1825Ø 1826Ø 1826Ø
				TASKER	1826Ø 1826Ø 1827Ø 1827Ø 1827Ø 1986Ø
				TASKER	2Ø17Ø
SYSINIT4	NOTLEAP	1EDA	2173Ø	SYSINIT4	2169Ø
FILPOSN	NOTOPEN	1584	Ø533Ø	FILPOSN	Ø519Ø
SYSINIT4	NOTSG	2Ø69	2416Ø	SYSINIT4	24Ø9Ø
FILPOSN	NSEC1	1379	Ø2Ø4Ø	FILPOSN	Ø153Ø Ø2Ø1Ø
FILPOSN	NSEC2	1382	Ø2Ø8Ø		
TASKER	NXTMSK	1C2D	1862Ø	TASKER	19Ø3Ø
FILPOSN	NXTSECT	1375	Ø2Ø2Ø	FILPOSN	Ø463Ø
				LOADER	179ØØ
TASKER	NXTVCT	1C29	1859Ø	TASKER	1864Ø
$MAIN	OPREG$	ØØ78	Ø214Ø	TASKER	1844Ø 1869Ø
				SYSINIT4	2Ø67Ø
LDOS6Ø	OPREG$	ØØ78	ØØ99Ø		
LDOS6Ø	OPREG_SV_AREA	Ø86E	Ø1ØØØ		
LDOS6Ø	OPREG_SV_PTR	Ø835	Ø1Ø1Ø		
FILPOSN	ORARET@	14DC	Ø418Ø	LOADER	1371Ø
$MAIN	OSRLS$	ØØ3B	ØØ6ØØ		
$MAIN	OSVER$	ØØ85	Ø283Ø		
$MAIN	OVERLAY	1EØØ	Ø377Ø	LOADER	1567Ø
				SYSINIT4	2461Ø
				$MAIN	Ø387Ø
$MAIN	OVRLY$	ØØ69	ØØ89Ø	LOADER	154ØØ 1562Ø
LOADER	OVRLYOLD	1A7Ø	1539Ø	LOADER	1566Ø
LDOS6Ø	PAKNAM$	Ø41Ø	Ø1Ø2Ø	SYSINIT4	2Ø35Ø
SYSINIT4	PARSDAT	2ØCC	2476Ø		
SYSINIT4	PARTYR	2117	2534Ø	SYSINIT4	23Ø1Ø
LOADER	PAT1	1AE1	16Ø9Ø	LOADER	1596Ø
LOADER	PAT1A	1AEC	1617Ø	LOADER	16Ø9Ø
LOADER	PAT1B	1AEF	1618Ø	LOADER	1617Ø
LDOS6Ø	PAUSE@	Ø382	Ø1Ø3Ø		
LDOS6Ø	PCSAVE$	Ø7AF	Ø1Ø4Ø	TASKER	1834Ø 1925Ø
LDOS6Ø	PDRV$	ØØ1B	Ø1Ø5Ø		
$MAIN	PDRV$	ØØ1B	ØØ45Ø		
$MAIN	PHIGH$	ØØ1C	ØØ46Ø	SYSINIT4	2Ø74Ø
LOADER	POPERR	1BØE	164ØØ	LOADER	166ØØ
TASKER	POPREGS	1C58	1898Ø	TASKER	1888Ø
FILPOSN	POSN1	145E	Ø328Ø	FILPOSN	Ø313Ø Ø317Ø Ø373Ø Ø381Ø Ø383Ø
FILPOSN	POSN2	1461	Ø329Ø	FILPOSN	Ø323Ø

Origin	Symbolic Label	Value	Line#	Usage	Line#'s of References
FILPOSN	POSN2A	1462	03300	FILPOSN	03520
LDOS60	PRDCB$	0218	01060		
SYSINIT4	PRSD1	20D1	24780	SYSINIT4	25060 25090
SYSINIT4	PRSD2	20E6	24960	SYSINIT4	24830 24910
SYSINIT4	PRSD3	20F8	25100	SYSINIT4	24820 24900
SYSINIT4	PRSD4	20FC	25130	SYSINIT4	25080
LDOS60	PUTA@DE	0DCD	01070		
LOADER	PUTLO	1965	13240	LOADER	13160
LDOS60	PUT_@	0DCA	01080		
LDOS60	PUT_@_ROWCOL	0DC6	01090		
FILPOSN	RDCHAR	1312	01500	FILPOSN	04660
FILPOSN	RDREC	1524	04640	FILPOSN	04720
FILPOSN	READIR	18F1	11920	FILPOSN	11760
FILPOSN	RELCYL	1919	12250	FILPOSN	06890 08590
FILPOSN	RESTREG	1589	05380	FILPOSN	05270
TASKER	RETINST	1C48	18780	$MAIN	00750 00760 00760 00770 00770 00770
				$MAIN	00770
$MAIN	RFLAG$	007B	02250		
LDOS60	RFLAG$	007B	01100		
LDOS60	ROWCOL_2_ADDR	0DD0	01110		
LOADER	RST28	1A5B	15270	$MAIN	00530
TASKER	RST38@	1BFF	18320	$MAIN	00590
LDOS60	RSTOR$	04C4	01120	SYSINIT4	24230
TASKER	RTCPROC	1C94	19450	$MAIN	00760
TASKER	RTCTASK	1CBB	19640	TASKER	19480 19550 19570 19590
FILPOSN	RWRIT1	13B0	02310	FILPOSN	01920 02280
FILPOSN	RWRIT2	13BD	02410	FILPOSN	04820
FILPOSN	RWRIT3	13C6	02450	FILPOSN	01680 02070 02520
FILPOSN	RWRIT4	13CA	02490	FILPOSN	02440
FILPOSN	RWRIT5	13D3	02530	FILPOSN	02500
FILPOSN	RWRIT@	13A2	02250	FILPOSN	03300 04340 04570
LDOS60	S1DCB$	0238	01130	SYSINIT4	20580
FILPOSN	SBIT1	1871	10960	FILPOSN	10940
$MAIN	SBUFF$	1D00	03710	$MAIN	02980 03090
				FILPOSN	11310 11680
				LOADER	16800 17000
SYSINIT4	SELDCT	1FC7	23350	SYSINIT4	23120
FILPOSN	SET5	132A	01620	FILPOSN	01580 01830 03350
LOADER	SET@EXEC	1A79	15440	LOADER	15330
FILPOSN	SETBIT	1868	10900	FILPOSN	09360
SYSINIT4	SETCYL0	206B	24180	SYSINIT4	24270
LDOS60	SET_SCROLL	0CF3	01140		
$MAIN	SFCB$	008C	02970	LOADER	15780 15810 15830 15950 15980
$MAIN	SFLAG$	007C	02370		
LDOS60	SFLAG$	007C	01150	LOADER	16510 16680 16780
				TASKER	19160
				SYSINIT4	24060
				SOUND	25600 25970
LDOS60	SIDCB$	0220	01160		
SOUND	SNDLEN	004F	26430		
SOUND	SNDOFF	00B4	26250	SOUND	26280 26300 26320 26340 26360 26380
				SOUND	26400 26420
SOUND	SNDPORT	0090	25510	SOUND	26070 26110
SOUND	SNDTAB	03D1	26270	SOUND	25830
LDOS60	SODCB$	0228	01170		
SYSINIT4	SPACE4	20FF	25180	SYSINIT4	22770
SYSINIT4	SPACE4$	2142	25430	SYSINIT4	25190
LDOS60	STACK$	0380	01180	LOADER	16350
				SYSINIT4	20470 20530

Origin	Symbolic Label	Value	Line#	Usage	Line#'s of References
				SOUND	25520
LDOS60	START$	0000 01190		$MAIN	00260 03270 03280 03640 03700
				SYSINIT4	20300
$MAIN	START$	0000 00250			
SYSINIT4	STUB	2056 24060		SYSINIT4	23750 24510
SYSINIT4	STUB1	2059 24070		SYSINIT4	23410
SYSINIT4	STUBEND	20AE 24500		SYSINIT4	24510
SYSINIT4	STUBLEN	0058 24510		SYSINIT4	23770
FILPOSN	STUFDEC	17AB 09600		FILPOSN	07670 09860 10030
LOADER	SVCERR	1AF4 16280		LOADER	12760 12800 12800 12800 12800 12920
				LOADER	12940 12940 12970 12980 12980 12980
				LOADER	12980 12990 12990 12990 12990 13000
				LOADER	13000 13000 13000 13010 13010 13010
				LOADER	13010 13020 13020 13020 13020
$MAIN	SVCRET$	000B 00360		LOADER	15110
$MAIN	SVCTAB$	0100 03200		$MAIN	02790
				LOADER	12670 15150
LOADER	SVCUSER	1A43 15070		LOADER	15280
LOADER	SYSERR	1AF7 16300		LOADER	16050
LOADER	SYSERR$	1B13 16450		LOADER	16330
LOADER	TAPDRV	19B8 13870		TASKER	19100
FILPOSN	TBIT1	1865 10850		FILPOSN	10840
$MAIN	TCB$	004E 00810		TASKER	18240 19650 19670 19830 19910 19930
				TASKER	20140 20160
$MAIN	TFLAG$	007D 02500		SYSINIT4	24430
$MAIN	TIME$	002D 00560			
LDOS60	TIME$	002D 01200		SYSINIT4	23290
SYSINIT4	TIMEPR	2130 25420		SYSINIT4	23140
LDOS60	TIMER$	002C 01210		TASKER	19600
$MAIN	TIMER$	002C 00550			
SYSINIT4	TIMIN	1F99 23100		SYSINIT4	21370
SYSINIT4	TIMIN0	1F9F 23130		SYSINIT4	23170 23210 23250 23280
LDOS60	TIMSL$	002B 01220		TASKER	19490
$MAIN	TIMSL$	002B 00540			
LDOS60	TIMTSK$	0713 01230		TASKER	19520
LDOS60	TMPMT$	04C3 01240		SYSINIT4	23100
SOUND	TONER	001C 26260		SOUND	26270 26290 26310 26330 26350 26370
				SOUND	26390 26410
LDOS60	TRACE_INT	07B1 01250			
LOADER	TRANSFR	1A74 15410		LOADER	16020
SYSINIT4	TRKREG	00F1 20240		SYSINIT4	21060
TASKER	TSKEXIT	1C36 18680		TASKER	19120 19180 19270 19300 19350
FILPOSN	TSTBIT	185B 10790		FILPOSN	08730 09250
TASKER	TSTBRK	1C31 18660		TASKER	18580
LDOS60	TYPHK$	0A8F 01260			
LDOS60	TYPTSK$	0B26 01270		TASKER	18270
$MAIN	USTOR$	0013 00420			
FILPOSN	VEROP	13E9 02610		FILPOSN	04780
$MAIN	VFLAG$	007F 02630			
LDOS60	VFLAG$	007F 01280			
FILPOSN	WEOF1	14F2 04350		FILPOSN	02780
FILPOSN	WRCH1	1343 01770		FILPOSN	01720
FILPOSN	WRCH2	136C 01980		FILPOSN	01710
FILPOSN	WRCHAR	132F 01670		FILPOSN	01450 04920
FILPOSN	WRERROR	1557 05000		FILPOSN	04950
$MAIN	WRINT$	0080 02740		SYSINIT4	20650
FILPOSN	WRIT1	1534 04780		FILPOSN	05100
FILPOSN	WRREC	1547 04880		FILPOSN	04960
FILPOSN	YESEOF	13FE 02750			

```
Origin  Symbolic Label Value Line# Usage    Line#'s of References

SYSINIT4 ZERDCB         1E2E 2Ø59Ø SYSINIT4 2Ø61Ø
LDOS6Ø   ZERO$          Ø4Ø1 Ø129Ø SYSINIT4 2116Ø
FILPOSN  ZEROA@         13AØ Ø22ØØ FILPOSN  Ø218Ø Ø229Ø Ø274Ø

ØØ51Ø Symbols declared  -  ØØ761 References
```

NOTES:

SYS1 is, among other things, the primary command interpreter. As such, it handles all requests for commands in the three system Libraries. It also contains the code for the SVCs @CMNDI, @CMNDR, @FSPEC, @FEXT, @PARAM, @EXIT and @ABORT. SYS1 is normally executed by doing an @EXIT or @ABORT SVC from within a program, or by executing a RET instruction as long as the stack pointer is at the same position it was in when the program was executed by the DOS.

```
                00100 ;SYS1/ASM - LS-DOS 6.2
0000            00110         TITLE   <SYS1 - LS-DOS 6.2>
                00120 ;
003A            00130 LD_A    EQU     3AH             ;LD A,(nnnn)
                00140 ;
0000            00150 @SMALL  EQU     0               ;Switch for "SMALL" or
                00160                                 ;  "FULL" library
                00170 ;
8000            00180 LIBA    EQU     08000H
A000            00190 LIBB    EQU     0A000H          ;Set bit 5
C000            00200 LIBC    EQU     0C000H          ;Set bit 6
000A            00210 LF      EQU     10
000D            00220 CR      EQU     13
                00230 *LIST   OFF                     ;Get SYS0/EQU
                00250 *LIST   ON
0000            00260 *GET    COPYCOM:3               ;Copyright message
                03010 ; COPYCOM - File for Copyright COMment block
                03020 ;
0000            03030         COM     '<*(C) 1982,83,84 by LSI*>'
                03040 ;
                00270 ;
1E00            00280         ORG     1E00H
                00290 ;
1E00 1802       00300 SYS1    JR      SYS1BGN         ;Hop around pointer
1E02 241F       00310         DW      LIBTBL$         ;LIBTBL pointer
1E04 E670       00320 SYS1BGN AND     70H             ;Strip all but ept
1E06 C8         00330         RET     Z               ;Back on zero entry
1E07 FE10       00340         CP      10H             ;Ck for @EXIT
1E09 2836       00350         JR      Z,CMD
1E0B FE40       00360         CP      40H             ;Ck for FSPEC
1E0D CA5D20     00370         JP      Z,FSPEC
1E10 FE50       00380         CP      50H             ;Ck for FEXT
1E12 CAD120     00390         JP      Z,FEXT
1E15 FE60       00400         CP      60H             ;Ck for PARAM
1E17 CAD821     00410         JP      Z,PARAM
1E1A FE70       00420         CP      70H             ;Ck for vacant entry
1E1C C8         00430         RET     Z
                00440 ;
                00450 ;       Entry code for CMNDI (30) and CMNDR (20) SVCs
                00460 ;
1E1D 112004     00470         LD      DE,INBUF$       ;Move 79 characters
1E20 D5         00480         PUSH    DE              ;  from (HL) to buffer
1E21 014F00     00490         LD      BC,79
1E24 EDB0       00500         LDIR
1E26 EB         00510         EX      DE,HL           ;Terminate with ETX
1E27 3603       00520         LD      (HL),3
1E29 E1         00530         POP     HL              ;Recover buffer start
1E2A FE30       00540         CP      30H             ;Ck entry for CMNDI
1E2C 280E       00550         JR      Z,CMD30         ;Go on CMNDI
1E2E CD5305     00560         CALL    @CKBRKC         ;Clear the Break bit
1E31 3A6C00     00570         LD      A,(CFLAG$)
1E34 F602       00580         OR      2               ;Set CMNDR bit
1E36 326C00     00590         LD      (CFLAG$),A      ;Put it back
1E39 C3BC1E     00600         JP      CMD20           ;  & go on CMNDR
                00610 ;
                00620 ;       Entry for @EXIT & @CMNDI
                00630 ;
1E3C CD461E     00640 CMD30   CALL    CLEANUP         ;Reset Break, stack, etc.
1E3F 1871       00650         JR      CMD3A
                00660 ;
1E41 CD461E     00670 CMD     CALL    CLEANUP         ;Reset Break, stack, etc.
```

```
1E44 1848      00680          JR      CMDCONT
               00690 ;
               00700 CLEANUP
1E46 F3        00710          DI                      ;Stop for a moment
1E47 210000    00720          LD      HL,0            ;Reset vectored BREAK
1E4A CD6F19    00730          CALL    @BREAK          ;  to system
1E4D E1        00740          POP     HL              ;P/u local RETurn
1E4E 31803     00750          LD      SP,STACK$       ;Reset stack pointer
1E51 010B1B    00760          LD      BC,@EXIT        ;Establish ret address
1E54 C5        00770          PUSH    BC
1E55 E5        00780          PUSH    HL              ;Put back local return
1E56 3A7C00    00790          LD      A,(SFLAG$)      ;DEBUG to be on or off?
1E59 07        00800          RLCA
1E5A 3EC9      00810          LD      A,0C9H          ;Bit 7, 1=on, 0=off
1E5C 3001      00820          JR      NC,DBGOFF       ;Go if OFF
1E5E AF        00830          XOR     A               ;  else reset to on
1E5F 329F19    00840 DBGOFF   LD      (@DBGHK),A
1E62 217400    00850          LD      HL,KFLAG$       ;Point to KFLAG$
1E65 3EF9      00860          LD      A,11111001B     ;Reset pause and enter
1E67 A6        00870          AND     (HL)            ;Merge together
1E68 77        00880          LD      (HL),A
1E69 217C00    00890          LD      HL,SFLAG$       ;Point to SFLAG
1E6C 3EF8      00900          LD      A,11111000B     ;Reset 3 lo bits
1E6E A6        00910          AND     (HL)            ;Merge with old
1E6F 77        00920          LD      (HL),A
1E70 21FF2F    00930          LD      HL,2FFFH        ;Reset LOW$
1E73 221E00    00940          LD      (LOW$),HL
               00950 ;
               00960 ;        Reset video ram handler pointer
               00970 ;
1E76 216E08    00980          LD      HL,OPREG_SV_AREA
1E79 223508    00990          LD      (OPREG_SV_PTR),HL
1E7C 3A6C00    01000          LD      A,(CFLAG$)      ;P/u CFLAG
1E7F E620      01010          AND     20H             ;Leave only bit 5
1E81 326C00    01020          LD      (CFLAG$),A      ;  and put it back
1E84 212004    01030          LD      HL,INBUF$       ;Point to command line
1E87 E5        01040          PUSH    HL              ;Xfer start
1E88 C1        01050          POP     BC              ;  to BC
1E89 FB        01060          EI
1E8A CD5305    01070          CALL    @CKBRKC         ;Check and clear BREAK
1E8D C9        01080          RET                     ;Local cleanup done
               01090 ;
1E8E 3A6E00    01100 CMDCONT  LD      A,(EFLAG$)      ;P/u ECI flag
1E91 B7        01110          OR      A               ;Check if set
1E92 2803      01120          JR      Z,CMD1A         ;Go if normal
1E94 F68F      01130          OR      10001111B       ;Set for SYS13 but
               01140                                  ;  leave user entry code
1E96 EF        01150          RST     40
               01160 ;
1E97 21EA22    01170 CMD1A    LD      HL,RDYMSG$      ;Display ready message
1E9A CD2D05    01180          CALL    @DSPLY
1E9D 216C00    01190 CMD2     LD      HL,CFLAG$       ;Let the world know we
1EA0 CBD6      01200          SET     2,(HL)          ;  are in the command
1EA2 E5        01210          PUSH    HL              ;  interpreter
1EA3 212004    01220          LD      HL,INBUF$       ;Get 79 chars max
1EA6 01004F    01230          LD      BC,79<8         ;No fill char for now
1EA9 CD8505    01240          CALL    @KEYIN
1EAC E3        01250          EX      (SP),HL         ;Turn off the interpreter
1EAD CB96      01260          RES     2,(HL)          ;  bit & reget the buffer
1EAF E1        01270          POP     HL
1EB0 388F      01280          JR      C,CMD           ;Jump on <BREAK>
```

```
                    Ø129Ø ;
                    Ø13ØØ ;       Entry from @EXIT & @CMNDI
                    Ø131Ø ;
                    Ø132Ø CMD3A
1EB2 7E             Ø133Ø         LD      A,(HL)          ;Check for comment
1EB3 FE2E           Ø134Ø         CP      '.'             ;If so go before CR
1EB5 28Ø5           Ø135Ø         JR      Z,CMD2Ø         ;  is displayed
                    Ø136Ø ;
1EB7 3EØD           Ø137Ø         LD      A,CR            ;Do a line feed on
1EB9 CD42Ø6         Ø138Ø         CALL    @DSP            ;  CMNDI and @EXIT
                    Ø139Ø ;
                    Ø14ØØ ;       Entry from @CMNDR plus the above
                    Ø141Ø ;
                    Ø142Ø ;       Always bring in bank Ø
                    Ø143Ø ;
1EBC AF             Ø144Ø CMD2Ø   XOR     A               ;Prepare for bank-Ø
1EBD 47             Ø145Ø         LD      B,A             ;Set function and
1EBE 4F             Ø146Ø         LD      C,A             ;  bank number to Ø
1EBF CD77Ø8         Ø147Ø         CALL    @BANK           ;Invoke bank Ø
                    Ø148Ø ;
                    Ø149Ø ;       Process the command entry
                    Ø15ØØ ;
1EC2 CDØ3Ø5         Ø151Ø         CALL    @LOGER          ;Log the entry
1EC5 11EØØØ         Ø152Ø         LD      DE,CFCB$        ;Point to command FCB
1EC8 7E             Ø153Ø         LD      A,(HL)          ;Jump on comment
1EC9 FE2E           Ø154Ø         CP      '.'
1ECB 2838           Ø155Ø         JR      Z,COMMENT
1ECD FE2A           Ø156Ø         CP      '*'             ;Check if alternate CMD
1ECF 2ØØ6           Ø157Ø         JR      NZ,CKNOEXC      ;  processor needed
1ED1 E5             Ø158Ø         PUSH    HL
1ED2 C1             Ø159Ø         POP     BC              ;Get Buffer in BC
1ED3 23             Ø16ØØ         INC     HL              ;Move HL past '*'
1ED4 3EFF           Ø161Ø         LD      A,ØFFH          ;Set up for SYS13 entry
1ED6 EF             Ø162Ø         RST     4Ø              ;  # 7, and do it
1ED7 D621           Ø163Ø CKNOEXC SUB     '!'             ;Test for program force
1ED9 2ØØ1           Ø164Ø         JR      NZ,NOEXC
1EDB 23             Ø165Ø         INC     HL              ;Bump past the '!'
1EDC 32E61E         Ø166Ø NOEXC   LD      (TSTEXC+1),A
1EDF CD5D2Ø         Ø167Ø         CALL    FSPEC           ;Fetch command spec
1EE2 2Ø1D           Ø168Ø         JR      NZ,WHAT         ;Jump on error
1EE4 E5             Ø169Ø         PUSH    HL              ;Save terminator pointer
1EE5 3EØØ           Ø17ØØ TSTEXC  LD      A,Ø             ;Test if prog force
1EE7 B7             Ø171Ø         OR      A
1EE8 28Ø8           Ø172Ø         JR      Z,NOTLIB        ;Jump if starting "!"
1EEA Ø1241F         Ø173Ø         LD      BC,LIBTBL$      ;Pt to tbl of LIB cmds
1EED CD5821         Ø174Ø         CALL    @FNDPRM         ;Check for a match
1EFØ 281F           Ø175Ø         JR      Z,CMD4          ;Jump if it is
1EF2 21E722         Ø176Ø NOTLIB  LD      HL,DFTEXT       ;Else assume prg file, so
1EF5 CDD12Ø         Ø177Ø         CALL    FEXT            ;  default 'EXT' to CMD
1EF8 E1             Ø178Ø         POP     HL              ;Rcvr terminator pointer
1EF9 3A6CØØ         Ø179Ø         LD      A,(CFLAG$)      ;Ck LIB only execution
1EFC E61Ø           Ø18ØØ         AND     1ØH             ;CFLAG$ bit-4
1EFE CA1D1B         Ø181Ø         JP      Z,@RUN          ;The program else WHAT?
                    Ø182Ø ;
                    Ø183Ø ;       Process non-entry
                    Ø184Ø ;
1FØ1 21FFFF         Ø185Ø WHAT    LD      HL,-1           ;Set to show abort
1FØ4 C9             Ø186Ø         RET
                    Ø187Ø ;
                    Ø188Ø ;       Process "dot" comment
                    Ø189Ø ;
```

```
1FØ5 3A7CØØ   Ø19ØØ COMMENT LD      A,(SFLAG$)      ;Ret if <DO> in effect
1FØ8 CB6F     Ø191Ø         BIT     5,A             ;  else get another
1FØA CA9D1E   Ø192Ø         JP      Z,CMD2          ;  input line
1FØD 21ØØØØ   Ø193Ø         LD      HL,Ø            ;Set for no error
1F1Ø C9       Ø194Ø         RET
              Ø195Ø ;
              Ø196Ø ;       Process LIB command
              Ø197Ø ;
1F11 E1       Ø198Ø CMD4    POP     HL              ;Rcvr terminator pointer
1F12 3EC9     Ø199Ø         LD      A,ØC9H          ;Turn off DEBUG
1F14 329F19   Ø2ØØØ         LD      (@DBGHK),A
1F17 7A       Ø2Ø1Ø         LD      A,D             ;Test bit 7 of high
1F18 Ø7       Ø2Ø2Ø         RLCA                    ;  order LIB address
1F19 D5       Ø2Ø3Ø         PUSH    DE              ;Ret to address of
1F1A DØ       Ø2Ø4Ø         RET     NC              ;  vector if bit 7 = Ø
1F1B D1       Ø2Ø5Ø         POP     DE
1F1C 43       Ø2Ø6Ø         LD      B,E             ;Else put overlay # in
1F1D Ø7       Ø2Ø7Ø         RLCA                    ;Calculate needed library
1F1E Ø7       Ø2Ø8Ø         RLCA                    ;  by rotating 7-5 into
1F1F C684     Ø2Ø9Ø         ADD     A,84H           ;  2-Ø & adding RST base
1F21 EF       Ø21ØØ         RST     28H
              Ø211Ø ;
              Ø212Ø ;       BOOT code brings back the ROM
              Ø213Ø ;
1F22 AF       Ø214Ø BOOTIT  XOR     A               ;SVC-Ø => @IPL
1F23 EF       Ø215Ø         RST     4Ø
              Ø216Ø ;
              Ø217Ø ;       LIBRARY look-up table starts here
              Ø218Ø ;
1F24          Ø219Ø LIBTBL$ EQU     $               ;Start of library table
              Ø22ØØ ;
              Ø221Ø         IF      @SMALL
              Ø222Ø ;
              Ø223Ø ;       Use this table for SMALL (OEM) library
              Ø224Ø ;
              Ø225Ø ; DB 'APPEND'
              Ø226Ø ; DW LIBA!31H
              Ø227Ø         DB      'ATTRIB'
              Ø228Ø         DW      LIBB!51H
              Ø229Ø         DB      'AUTO  '
              Ø23ØØ         DW      LIBB!11H
              Ø231Ø ;DB 'BOOT  '
              Ø232Ø ; DW BOOTIT
              Ø233Ø ; DB 'BUILD '
              Ø234Ø ; DW LIBB!33H
              Ø235Ø ; DB 'CAT   '
              Ø236Ø ; DW LIBA!2ØH
              Ø237Ø ; DB 'CLS   '
              Ø238Ø ; DW LIBA!24H
              Ø239Ø         DB      'COPY  '
              Ø24ØØ         DW      LIBA!32H
              Ø241Ø ; DB 'CREATE'
              Ø242Ø ; DW LIBB!13H
              Ø243Ø         DB      'DATE  '
              Ø244Ø         DW      LIBB!15H
              Ø245Ø ; DB 'DEBUG '
              Ø246Ø ; DW LIBB!14H
              Ø247Ø ; DB 'DEVICE'
              Ø248Ø ; DW LIBA!61H
              Ø249Ø         DB      'DIR   '
              Ø25ØØ         DW      LIBA!21H
```

```
                02510           DB      'DO    '
                02520           DW      LIBA!91H
                02530 ; DB 'DUMP  '
                02540 ; DW LIBB!71H
                02550           DB      'FILTER'
                02560           DW      LIBA!66H
                02570           DB      'FORMS '
                02580           DW      LIBC!0B1H
                02590 ; DB 'FREE  '
                02600 ; DW LIBB!22H
                02610 ; DB 'LIB   '
                02620 ; DW LIBA!19H
                02630 ; DB 'LINK  '
                02640 ; DW LIBA!62H
                02650 ; DB 'LIST  '
                02660 ; DW LIBA!41H
                02670 ; DB 'LOAD  '
                02680 ; DW LIBA!81H
                02690 ; DB 'MEMORY'
                02700 ; DW LIBA!1EH
                02710 ; DB 'PURGE '
                02720 ; DW LIBB!72H
                02730           DB      'REMOVE'
                02740           DW      LIBA!18H
                02750 ; DB 'RENAME'
                02760 ; DW LIBA!53H
                02770 ; DB 'RESET '
                02780 ; DW LIBA!63H
                02790 ; DB 'ROUTE '
                02800 ; DW LIBA!64H
                02810 ; DB 'RUN   '
                02820 ; DW LIBA!82H
                02830           DB      'SET   '
                02840           DW      LIBA!65H
                02850 ; DB 'SETCOM'
                02860 ; DW LIBC!0B2H
                02870 ; DB 'SETKI '
                02880 ; DW LIBC!0B3H
                02890 ; DB 'SPOOL '
                02900 ; DW LIBC!0A2H
                02910           DB      'SYSGEN'
                02920           DW      LIBC!1CH
                02930           DB      'SYSTEM'
                02940           DW      LIBC!0A1H
                02950           DB      'TIME  '
                02960           DW      LIBB!16H
                02970 ; DB 'TOF   '
                02980 ; DW LIBA!25H
                02990           DB      'VERIFY'
                03000           DW      LIBB!1BH
                03010           NOP                     ;Patch 'K' here for KILL
                03020           DB      'ILL  '
                03030           DW      LIBA!18H
                03040           NOP
                03050 ;
                03060 ;
                03070           ELSE
                03080 ;
                03090 ; This table for FULL library
                03100 ;
1F24 41         03110           DB      'APPEND'
```

```
     50 50 45 4E 44
1F2A 3180      03120          DW      LIBA!31H
1F2C 41        03130          DB      'ATTRIB'
     54 54 52 49 42
1F32 51A0      03140          DW      LIBB!51H
1F34 41        03150          DB      'AUTO  '
     55 54 4F 20 20
1F3A 11A0      03160          DW      LIBB!11H
1F3C 42        03170          DB      'BOOT  '
     4F 4F 54 20 20
1F42 221F      03180          DW      BOOTIT
1F44 42        03190          DB      'BUILD '
     55 49 4C 44 20
1F4A 33A0      03200          DW      LIBB!33H
1F4C 43        03210          DB      'CAT   '
     41 54 20 20 20
1F52 2080      03220          DW      LIBA!20H
1F54 43        03230          DB      'CLS   '
     4C 53 20 20 20
1F5A 2480      03240          DW      LIBA!24H
1F5C 43        03250          DB      'COPY  '
     4F 50 59 20 20
1F62 3280      03260          DW      LIBA!32H
1F64 43        03270          DB      'CREATE'
     52 45 41 54 45
1F6A 13A0      03280          DW      LIBB!13H
1F6C 44        03290          DB      'DATE  '
     41 54 45 20 20
1F72 15A0      03300          DW      LIBB!15H
1F74 44        03310          DB      'DEBUG '
     45 42 55 47 20
1F7A 14A0      03320          DW      LIBB!14H
1F7C 44        03330          DB      'DEVICE'
     45 56 49 43 45
1F82 6180      03340          DW      LIBA!61H
1F84 44        03350          DB      'DIR   '
     49 52 20 20 20
1F8A 2180      03360          DW      LIBA!21H
1F8C 44        03370          DB      'DO    '
     4F 20 20 20 20
1F92 9180      03380          DW      LIBA!91H
1F94 44        03390          DB      'DUMP  '
     55 4D 50 20 20
1F9A 71A0      03400          DW      LIBB!71H
1F9C 46        03410          DB      'FILTER'
     49 4C 54 45 52
1FA2 6680      03420          DW      LIBA!66H
1FA4 46        03430          DB      'FORMS '
     4F 52 4D 53 20
1FAA B1C0      03440          DW      LIBC!0B1H
1FAC 46        03450          DB      'FREE  '
     52 45 45 20 20
1FB2 22A0      03460          DW      LIBB!22H
1FB4 4C        03470          DB      'LIB   '
     49 42 20 20 20
1FBA 1980      03480          DW      LIBA!19H
1FBC 4C        03490          DB      'LINK  '
     49 4E 4B 20 20
1FC2 6280      03500          DW      LIBA!62H
1FC4 4C        03510          DB      'LIST  '
     49 53 54 20 20
```

```
1FCA 4180      03520          DW      LIBA!41H
1FCC 4C        03530          DB      'LOAD  '
     4F 41 44 20 20
1FD2 8180      03540          DW      LIBA!81H
1FD4 4D        03550          DB      'MEMORY'
     45 4D 4F 52 59
1FDA 1E80      03560          DW      LIBA!1EH
1FDC 50        03570          DB      'PURGE '
     55 52 47 45 20
1FE2 72A0      03580          DW      LIBB!72H
1FE4 52        03590          DB      'REMOVE'
     45 4D 4F 56 45
1FEA 1880      03600          DW      LIBA!18H
1FEC 52        03610          DB      'RENAME'
     45 4E 41 4D 45
1FF2 5380      03620          DW      LIBA!53H
1FF4 52        03630          DB      'RESET '
     45 53 45 54 20
1FFA 6380      03640          DW      LIBA!63H
1FFC 52        03650          DB      'ROUTE '
     4F 55 54 45 20
2002 6480      03660          DW      LIBA!64H
2004 52        03670          DB      'RUN   '
     55 4E 20 20 20
200A 8280      03680          DW      LIBA!82H
200C 53        03690          DB      'SET   '
     45 54 20 20 20
2012 6580      03700          DW      LIBA!65H
2014 53        03710          DB      'SETCOM'
     45 54 43 4F 4D
201A B2C0      03720          DW      LIBC!0B2H
201C 53        03730          DB      'SETKI '
     45 54 4B 49 20
2022 B3C0      03740          DW      LIBC!0B3H
2024 53        03750          DB      'SPOOL '
     50 4F 4F 4C 20
202A A2C0      03760          DW      LIBC!0A2H
202C 53        03770          DB      'SYSGEN'
     59 53 47 45 4E
2032 1CC0      03780          DW      LIBC!1CH
2034 53        03790          DB      'SYSTEM'
     59 53 54 45 4D
203A A1C0      03800          DW      LIBC!0A1H
203C 54        03810          DB      'TIME  '
     49 4D 45 20 20
2042 16A0      03820          DW      LIBB!16H
2044 54        03830          DB      'TOF   '
     4F 46 20 20 20
204A 2580      03840          DW      LIBA!25H
204C 56        03850          DB      'VERIFY'
     45 52 49 46 59
2052 1BA0      03860          DW      LIBB!1BH
2054 00        03870          NOP                     ;Patch 'K' here for KILL
2055 49        03880          DB      'ILL   '
     4C 4C 20 20
205A 1880      03890          DW      LIBA!18H
205C 00        03900          NOP
               03910 ;
               03920          ENDIF
               03930 ;
               03940 ;
```

```
                    Ø3950 ;       Routine to fetch a filespec/devicespec
                    Ø3960 ;
205D D5             Ø3970 FSPEC   PUSH    DE              ;Save pointer to DCB
205E CDØ521         Ø3980         CALL    @PARSER         ;Parse expected command
2061 203E           Ø3990         JR      NZ,FSP5         ;NZ=not file, ck for device
2063 FE2F           Ø4000         CP      '/'             ;EXT separator?
2065 2007           Ø4010         JR      NZ,FSP1
2067 12             Ø4020         LD      (DE),A          ;File extent coming,
2068 13             Ø4030         INC     DE              ;  get it
2069 Ø6Ø3           Ø4040         LD      B,3             ;EXT is 3-chars max
206B CDØ721         Ø4050         CALL    @PAR1
206E FE2E           Ø4060 FSP1    CP      '.'             ;Password entered?
2070 2007           Ø4070         JR      NZ,FSP2
2072 12             Ø4080         LD      (DE),A          ;Password coming,
2073 13             Ø4090         INC     DE              ;  get it also
2074 CDØ521         Ø4100         CALL    @PARSER
2077 2Ø35           Ø4110         JR      NZ,FSP6         ;Return if error
2079 FE3A           Ø4120 FSP2    CP      ':'             ;Drive entered?
207B 2009           Ø4130         JR      NZ,FSP3
207D 12             Ø4140         LD      (DE),A          ;A one-byte drive
207E 13             Ø4150         INC     DE              ;  has been had
207F Ø6Ø1           Ø4160         LD      B,1
2081 CDØ721         Ø4170         CALL    @PAR1
2084 2Ø28           Ø4180         JR      NZ,FSP6         ;Return if error
2086 FE21           Ø4190 FSP3    CP      '!'             ;Update EOF always?
2088 2004           Ø4200         JR      NZ,FSP4
208A 12             Ø4210         LD      (DE),A          ;Yes, slow but accurate
208B 13             Ø4220         INC     DE              ;Inc buffer pointers
208C 23             Ø4230         INC     HL
208D 7E             Ø4240         LD      A,(HL)
208E 4F             Ø4250 FSP4    LD      C,A             ;Save separator char
208F 3EØ3           Ø4260         LD      A,3
2091 12             Ø4270         LD      (DE),A          ;Stuff an ETX
2092 AF             Ø4280         XOR     A
2093 79             Ø4290         LD      A,C             ;P/u separator
2094 D1             Ø4300         POP     DE              ;P/u start of DCB
2095 D5             Ø4310         PUSH    DE
2096 Ø1BØ2Ø         Ø4320         LD      BC,PREPTBL      ;Ck on prepositions
2099 CD5821         Ø4330         CALL    @FNDPRM
209C D1             Ø4340         POP     DE              ;Can use TO, ON,
209D 28BE           Ø4350         JR      Z,FSPEC         ;  OVER, USING
209F AF             Ø4360         XOR     A
20AØ C9             Ø4370         RET
20A1 FE2A           Ø4380 FSP5    CP      '*'             ;Ck on device spec
20A3 2009           Ø4390         JR      NZ,FSP6         ;Jump if not device
20A5 12             Ø4400         LD      (DE),A          ;  else stuff the '*'
20A6 13             Ø4410         INC     DE
20A7 Ø6Ø2           Ø4420         LD      B,2             ;Xfer two char device
20A9 CDØ721         Ø4430         CALL    @PAR1
20AC 28EØ           Ø4440         JR      Z,FSP4          ;Terminate buffer
20AE D1             Ø4450 FSP6    POP     DE
20AF C9             Ø4460         RET
                    Ø4470 ;
                    Ø4480 ;       Preposition table
                    Ø4490 ;
20BØ 54             Ø4500 PREPTBL DB      'TO    '
    4F 2Ø 2Ø 2Ø 2Ø
20B6 ØØ1D           Ø4510         DW      SBUFF$
20B8 4F             Ø4520         DB      'ON    '
    4E 2Ø 2Ø 2Ø 2Ø
20BE ØØ1D           Ø4530         DW      SBUFF$
```

```
20C0 4F          0454Ø          DB      'OVER  '
     56 45 52 2Ø 2Ø
20C6 ØØ1D        0455Ø          DW      SBUFF$
20C8 55          0456Ø          DB      'USING '
     53 49 4E 47 2Ø
20CE ØØ1D        0457Ø          DW      SBUFF$
20DØ ØØ          0458Ø          NOP
                 0459Ø ;
                 0460Ø ;        Fetch default file extension
                 0461Ø ;
20D1 D5          0462Ø FEXT     PUSH    DE              ;Save FCB pointer
20D2 E5          0463Ø          PUSH    HL              ;Save EXT default pointer
20D3 EB          0464Ø          EX      DE,HL           ;Exchange pointers
20D4 23          0465Ø          INC     HL
20D5 Ø609        0466Ø          LD      B,9             ;Init for 9-char test
20D7 7E          0467Ø FEX1     LD      A,(HL)          ;Ret if extension start
20D8 FE2F        0468Ø          CP      '/'             ;  is found
20DA 28ØD        0469Ø          JR      Z,FEX3
20DC 38ØE        0470Ø          JR      C,FEX4          ;Jump on other separator
20DE FE3A        0471Ø          CP      ':'             ;Jump on digit Ø-9
20EØ 3804        0472Ø          JR      C,FEX2
20E2 FE41        0473Ø          CP      'A'             ;Jump on special char
20E4 3806        0474Ø          JR      C,FEX4
20E6 23          0475Ø FEX2     INC     HL              ;Advance past A-Z,Ø-9
20E7 10EE        0476Ø          DJNZ    FEX1
20E9 E1          0477Ø FEX3     POP     HL              ;User entered file ext
20EA D1          0478Ø          POP     DE              ;FCB start
20EB C9          0479Ø          RET
                 0480Ø ;
                 0481Ø ;        Use default extension
                 0482Ø ;
20EC 010F00      0483Ø FEX4     LD      BC,15           ;Point to position past
20EF 09          0484Ø          ADD     HL,BC           ;  the filespec
20FØ 54          0485Ø          LD      D,H
20F1 5D          0486Ø          LD      E,L
20F2 13          0487Ø          INC     DE              ;Make room for '/EXT'
20F3 13          0488Ø          INC     DE              ;  which is 4 chars
20F4 13          0489Ø          INC     DE
20F5 13          0490Ø          INC     DE
20F6 Ø3          0491Ø          INC     BC              ;Now move 16 bytes
20F7 EDB8        0492Ø          LDDR
20F9 E1          0493Ø          POP     HL              ;Recover pointer to EXT
20FA 23          0494Ø          INC     HL              ;Point to 3rd char
20FB 23          0495Ø          INC     HL
20FC ØE03        0496Ø          LD      C,3             ;Move in 3 chars
20FE EDB8        0497Ø          LDDR
2100 3E2F        0498Ø          LD      A,'/'           ;Put in the slash
2102 12          0499Ø          LD      (DE),A
2103 D1          0500Ø          POP     DE              ;Point back to FCB
2104 C9          0501Ø          RET
                 0502Ø ;
                 0503Ø ;        Get the code for the @PARAM SVC
                 0504Ø ;
2105             0505Ø *GET     PARAM:3
                 0305Ø ;PARAM/ASM - LS-DOS 6.2
                 0306Ø ;
                 0307Ø ;        Parse a field
                 0308Ø ;        (HL) => command line
                 0309Ø ;        (DE) => FCB area
                 0310Ø ;        (HL) <= 1st byte past non-<A-Z, a-z, Ø-9>
                 0311Ø ;                except 13, 3, "("
```

```
                    Ø312Ø ;       Z    <= found valid field
                    Ø313Ø ;       NZ   <= found invalid field
                    Ø314Ø ;
21Ø5 Ø6Ø8           Ø315Ø @PARSER LD     B,8             ;Set length
21Ø7 78             Ø316Ø @PAR1   LD     A,B
21Ø8 324621         Ø317Ø         LD     (PAR6+1),A      ;Stuff length for test
21ØB Ø4             Ø318Ø         INC    B
21ØC 7E             Ø319Ø PAR2    LD     A,(HL)
21ØD FEØ3           Ø32ØØ         CP     3               ;ETX?
21ØF 2826           Ø321Ø         JR     Z,PAR5
2111 FEØD           Ø322Ø         CP     CR              ;<ENTER>?
2113 2822           Ø323Ø         JR     Z,PAR5
2115 FE28           Ø324Ø         CP     '('             ;Begin of parm?
2117 281E           Ø325Ø         JR     Z,PAR5
2119 23             Ø326Ø         INC    HL              ;Bump pointer to next
211A CD4A21         Ø327Ø         CALL   TSTØ9AZ         ;Test if Ø-9,A-Z
211D 3ØØA           Ø328Ø         JR     NC,PAR3         ;Go if one of the above
211F FE61           Ø329Ø         CP     'a'             ;Check on lower case
2121 3814           Ø33ØØ         JR     C,PAR5          ;Jump on non-alpha
2123 FE7B           Ø331Ø         CP     'z'+1           ;Is it a-z?
2125 3Ø1Ø           Ø332Ø         JR     NC,PAR5         ;Jump on non-alpha
2127 CBAF           Ø333Ø         RES    5,A             ;Convert lower to upper
2129 Ø5             Ø334Ø PAR3    DEC    B               ;Count down
212A 28Ø8           Ø335Ø         JR     Z,PAR4
212C 12             Ø336Ø         LD     (DE),A          ;Xfer the char
212D AF             Ø337Ø         XOR    A               ;Show at least 1 valid
212E 324621         Ø338Ø         LD     (PAR6+1),A      ;Char was detected
2131 13             Ø339Ø         INC    DE              ;Bump FCB pointer
2132 18D8           Ø34ØØ         JR     PAR2            ;Loop
2134 Ø4             Ø341Ø PAR4    INC    B               ;Here on max chars ck'd
2135 18D5           Ø342Ø         JR     PAR2
2137 4F             Ø343Ø PAR5    LD     C,A             ;Save separator
2138 3EØ3           Ø344Ø         LD     A,3             ;Stuff ETX
213A 12             Ø345Ø         LD     (DE),A
                    Ø346Ø ;
                    Ø347Ø ;       Skip over spaces
                    Ø348Ø ;
213B 79             Ø349Ø         LD     A,C             ;Was separator a space?
213C FE2Ø           Ø35ØØ         CP     ' '
213E 2ØØ5           Ø351Ø         JR     NZ,PAR6         ;Don't skip if not
214Ø BE             Ø352Ø PAR5A   CP     (HL)            ;Next char a space?
2141 23             Ø353Ø         INC    HL
2142 28FC           Ø354Ø         JR     Z,PAR5A         ;Loop until not
2144 2B             Ø355Ø         DEC    HL              ;Backup to last non-space
                    Ø356Ø ;
                    Ø357Ø ;       Return status of field validity
                    Ø358Ø ;
2145 3EØØ           Ø359Ø PAR6    LD     A,Ø             ;Set Z-flag if at least
2147 B7             Ø36ØØ         OR     A               ;  1 valid char detected
2148 79             Ø361Ø         LD     A,C             ;Recover separator char
2149 C9             Ø362Ø         RET
                    Ø363Ø ;
                    Ø364Ø ;       Test if Ø-9 or A-Z
                    Ø365Ø ;
214A FE3Ø           Ø366Ø TSTØ9AZ CP     'Ø'             ;Special character?
214C D8             Ø367Ø         RET    C               ;Go if not in range
214D FE3A           Ø368Ø         CP     '9'+1           ;Jump on digit Ø-9
214F 38Ø5           Ø369Ø         JR     C,EXITC         ;Go if Ø-9 & make NC
2151 FE41           Ø37ØØ         CP     'A'             ;Jump on spec char
2153 D8             Ø371Ø         RET    C               ;Go if 3B-4Ø
2154 FE5B           Ø372Ø         CP     'Z'+1           ;Jump on A-Z
```

```
2156 3F       03730 EXITC   CCF                     ;Switch flag of result
2157 C9       03740         RET
              03750 ;
              03760 ;       Find parameter in table
              03770 ;       (HL) => pointer to line
              03780 ;       (DE) => pointer to buffer area
              03790 ;       (BC) => pointer to parameter table
              03800 ;       (BC) <= pointer to possible response byte
              03810 ;       (DE) <= parm vector address
              03820 ;         Z <= set if found
              03830 ;        NZ <= if not found in table
              03840 ;
2158 E5       03850 @FNDPRM PUSH    HL
2159 60       03860         LD      H,B             ;Xfer table addr
215A 69       03870         LD      L,C
215B 7E       03880         LD      A,(HL)          ;P/u 1st byte of table
215C 07       03890         RLCA                    ;  & test for enhanced
215D F5       03900         PUSH    AF              ;  table format
215E 3001     03910         JR      NC,FND1
2160 23       03920         INC     HL              ;Bump past indicator
2161 F1       03930 FND1    POP     AF              ;Old or enhanced format?
2162 F5       03940         PUSH    AF
2163 3E05     03950         LD      A,5             ;Init for old lengths
2165 010201   03960         LD      BC,1<8!2
2168 3006     03970         JR      NC,FND1A        ;Branch if old format
216A 7E       03980         LD      A,(HL)          ;  else get parm length
216B E60F     03990         AND     0FH             ;Strip flags
216D 3D       04000         DEC     A               ;Adjust for length-1
216E 04       04010         INC     B               ;Update offset to address
216F 23       04020         INC     HL              ;Bump past TYPE byte
2170 329E21   04030 FND1A   LD      (FND3A+1),A     ;Stuff the lengths
2173 80       04040         ADD     A,B
2174 32B821   04050         LD      (FND5A+1),A
2177 81       04060         ADD     A,C
2178 328021   04070         LD      (FND2+1),A
217B 1A       04080         LD      A,(DE)          ;P/u command line byte
217C BE       04090         CP      (HL)            ;Match 1st char of table?
217D 280C     04100         JR      Z,FND3          ;Jump if 1st char matches
217F 010800   04110 FND2    LD      BC,8            ;  else bypass that entry
2182 09       04120         ADD     HL,BC
2183 7E       04130         LD      A,(HL)          ;Test for table end
2184 B7       04140         OR      A
2185 20DA     04150         JR      NZ,FND1         ;Loop if more
2187 E1       04160         POP     HL              ;Clean flag from stack
2188 E1       04170         POP     HL              ;Rcvr saved reg &
2189 3C       04180         INC     A               ;  set NZ for not found
218A C9       04190         RET
218B F1       04200 FND3    POP     AF              ;Ck old or new table
218C F5       04210         PUSH    AF
218D 300E     04220         JR      NC,FND3A        ;Go if old format table
218F 2B       04230         DEC     HL              ;Ck if type byte permits
2190 CB66     04240         BIT     4,(HL)          ;  single-char abbrev
2192 23       04250         INC     HL
2193 2808     04260         JR      Z,FND3A         ;Go on no abbrev
2195 13       04270         INC     DE              ;Make sure the next char
2196 1A       04280         LD      A,(DE)          ;  is not in the range
2197 1B       04290         DEC     DE              ;  <0-9,A-Z> before
2198 CD4A21   04300         CALL    TST09AZ         ;  assuming abbrev
219B 381A     04310         JR      C,FND5A         ;Go on 1-char abbrevs
219D 0605     04320 FND3A   LD      B,5             ;5 more chars to match
219F E5       04330         PUSH    HL
```

```
21AØ D5          Ø434Ø          PUSH    DE
21A1 78          Ø435Ø          LD      A,B             ;Don't if trailing length
21A2 B7          Ø436Ø          OR      A               ;  is zero
21A3 281Ø        Ø437Ø          JR      Z,FND5
21A5 13          Ø438Ø FND4     INC     DE
21A6 23          Ø439Ø          INC     HL
21A7 1A          Ø44ØØ          LD      A,(DE)
21A8 FEØ3        Ø441Ø          CP      3               ;ETX?
21AA 2822        Ø442Ø          JR      Z,FND7
21AC FEØD        Ø443Ø          CP      CR              ;Jump on <ENTER>
21AE 281E        Ø444Ø          JR      Z,FND7
21BØ BE          Ø445Ø          CP      (HL)            ;Match?
21B1 2Ø16        Ø446Ø          JR      NZ,FND6         ;Jump if not
21B3 1ØFØ        Ø447Ø          DJNZ    FND4            ;  else loop
21B5 D1          Ø448Ø FND5     POP     DE              ;Parm matched
21B6 E1          Ø449Ø          POP     HL              ;Recover begin of parm
21B7 Ø1Ø6ØØ      Ø45ØØ FND5A    LD      BC,6            ;Point to address field
21BA Ø9          Ø451Ø          ADD     HL,BC
21BB 4D          Ø452Ø          LD      C,L             ;Save the response-byte
21BC 44          Ø453Ø          LD      B,H             ;  pointer in BC
21BD ØB          Ø454Ø          DEC     BC
21BE 5E          Ø455Ø          LD      E,(HL)          ;P/u parm table address
21BF 23          Ø456Ø          INC     HL
21CØ 56          Ø457Ø          LD      D,(HL)
21C1 F1          Ø458Ø          POP     AF              ;If not enhanced, change
21C2 38Ø2        Ø459Ø          JR      C,$+4           ;  pointer to bucket
21C4 Ø61D        Ø46ØØ          LD      B,SBUFF$<-8     ;  so we don't alter user
21C6 E1          Ø461Ø          POP     HL              ;Recover line position
21C7 AF          Ø462Ø          XOR     A               ;Show found
21C8 C9          Ø463Ø          RET
21C9 CD4A21      Ø464Ø FND6     CALL    TSTØ9AZ         ;Ck if Ø-9, A-Z
21CC 3ØØ5        Ø465Ø          JR      NC,FND8         ;Go if in range of above
21CE 7E          Ø466Ø FND7     LD      A,(HL)          ;Loop if table has
21CF FE2Ø        Ø467Ø          CP      ' '             ;  trailing spaces
21D1 28E2        Ø468Ø          JR      Z,FND5
21D3 D1          Ø469Ø FND8     POP     DE
21D4 E1          Ø47ØØ          POP     HL
21D5 18A8        Ø471Ø          JR      FND2
                 Ø472Ø ;
                 Ø473Ø ;        PARAM routine
                 Ø474Ø ;         (HL) => param line
                 Ø475Ø ;         (DE) => parm table
                 Ø476Ø ;         (DE) <= table address value
                 Ø477Ø ;            C <= # of parm
                 Ø478Ø ;            Z = OK
                 Ø479Ø ;           NZ = parm error
                 Ø48ØØ ;
21D7 23          Ø481Ø PARAMØ   INC     HL              ;Bump the pointer
21D8 7E          Ø482Ø PARAM    LD      A,(HL)          ;  and p/u char
21D9 FEØD        Ø483Ø          CP      CR
21DB C8          Ø484Ø          RET     Z               ;Return on enter
21DC FE2Ø        Ø485Ø          CP      ' '
21DE 28F7        Ø486Ø          JR      Z,PARAMØ        ;Loop on space
21EØ FE28        Ø487Ø          CP      '('
21E2 2Ø5E        Ø488Ø          JR      NZ,PARAM5       ;Jump if not left paren
21E4 1A          Ø489Ø          LD      A,(DE)          ;Check if enhanced table
21E5 Ø7          Ø49ØØ          RLCA
21E6 3Ø17        Ø491Ø          JR      NC,PARAM1
21E8 D5          Ø492Ø          PUSH    DE              ;Save pointer to start
21E9 13          Ø493Ø          INC     DE              ;Point to 1st TYPE byte
21EA E5          Ø494Ø          PUSH    HL              ;Save this posn
```

```
                    04950 ;
21EB 1A             04960 $?1       LD      A,(DE)          ;P/u TYPE byte
21EC E60F           04970           AND     0FH
21EE 280D           04980           JR      Z,$?2           ;Exit on end of table
21F0 6F             04990           LD      L,A             ;Point to response byte
21F1 2600           05000           LD      H,0
21F3 2C             05010           INC     L
21F4 19             05020           ADD     HL,DE
21F5 3600           05030           LD      (HL),0          ;Zero the response
21F7 23             05040           INC     HL              ;Bump to the next TYPE
21F8 23             05050           INC     HL
21F9 23             05060           INC     HL
21FA EB             05070           EX      DE,HL           ;Table pointer back to DE
21FB 18EE           05080           JR      $?1             ;Loop thru all response bytes
                    05090 ;
21FD E1             05100 $?2       POP     HL              ;Rcvr reg
21FE D1             05110           POP     DE              ;  & start of parm table
21FF D5             05120 PARAM1    PUSH    DE
2200 060F           05130           LD      B,15            ;Max 15-char field
2202 11001D         05140           LD      DE,SBUFF$       ;Point to buffer region
2205 23             05150           INC     HL              ;Bypass the '('
2206 CD0721         05160           CALL    @PAR1           ;Get the field
2209 2B             05170           DEC     HL              ;Backup to separator
220A D1             05180           POP     DE
220B 2013           05190           JR      NZ,ERROUT       ;Return if bad field
220D FE0D           05200           CP      CR              ;If separator was a CR,
220F 2001           05210           JR      NZ,$+3          ;  we need to counteract
2211 23             05220           INC     HL              ;  the DEC HL above
2212 D5             05230           PUSH    DE
2213 42             05240           LD      B,D             ;Table pointer to BC
2214 4B             05250           LD      C,E
2215 11001D         05260           LD      DE,SBUFF$       ;Parm in table?
2218 CD5821         05270           CALL    @FNDPRM
221B C5             05280           PUSH    BC              ;Save response pointer
221C 2805           05290           JR      Z,PARAM3        ;Jump if found in table
                    05300 ;
                    05310 ;         Parameter not in table - NZ condition
                    05320 ;
221E D1             05330 PARAM2    POP     DE              ;Pop response pointer
221F D1             05340           POP     DE              ;Pop parm table pointer
2220 3E2C           05350 ERROUT    LD      A,44            ;Set up PARM ERROR
2222 C9             05360           RET
                    05370 ;
                    05380 ;         Parameter found in table - parse the value
                    05390 ;
2223 7E             05400 PARAM3    LD      A,(HL)          ;Test for assignment
2224 FE3D           05410           CP      '='
2226 281C           05420           JR      Z,ASSIGN        ;Jump if parm=value
2228 01FFFF         05430           LD      BC,-1           ;  else set symbol TRUE
222B E3             05440 PARMSW    EX      (SP),HL         ;Get response byte
222C CBF6           05450           SET     6,(HL)          ;Turn on FLAG-SWITCH
                    05460 ;
                    05470 ;         Valid parm argument parsed into reg BC
                    05480 ;
222E EB             05490 PARAM4    EX      DE,HL           ;Address pointer to HL
222F 71             05500           LD      (HL),C          ;Stuff lo-order value
2230 23             05510           INC     HL
2231 70             05520           LD      (HL),B          ;Stuff hi-order value
2232 E1             05530           POP     HL              ;Rcvr parm line pointer
2233 D1             05540           POP     DE              ;Rcvr parm table pointer
2234 7E             05550           LD      A,(HL)
```

```
2235 FE2C     Ø5560          CP      ','             ;Comma separator?
2237 28C6     Ø5570          JR      Z,PARAM1
2239 FEØD     Ø5580          CP      CR
223B 28Ø5     Ø5590          JR      Z,PARAM5
223D FE29     Ø56ØØ          CP      ')'             ;Closing paren?
223F 2ØDF     Ø561Ø          JR      NZ,ERROUT       ;Leave with ERROR
2241 23       Ø562Ø          INC     HL              ;Bump line pointer
2242 AF       Ø563Ø PARAM5   XOR     A               ;Show all OK
2243 C9       Ø564Ø          RET
              Ø565Ø ;
              Ø566Ø ;        Parameter assignment statement
              Ø567Ø ;
2244 23       Ø568Ø ASSIGN   INC     HL              ;Advance token past '='
2245 7E       Ø569Ø          LD      A,(HL)
2246 FE22     Ø57ØØ          CP      '"'             ;Double quote string?
2248 282A     Ø571Ø          JR      Z,STRING
224A FE41     Ø572Ø          CP      'A'             ;Ck on digit or
224C 3815     Ø573Ø          JR      C,ASS3          ;  special character
224E CBAF     Ø574Ø          RES     5,A             ;Strip l/c if present
225Ø FE58     Ø575Ø          CP      'X'             ;Hexadecimal?
2252 28Ø7     Ø576Ø          JR      Z,ASS1
2254 CD9222   Ø577Ø          CALL    ONOFF           ;Ck on Y, N, ON, OFF
2257 28D2     Ø578Ø          JR      Z,PARMSW        ;Set FLAG-SWITCH if OK
2259 18C3     Ø579Ø          JR      PARAM2          ;  else error exit
225B 23       Ø58ØØ ASS1     INC     HL
225C CDBA22   Ø581Ø          CALL    HEXVAL          ;Ck on hex format
225F 2ØBD     Ø582Ø          JR      NZ,PARAM2       ;Error if bad format
2261 18Ø7     Ø583Ø          JR      ASS3A           ;  else bypass & set resp
              Ø584Ø ;
              Ø585Ø ;        Is the parameter numeric or flag ?
              Ø586Ø ;
2263 FE3Ø     Ø587Ø ASS3     CP      'Ø'             ;Parameter=number ?
2265 F5       Ø588Ø          PUSH    AF              ;CF = Ø if number
2266 CDE1Ø3   Ø589Ø          CALL    @DECHEX         ;Cvt # @ HL to bin in DE
2269 F1       Ø59ØØ          POP     AF
226A E3       Ø591Ø ASS3A    EX      (SP),HL         ;Get response pointer
226B 3ØØ3     Ø592Ø          JR      NC,ASS4         ;Show numeric if CF=Ø
226D CBF6     Ø593Ø          SET     6,(HL)          ;  otherwise show switch
226F 3A       Ø594Ø          DB      LD__A           ;Skip next instruction
227Ø CBFE     Ø595Ø ASS4     SET     7,(HL)          ;Set Numeric response bit
2272 18BA     Ø596Ø          JR      PARAM4
              Ø597Ø ;
              Ø598Ø ;        Parameter string entry
              Ø599Ø ;
2274 23       Ø6ØØØ STRING   INC     HL              ;Bypass '"'
2275 44       Ø6Ø1Ø          LD      B,H             ;Save starting address
2276 4D       Ø6Ø2Ø          LD      C,L
2277 7E       Ø6Ø3Ø STR1     LD      A,(HL)          ;P/u a char
2278 FE2Ø     Ø6Ø4Ø          CP      2ØH
227A 38A2     Ø6Ø5Ø          JR      C,PARAM2        ;Exit on control char
227C 23       Ø6Ø6Ø          INC     HL              ;Bump pointer
227D FE22     Ø6Ø7Ø          CP      '"'             ;Closing double quote
227F 2ØF6     Ø6Ø8Ø          JR      NZ,STR1
2281 E5       Ø6Ø9Ø          PUSH    HL              ;Save current pointer
2282 ED42     Ø61ØØ          SBC     HL,BC           ;Calc length of string
2284 7D       Ø611Ø          LD      A,L
2285 3D       Ø612Ø          DEC     A               ;Adjust for INC HL
2286 FE2Ø     Ø613Ø          CP      32              ;If len > 31, set to Ø
2288 3801     Ø614Ø          JR      C,$+3
228A AF       Ø615Ø          XOR     A
228B E1       Ø616Ø          POP     HL              ;Rcvr pointer
```

```
228C E3       Ø617Ø          EX      (SP),HL         ;Get response byte
228D F62Ø     Ø618Ø          OR      2ØH             ;Set FLAG-STRING
228F 77       Ø619Ø          LD      (HL),A
2290 189C     Ø62ØØ          JR      PARAM4
              Ø621Ø ;
              Ø622Ø ;        Check for Yes, No, On, Off
              Ø623Ø ;
2292 Ø1ØØØØ   Ø624Ø ONOFF    LD      BC,Ø            ;Init to FALSE
2295 D659     Ø625Ø          SUB     'Y'             ;Is it Yes?
2297 2811     Ø626Ø          JR      Z,ONO1          ;Jump on yes
2299 C6ØB     Ø627Ø          ADD     A,'Y'-'N'       ;Is it No?
229B 281Ø     Ø628Ø          JR      Z,ONO2          ;Jump on no
229D 3D       Ø629Ø          DEC     A               ;Is it 'O'n or 'O'ff?
229E CØ       Ø63ØØ          RET     NZ              ;Return if not on/off
229F 23       Ø631Ø          INC     HL              ;Bump pointer to next
22AØ 7E       Ø632Ø          LD      A,(HL)          ;  character & p/u
22A1 CBAF     Ø633Ø          RES     5,A             ;Set lower to upper
22A3 FE46     Ø634Ø          CP      'F'
22A5 28Ø6     Ø635Ø          JR      Z,ONO2          ;Jump on off
22A7 FE4E     Ø636Ø          CP      'N'
22A9 CØ       Ø637Ø          RET     NZ              ;Return if not on
22AA Ø1FFFF   Ø638Ø ONO1     LD      BC,-1           ;Init to true
22AD 23       Ø639Ø ONO2     INC     HL              ;Ignore trailing part
22AE 7E       Ø64ØØ          LD      A,(HL)          ;  of word until closing
22AF FE29     Ø641Ø          CP      ')'             ;  ")" or comma separator
22B1 C8       Ø642Ø          RET     Z               ;  or CR
22B2 FEØD     Ø643Ø          CP      CR
22B4 C8       Ø644Ø          RET     Z
22B5 FE2C     Ø645Ø          CP      ','
22B7 C8       Ø646Ø          RET     Z
22B8 18F3     Ø647Ø          JR      ONO2            ;Loop
              Ø648Ø ;
              Ø649Ø ;        Process hexadecimal assignment
              Ø65ØØ ;
22BA Ø1ØØØØ   Ø651Ø HEXVAL   LD      BC,Ø            ;Init value to zero
22BD 7E       Ø652Ø          LD      A,(HL)          ;P/u a char
22BE FE27     Ø653Ø          CP      27H             ;Must be single quote
22CØ CØ       Ø654Ø          RET     NZ              ;Ret if not
22C1 23       Ø655Ø HEX1     INC     HL              ;Bump past it
22C2 7E       Ø656Ø          LD      A,(HL)          ;P/u possible hex digit
22C3 D63Ø     Ø657Ø          SUB     3ØH             ;Begin conversion
22C5 38ØC     Ø658Ø          JR      C,HEX2          ;Jump if < "Ø"
22C7 FEØA     Ø659Ø          CP      1Ø              ;Ck for Ø-9
22C9 381Ø     Ø66ØØ          JR      C,HEX3          ;Jump if digit is Ø-9
22CB CBAF     Ø661Ø          RES     5,A             ;Strip l/c if present
22CD D6Ø7     Ø662Ø          SUB     7               ;  else ck A-F
22CF FE1Ø     Ø663Ø          CP      16
22D1 38Ø8     Ø664Ø          JR      C,HEX3          ;Jump if A-F
22D3 7E       Ø665Ø HEX2     LD      A,(HL)          ;Test for closing quote
22D4 FE27     Ø666Ø          CP      27H
22D6 23       Ø667Ø          INC     HL              ;Bump pointer
22D7 C8       Ø668Ø          RET     Z               ;Ret if closing quote
22D8 2B       Ø669Ø          DEC     HL              ;  else backup, set OK,
22D9 AF       Ø67ØØ          XOR     A               ;  then return
22DA C9       Ø671Ø          RET
22DB C5       Ø672Ø HEX3     PUSH    BC              ;Exchange BC & HL
22DC E3       Ø673Ø          EX      (SP),HL         ;  and save HL
22DD 29       Ø674Ø          ADD     HL,HL           ;Multiply by 16
22DE 29       Ø675Ø          ADD     HL,HL
22DF 29       Ø676Ø          ADD     HL,HL
22EØ 29       Ø677Ø          ADD     HL,HL
```

```
22E1 44          Ø678Ø          LD      B,H             ;Merge new digit
22E2 85          Ø679Ø          ADD     A,L
22E3 4F          Ø68ØØ          LD      C,A
22E4 E1          Ø681Ø          POP     HL              ;Recover pointer
22E5 18DA        Ø682Ø          JR      HEX1            ;Loop
                 Ø5Ø6Ø ;
22E7 43          Ø5Ø7Ø DFTEXT   DB      'CMD'           ;Default extension
     4D 44
                 Ø5Ø8Ø          IF      @MOD2
                 Ø5Ø9Ø RDYMSG$  DB      LF,14,'LS-DOS Ready',CR
                 Ø51ØØ          ELSE
22EA ØA          Ø511Ø RDYMSG$  DB      LF,14,'TRSDOS Ready',CR
     ØE 54 52 53 44 4F 53 2Ø
     52 65 61 64 79 ØD
                 Ø512Ø          ENDIF
22F9             Ø513Ø LAST     EQU     $
                 Ø514Ø          IFGT    $,DIRBUF$
                 Ø515Ø          ERR     'Module too big'
                 Ø516Ø          ENDIF
23FE             Ø517Ø          ORG     MAXCOR$-2
23FE F9Ø4        Ø518Ø          DW      LAST-SYS1       ;Size of overlay
1EØØ             Ø519Ø          END     SYS1
```

$?1	21EB	$?2	21FD
$A2	Ø3B8	$A3	Ø3B9
@$SYS	Ø8FØ	@@1	ØØØØ
@@3	ØØØØ	@@4	ØØØØ
@ADTSK	1CDA	@BANK	Ø877
@BREAK	196F	@BYTEIO	13ØØ
@CKBRKC	Ø553	@CKDRV	1993
@CKTSK	1CF5	@CLOSE	1999
@CMNDI	197E	@CMNDR	197B
@DATE	Ø7A8	@DBGHK	199F
@DCRES	19C4	@DCSTAT	19B5
@DEBUG	19AØ	@DECHEX	Ø3E1
@DIRRD	18BB	@DIRWR	18Ø3
@DIV8	1927	@DODIR	19AF
@DSP	Ø642	@DSPLY	Ø52D
@EXIT	1BØB	@FEXT	1984
@FNAME	199C	@FNDPRM	2158
@FSPEC	1981	@GATRD	1874
@GERMAN	ØØØØ	@GET	Ø638
@GTDCT	1A1E	@GTMOD	19B2
@HEX16	Ø7BD	@HEX8	Ø7C2
@HIGH$	1948	@HITRD	1897
@HZ5Ø	ØØØØ	@ICNFG	ØØ86
@INTL	ØØØØ	@IPL	1BF2
@KBD	Ø635	@KEY	Ø628
@KITSK	ØØ89	@KLTSK	1CDØ
@LOC	14B3	@LOF	14DE
@LOGOT	Ø5ØØ	@MOD2	ØØØØ
@MSG	Ø53Ø	@MUL16	Ø6C9
@NMI	ØØ66	@OPEN	198A
@PAR1	21Ø7	@PARAM	1987
@PAUSE	Ø382	@PEOF	14A2
@PRINT	Ø528	@PRT	Ø63D
@RAMDIR	19AC	@RDHDR	19D8
@RDSSC	18D8	@RDTRK	19EØ
@REMOVE	19A6	@RENAME	1996
@RMTSK	1CD7	@RPTSK	1CEB
@RSLCT	19D4	@RST ØØ	ØØØØ
@RST1Ø	ØØ1Ø	@RST18	ØØ18
@RST28	ØØ28	@RST3Ø	ØØ3Ø
@RSTNMI	ØFE9	@RSTOR	19C8
@RUN	1B1D	@RWRIT	13AD
@SEEKSC	1421	@SKIP	143Ø
@SMALL	ØØØØ	@SOUND	Ø392
@TIME	Ø78D	@USA	FFFF
@VDCTL3	ØD38	@VER	156Ø
@WEOF	14EC	@WHERE	1979
@WRSEC	19E8	@WRSSC	19EC
@_VDCTL	ØD42	ADDR_2_ROWCOL	ØDF1
ASS1	225B	ASS3	2263
ASS4	227Ø	ASSIGN	2244
BAR$	Ø2Ø1	BOOTIT	1F22
BREAK?	1C6Ø	BRKVEC$	1C88
CASHK$	ØA7B	CFCB$	ØØEØ
CFLAG$	ØØ6C	CKMOD@	1A7F
CKOPEN@	1568	CLEANUP	1E46
CMD1A	1E97	CMD2	1E9D
CMD3Ø	1E3C	CMD3A	1EB2
CMDCONT	1E8E	COMMENT	1FØ5
CORE$	Ø3ØØ	CR	ØØØD

$A1	Ø3B7		
$CKEOF	147Ø		
@@2	ØØØØ		
@ABORT	1BØ8		
@BKSP	1486		
@CHNIO	Ø689		
@CKEOF	158F		
@CLS	Ø545		
@CTL	Ø623		
@DCINIT	19CØ		
@DCTBYT	1A2B		
@DIRCYL	18F7		
@DIV16	Ø6E3		
@DOKEY	19A9		
@ERROR	1BØF		
@FLAGS	196A		
@FRENCH	ØØØØ		
@GATWR	1875		
@GTDCB	199Ø		
@HDFMT	19E4		
@HEXDEC	Ø6F6		
@HITWR	1898		
@INIT	198D		
@JCL	Ø63Ø		
@KEYIN	Ø585		
@LOAD	1B38		
@LOGER	Ø5Ø3		
@MOD4	FFFF		
@MUL8	19ØA		
@OPREG	ØØ84		
@PARSER	21Ø5		
@POSN	1434		
@PUT	Ø645		
@RDSEC	19F4		
@READ	1513		
@REW	149B		
@RREAD	1473		
@RSTØ8	ØØØ8		
@RST2Ø	ØØ2Ø		
@RST38	ØØ38		
@RSTREG	Ø68Ø		
@SEEK	19DØ		
@SLCT	19BC		
@STEPI	19CC		
@VDCTL	ØB99		
@VRSEC	19DC		
@WRITE	1531		
@WRTRK	19FØ		
AFLAG$	ØØ6A		
ASS3A	226A		
AUTO?	1FF1		
BOOTST$	439D		
BUR$	Ø2ØØ		
CFGFCB$	ØØEØ		
CKNOEXC	1ED7		
CMD	1E41		
CMD2Ø	1EBC		
CMD4	1F11		
CONFIG$	2Ø3F		
CRTBGN$	F8ØØ		

CYL_GRN	16AE	D@FBYT8	1A26	DATE$	ØØ33
DAYTBL$	Ø4C7	DBGOFF	1E5F	DBGSV$	ØØAØ
DCBKL$	ØØ31	DCT$	Ø47Ø	DCTBYT8@	1A29
DCTFLD@	1A34	DFLAG$	ØØ6D	DFTEXT	22E7
DIRBUF$	23ØØ	DIS_DO_RAM	Ø846	DODATA$	ØB94
DODCB$	Ø21Ø	DO_CONTROL	ØC44	DO_DSPCHAR	ØCB8
DO_INVERT_DIS	ØC8C	DO_INVERT_ENA	ØC89	DO_INVERT_OFF	ØC9B
DO_MASK	ØØØØ	DO_RET	ØBCB	DO_RET1	ØBCC
DO_SCROLL	ØCCE	DO_TABS	ØBEA	DSKTYP$	Ø4CØ
DTPMT$	Ø4C2	DVREND$	ØFF4	DVRHI$	Ø2Ø6
EFLAG$	ØØ6E	ENADIS_DO_RAM	Ø817	ERROUT	222Ø
EXITC	2156	EXTDBG$	19A4	FDDINT$	ØØØE
FEMSK$	ØØ6F	FEX1	2ØD7	FEX2	2ØE6
FEX3	2ØE9	FEX4	2ØEC	FEXT	2ØD1
FLGTAB$	ØØ6A	FND1	2161	FND1A	217Ø
FND2	217F	FND3	218B	FND3A	219D
FND4	21A5	FND5	21B5	FND5A	21B7
FND6	21C9	FND7	21CE	FND8	21D3
FSP1	2Ø6E	FSP2	2Ø79	FSP3	2Ø86
FSP4	2Ø8E	FSP5	2ØA1	FSP6	2ØAE
FSPEC	2Ø5D	GET_@_ROWCOL	ØDAE	HERTZ$	Ø75Ø
HEX1	22C1	HEX2	22D3	HEX3	22DB
HEXVAL	22BA	HIGH$	Ø4ØE	HKRES$	1A6C
IFLAG$	ØØ72	INBUF$	Ø42Ø	INTIM$	ØØ3C
INTMSK$	ØØ3D	INTVC$	ØØ3E	JCLCB$	Ø2Ø3
JDCB$	ØØ24	JFCB$	ØØCØ	JLDCB$	Ø23Ø
JRET$	ØØ26	KCK@	Ø7D6	KFLAG$	ØØ74
KIDATA$	Ø8FC	KIDCB$	Ø2Ø8	LAST	22F9
LBANK$	Ø2Ø2	LDRV$	ØØ23	LD__A	ØØ3A
LF	ØØØA	LFLAG$	ØØ75	LIBA	8ØØØ
LIBB	AØØØ	LIBC	CØØØ	LIBTBL$	1F24
LNKFCB@	1566	LOW$	ØØ1E	LSVC$	ØØØD
MAXCOR$	24ØØ	MAXDAY$	Ø4Ø1	MINCOR$	3ØØØ
MODOUT$	ØØ76	MONTBL$	Ø4DC	NFLAG$	ØØ77
NOEXC	1EDC	NOTLIB	1EF2	ONO1	22AA
ONO2	22AD	ONOFF	2292	OPREG$	ØØ78
OPREG_SV_AREA	Ø86E	OPREG_SV_PTR	Ø835	ORARET@	14DC
OSRLS$	ØØ3B	OSVER$	ØØ85	OVRLY$	ØØ69
PAKNAM$	Ø41Ø	PAR2	21ØC	PAR3	2129
PAR4	2134	PAR5	2137	PAR5A	214Ø
PAR6	2145	PARAM	21D8	PARAMØ	21D7
PARAM1	21FF	PARAM2	221E	PARAM3	2223
PARAM4	222E	PARAM5	2242	PARMSW	222B
PAUSE@	Ø382	PCSAVE$	Ø7AF	PDRV$	ØØ1B
PHIGH$	ØØ1C	PRDCB$	Ø218	PREPTBL	2ØBØ
PUTA@DE	ØDCD	PUT_@	ØDCA	PUT_@_ROWCOL	ØDC6
RDYMSG$	22EA	RFLAG$	ØØ7B	ROWCOL_2_ADDR	ØDDØ
RST38@	1BFF	RSTOR$	Ø4C4	RWRIT@	13A2
S1DCB$	Ø238	SBUFF$	1DØØ	SET@EXEC	1A79
SET_SCROLL	ØCF3	SFCB$	ØØ8C	SFLAG$	ØØ7C
SIDCB$	Ø22Ø	SODCB$	Ø228	SPACE4$	2142
STACK$	Ø38Ø	START$	ØØØØ	STR1	2277
STRING	2274	SVCRET$	ØØØB	SVCTAB$	Ø1ØØ
SYS1	1EØØ	SYS1BGN	1EØ4	SYSERR$	1B13
TCB$	ØØ4E	TFLAG$	ØØ7D	TIME$	ØØ2D
TIMER$	ØØ2C	TIMSL$	ØØ2B	TIMTSK$	Ø713
TMPMT$	Ø4C3	TRACE_INT	Ø7B1	TSTØ9AZ	214A
TSTEXC	1EE5	TYPHK$	ØA8F	TYPTSK$	ØB26
USTOR$	ØØ13	VFLAG$	ØØ7F	WHAT	1FØ1
WRINT$	ØØ8Ø	ZERO$	Ø4Ø1	ZEROA@	13AØ

ØØØØØ Total errors

NOTES:

NOTES:

SYS2 is a multi-function overlay. It handles creating, opening, and renaming files, hashing filenames and passwords, checking a drive for a mounted diskette, and locating a specified or free DCB. It contains the code for the SVCs @INIT, @OPEN, @RENAME, @GTDCB, and @CKDRV.

```
                00100 ;SYS2/ASM - LS-DOS 6.2
0000            00110         TITLE   <SYS2 - LS-DOS 6.2>
                00120 ;
                00130 ; This SYS module performs the following functions:
                00140 ; . OPENs an existing File or Device
                00150 ; . INITs a new File
                00160 ; . Checks availability of a specific drive
                00170 ; . Hashes an 11-byte field (file name & ext)
                00180 ; . Hashes an 8-byte field (password)
                00190 ; . Renames a filespec/devspec
                00200 ; . Gets the address of a device control block
                00210 ;
000D            00220 CR      EQU     13
                00230 *LIST   OFF                     ;Get SYS0/EQU
                00250 *LIST   ON
0000            00260 *GET    COPYCOM:3               ;Copyright message
                03010 ; COPYCOM - File for Copyright COMment block
                03020 ;
0000            03030         COM     '<*(C) 1982,83,84 by LSI*>'
                03040 ;
                00270 ;
1E00            00280         ORG     1E00H
                00290 ;
1E00 E670       00300 SYS2    AND     70H             ;Strip all but entry
1E02 C8         00310         RET     Z               ;Back on zero entry
1E03 FE10       00320         CP      10H             ;Check for OPEN
1E05 CA411F     00330         JP      Z,OPEN
1E08 FE20       00340         CP      20H             ;Check for INIT
1E0A CA0621     00350         JP      Z,INIT
1E0D FE70       00360         CP      70H             ;Check for rename
1E0F CA3922     00370         JP      Z,RENAME
1E12 FE30       00380         CP      30H             ;Get a DCB?
1E14 2849       00390         JR      Z,GTDCB
1E16 FE40       00400         CP      40H             ;Drive availability?
1E18 2860       00410         JR      Z,CKDRV
1E1A FE60       00420         CP      60H             ;Check password hash
1E1C 2810       00430         JR      Z,HASHPSWD
                00440 ;
                00450 ;       Routine to hash a file name
                00460 ;
                00470 HASHNAME
1E1E 060B       00480         LD      B,11            ;Init for 11 chars
1E20 AF         00490         XOR     A               ;Clear for start
1E21 AE         00500 HNAME1  XOR     (HL)            ;Modulo 2 addition
1E22 23         00510         INC     HL              ;Bump to next character
1E23 07         00520         RLCA                    ;Rotate bit structure
1E24 10FB       00530         DJNZ    HNAME1          ;  & loop for field len
1E26 B7         00540         OR      A               ;Do not permit a zero
1E27 2001       00550         JR      NZ,HNAME2       ;  hash code
1E29 3C         00560         INC     A
1E2A 32E022     00570 HNAME2  LD      (FILEHASH),A    ;Stuff code for later
1E2D C9         00580         RET
                00590 ;
                00600 ;       Hash a password
                00610 ;
                00620 HASHPSWD
1E2E 210700     00630         LD      HL,7            ;Hashing will be from
1E31 19         00640         ADD     HL,DE           ;  right to left so
1E32 EB         00650         EX      DE,HL           ;  point to lo-order
1E33 21FFFF     00660         LD      HL,-1           ;Init shift reg to 1's
1E36 0608       00670         LD      B,8             ;Init for 8-char string
```

```
1E38 1A      00680 HPSWD1  LD      A,(DE)          ;P/u the next byte
1E39 D5      00690         PUSH    DE              ;  & save the pointer
1E3A 57      00700         LD      D,A
1E3B 5C      00710         LD      E,H
1E3C 7D      00720         LD      A,L             ;Modulo 2 add bits 0-2
1E3D E607    00730         AND     7               ;  to bits 4-6 of the
1E3F 0F      00740         RRCA                    ;  16-bit shift register
1E40 0F      00750         RRCA
1E41 0F      00760         RRCA
1E42 AD      00770         XOR     L
1E43 6F      00780         LD      L,A             ;Shift shift-register
1E44 2600    00790         LD      H,0             ;  left by 4-bits to
1E46 29      00800         ADD     HL,HL           ;  isolate bits 4-7
1E47 29      00810         ADD     HL,HL
1E48 29      00820         ADD     HL,HL
1E49 29      00830         ADD     HL,HL
1E4A AC      00840         XOR     H               ;Mod 2 add SR bits 4-7
1E4B AA      00850         XOR     D               ;Mod 2 add new byte
1E4C 57      00860         LD      D,A             ;Save tempy for hi-order
1E4D 7D      00870         LD      A,L
1E4E 29      00880         ADD     HL,HL
1E4F AC      00890         XOR     H
1E50 AB      00900         XOR     E
1E51 5F      00910         LD      E,A
1E52 EB      00920         EX      DE,HL           ;SR result to HL
1E53 D1      00930         POP     DE              ;P/u pointer to string
1E54 1B      00940         DEC     DE              ;  & point to next byte
1E55 10E1    00950         DJNZ    HPSWD1          ;Loop for field length
1E57 AF      00960         XOR     A
1E58 C9      00970         RET
             00980 ;
             00990 ;       Routine to locate a Device Control Block
             01000 ;
1E59 DD5E01  01010 GETDCB  LD      E,(IX+1)        ;P/u the 2-character
1E5C DD5602  01020         LD      D,(IX+2)        ;  device name
1E5F 218802  01030 GTDCB   LD      HL,KIDCB$       ;Point to 1st DCB
1E62 E5      01040 DEV1    PUSH    HL
1E63 7D      01050         LD      A,L             ;Point to device
1E64 C606    01060         ADD     A,6             ;  name field
1E66 6F      01070         LD      L,A
1E67 7E      01080         LD      A,(HL)          ;P/u 1st char of name
1E68 2C      01090         INC     L               ;Point to 2nd char
1E69 BB      01100         CP      E               ;Compare 1st for match
1E6A 2006    01110         JR      NZ,DEV2         ;No match? then loop
1E6C 7E      01120         LD      A,(HL)          ;1st matches, does 2nd?
1E6D BA      01130         CP      D
1E6E 2002    01140         JR      NZ,DEV2         ;Loop if no match
1E70 E1      01150         POP     HL              ;Get start of DCB
1E71 C9      01160         RET
1E72 F1      01170 DEV2    POP     AF              ;Pop last DCB start
1E73 2C      01180         INC     L               ;Inc to start of next DCB
1E74 20EC    01190         JR      NZ,DEV1         ;Bypass if not at end
             01200 ;
             01210 ;       Device not found in tables
             01220 ;
1E76 3E08    01230         LD      A,8             ;"device not available"
1E78 B7      01240         OR      A
1E79 C9      01250         RET
             01260 ;
             01270 ;       Check a drive for availability
             01280 ;
```

```
1E7A FDE5      01290 CKDRV   PUSH    IY              ;We use IY in disk I/O
1E7C CD1E1A    01300         CALL    @GTDCT          ;Get driver routine addr
1E7F FD7E00    01310         LD      A,(IY+0)        ;P/u drive vector
1E82 FEC3      01320         CP      0C3H            ;Ck for enabled
1E84 C2141F    01330         JP      NZ,CKDR5        ;Bypass if disabled
1E87 E5        01340         PUSH    HL
1E88 D5        01350         PUSH    DE
1E89 FDCB035E  01360         BIT     3,(IY+3)        ;Test for HARD drive
1E8D 2018      01370         JR      NZ,CKDRV1A      ;If so bypass range check
1E8F FD7E06    01380         LD      A,(IY+6)        ;Make sure the current
1E92 FDBE05    01390         CP      (IY+5)          ;  cylinder is in range
1E95 3006      01400         JR      NC,CKDRV1       ;Go if in range
1E97 CDC819    01410         CALL    @RSTOR          ;Restore drive
1E9A C2231F    01420         JP      NZ,CKDR7A       ;Go if error
               01430 ;
1E9D FD5605    01440 CKDRV1  LD      D,(IY+5)        ;P/u current track
1EA0 1E00      01450         LD      E,0             ;Set for sector 0
1EA2 CDD019    01460         CALL    @SEEK           ;Set track info to FDC
1EA5 207C      01470         JR      NZ,CKDR7A       ;Go if error
1EA7 CDD419    01480 CKDRV1A CALL    @RSLCT          ;Wait until not busy
1EAA 2077      01490         JR      NZ,CKDR7A       ;Not there - ret NZ
1EAC FDCB035E  01500         BIT     3,(IY+3)        ;If hard drive, bypass
1EB0 2055      01510         JR      NZ,CKDR3A       ;  GAT data update
1EB2 FDCB0466  01520         BIT     4,(IY+4)        ;If "ALIEN" by pass
1EB6 202C      01530         JR      NZ,CKDR2B       ;  test of index pulses
               01540         IF      @MOD4
1EB8 3A0E00    01550         LD      A,(FDDINT$)     ;Check 'SMOOTH' state
1EBB B7        01560         OR      A
1EBC 3E09      01570         LD      A,09            ;Set MSB of count down
1EBE 2803      01580         JR      Z,INTRON        ;Go if not SMOOTH
1EC0 CB3F      01590         SRL     A               ;Divide the count by two
1EC2 F3        01600         DI
               01610         ENDIF
               01620         IF      @MOD2
               01630         LD      A,20
               01640         ENDIF
1EC3 32D51E    01650 INTRON  LD      (CDCNT+1),A     ;Store in 'LD H' instruction
1EC6 212000    01660         LD      HL,0020H        ;Set up count (short)
               01670 ;
               01680 ;       Test for diskette in drive & rotating
               01690 ;
1EC9 CD171F    01700 CKDR1   CALL    INDEX           ;Test index pulse
1ECC 20FB      01710         JR      NZ,CKDR1        ;Jump on index
1ECE FDCB047E  01720         BIT     7,(IY+4)        ;Check CKDRV inhibit bit
1ED2 2010      01730         JR      NZ,CKDR2B       ;If on skip index test
1ED4 2600      01740 CDCNT   LD      H,00H           ;CKDRV counter (long)
               01750                                 ;Count set from above
1ED6 CD171F    01760 CKDR2   CALL    INDEX           ;Test index pulse
1ED9 28FB      01770         JR      Z,CKDR2         ;Jump on no index
               01780         IF      @MOD4
1EDB FB        01790         EI                      ;OK for INTs now
               01800         ENDIF
1EDC 212000    01810         LD      HL,0020H        ;Index off wait (short)
1EDF CD171F    01820 CKDR2A  CALL    INDEX
1EE2 20FB      01830         JR      NZ,CKDR2A       ;Jump on index
               01840 ;
               01850 ;       Diskette is rotating
               01860 ;
1EE4 F5        01870 CKDR2B  PUSH    AF              ;Save FDC status
1EE5 CDF718    01880         CALL    @DIRCYL         ;Get directory track in D
1EE8 21001D    01890         LD      HL,SBUFF$       ;Point to HIT buffer
```

```
1EEB 5D       Ø19ØØ          LD      E,L             ;Sector Ø for GAT
1EEC CDD818   Ø191Ø          CALL    @RDSSC          ;Read the GAT
1EEF 2Ø31     Ø192Ø          JR      NZ,CKDR7        ;Jump on error
1EF1 2ACC1D   Ø193Ø          LD      HL,(SBUFF$+ØCCH)        ;P/u excess tracks
1EF4 3E22     Ø194Ø          LD      A,22H           ;Add offset
1EF6 85       Ø195Ø          ADD     A,L
1EF7 FD77Ø6   Ø196Ø          LD      (IY+6),A        ;Max track # to DCT
1EFA FDCBØ4AE Ø197Ø          RES     5,(IY+4)        ;Set to side Ø
1EFE CB6C     Ø198Ø          BIT     5,H             ;Test double sided
1FØØ 28Ø4     Ø199Ø          JR      Z,CKDR3         ;Jump if only single
1FØ2 FDCBØ4EE Ø2ØØØ          SET     5,(IY+4)        ;Set for side 2
1FØ6 F1       Ø2Ø1Ø CKDR3    POP     AF              ;Recover FDC status
1FØ7 Ø7       Ø2Ø2Ø CKDR3A   RLCA                    ;Shift write prot to 7
1FØ8 FDB6Ø3   Ø2Ø3Ø          OR      (IY+3)          ;Merge Soft WP bit
1FØB E68Ø     Ø2Ø4Ø          AND     8ØH             ;Strip all but 7
1FØD 326E2Ø   Ø2Ø5Ø          LD      (OPNCB9+1),A    ;Save WP status for OPNCB
1F1Ø 87       Ø2Ø6Ø          ADD     A,A             ;Write prot to carry flg
              Ø2Ø7Ø ;
1F11          Ø2Ø8Ø CKDR4    EQU     $
1F11 FB       Ø2Ø9Ø          EI
1F12 D1       Ø21ØØ          POP     DE
1F13 E1       Ø211Ø          POP     HL
1F14 FDE1     Ø212Ø CKDR5    POP     IY
1F16 C9       Ø213Ø          RET
1F17 7C       Ø214Ø INDEX    LD      A,H
1F18 B5       Ø215Ø          OR      L
1F19 28Ø7     Ø216Ø          JR      Z,CKDR7
1F1B 2B       Ø217Ø          DEC     HL
1F1C CDD419   Ø218Ø          CALL    @RSLCT          ;Check for index pulse
1F1F CB4F     Ø219Ø          BIT     1,A             ;Test index
1F21 C9       Ø22ØØ          RET
1F22 F1       Ø221Ø CKDR7    POP     AF
              Ø222Ø ;
1F23 B7       Ø223Ø CKDR7A   OR      A               ;Set NZ ret
1F24 18EB     Ø224Ø          JR      CKDR4           ;  and exit
              Ø225Ø ;
              Ø226Ø ;        OPEN a device
              Ø227Ø ;        Device Control Blocks are from X'Ø2Ø8' - X'Ø2FF'
              Ø228Ø ;
1F26 CD591E   Ø229Ø DEVOPEN  CALL    GETDCB          ;Find the DCB named
1F29 CØ       Ø23ØØ          RET     NZ              ;  in the IX pointer
              Ø231Ø ;
              Ø232Ø ;        Found the needed Device Control Block
              Ø233Ø ;
1F2A 44       Ø234Ø DEV4     LD      B,H             ;Xfer dcb vector to BC
1F2B 4D       Ø235Ø          LD      C,L
1F2C DDE5     Ø236Ø          PUSH    IX              ;User DCB to HL
1F2E E1       Ø237Ø          POP     HL
1F2F 361Ø     Ø238Ø          LD      (HL),1ØH        ;Show routed
1F31 23       Ø239Ø          INC     HL
1F32 71       Ø24ØØ          LD      (HL),C          ;Stuff dcb vector
1F33 23       Ø241Ø          INC     HL
1F34 7Ø       Ø242Ø          LD      (HL),B
1F35 23       Ø243Ø          INC     HL
1F36 AF       Ø244Ø          XOR     A               ;Zero next 3 bytes
1F37 77       Ø245Ø          LD      (HL),A
1F38 23       Ø246Ø          INC     HL
1F39 77       Ø247Ø          LD      (HL),A
1F3A 23       Ø248Ø          INC     HL
1F3B 77       Ø249Ø          LD      (HL),A
1F3C 23       Ø25ØØ          INC     HL
```

```
1F3D 73       02510          LD      (HL),E          ;Stuff dcb name
1F3E 23       02520          INC     HL
1F3F 72       02530          LD      (HL),D
1F40 C9       02540          RET
              02550 ;
              02560 ;
              02570 ;        OPEN a file
              02580 ;         HL <= the address of a 256-byte buffer
              02590 ;         DE <= the address of a 32-byte FCB
              02600 ;          B <= the logical record length (LREC)
              02610 ;
1F41 CD6615   02620 OPEN     CALL    LNKFCB@         ;Set up link to dcb
1F44 3A7C00   02630 OPEN1    LD      A,(SFLAG$)      ;Stuff current sysflag
1F47 322420   02640          LD      (OPEN14+1),A    ;  to check later then
1F4A E6F8     02650          AND     0F8H            ;  remove bits 0, 1 & 2
1F4C 327C00   02660          LD      (SFLAG$),A
1F4F DD7E00   02670          LD      A,(IX+0)
1F52 FE2A     02680          CP      '*'             ;If name starts with '*',
1F54 28D0     02690          JR      Z,DEVOPEN       ;  it is a device spec
1F56 78       02700          LD      A,B             ;P/u Lrl requested
1F57 32FE22   02710          LD      (LREC$),A
1F5A 22B320   02720          LD      (OPNCB4+1),HL   ;Stuff disk I/O buffer
1F5D DDE5     02730          PUSH    IX              ;Transfer the filespec
1F5F E1       02740          POP     HL              ;  into the system
1F60 CD8721   02750          CALL    XFRSPEC         ;  buffer area
1F63 C0       02760          RET     NZ              ;Return if bad name
1F64 21E922   02770          LD      HL,NAME$EXT     ;Point to name/ext field
1F67 CD1E1E   02780          CALL    HASHNAME        ;  & hash it (11 chars)
1F6A 11E122   02790          LD      DE,PSWDBUF      ;Point to the password
1F6D CD2E1E   02800          CALL    HASHPSWD        ;  & hash it
1F70 22F422   02810          LD      (PW$HASH1),HL   ;Stuff owner pswd
1F73 22F622   02820          LD      (PW$HASH2),HL   ;Stuff user pswd
1F76 3E00     02830 OPEN2    LD      A,0             ;P/u drive <FF-07>
1F78 4F       02840          LD      C,A
1F79 3C       02850          INC     A               ;Jump if :d entered
1F7A 2001     02860          JR      NZ,OPEN3
1F7C 4F       02870          LD      C,A
1F7D CD7A1E   02880 OPEN3    CALL    CKDRV           ;Drive available?
1F80 2013     02890          JR      NZ,OPEN6        ;Jump if not
1F82 CD9718   02900          CALL    @HITRD          ;Get hash index table
1F85 C0       02910          RET     NZ              ;Return if read error
              02920 ;
              02930 ;        Compare hashed filename/ext with each entry
              02940 ;        in the HIT to see if file is on this drive
              02950 ;
1F86 7E       02960 OPEN4    LD      A,(HL)          ;Bypass HIT entry if
1F87 B7       02970          OR      A               ;  unused
1F88 2808     02980          JR      Z,OPEN5
1F8A E5       02990          PUSH    HL              ;Not vacant
1F8B 21E022   03000          LD      HL,FILEHASH     ;Point to DEC
1F8E BE       03010          CP      (HL)            ;Compare with HIT entry
1F8F E1       03020          POP     HL
1F90 2821     03030          JR      Z,OPEN9         ;Jump if a match else
1F92 2C       03040 OPEN5    INC     L               ;  bump to next entry
1F93 20F1     03050          JR      NZ,OPEN4        ;Loop until 256 bytes
              03060 ;
              03070 ;        File not on this drive
              03080 ;
1F95 CD9E1F   03090 OPEN6    CALL    TESTDRV         ;Bump drive if we can
1F98 38E3     03100          JR      C,OPEN3         ;Loop if another to test
1F9A 3E18     03110 OPEN7    LD      A,24            ;File not found error
```

```
1F9C B7       03120          OR      A
1F9D C9       03130          RET
1F9E 3A771F   03140 TESTDRV  LD      A,(OPEN2+1)     ;If drive still X'FF',
1FA1 3C       03150          INC     A               ;  then advance to next
1FA2 B7       03160          OR      A               ;Reset Carry for ret w/o
1FA3 C0       03170          RET     NZ              ;  affecting Z/NZ result
1FA4 0C       03180          INC     C               ;Bump drive counter
1FA5 79       03190          LD      A,C
1FA6 FE08     03200          CP      8               ;Loop end, 8 max
1FA8 C9       03210          RET
              03220 ;
              03230 ;        Although the HIT entry matched, the filename/ext
              03240 ;        did not (due to a collision). Continue to scan
              03250 ;        the rest of the hash index table.
              03260 ;
1FA9 C1       03270 OPEN8    POP     BC              ;Remove ret address and
1FAA E1       03280          POP     HL              ;  excess registers
1FAB C1       03290          POP     BC
1FAC CD9718   03300          CALL    @HITRD          ;Re-read the hit
1FAF E1       03310          POP     HL
1FB0 C0       03320          RET     NZ              ;Go on i/o error
1FB1 18DF     03330          JR      OPEN5
              03340 ;
              03350 ;        The hashed name matches, read the directory
              03360 ;
1FB3 E5       03370 OPEN9    PUSH    HL
1FB4 C5       03380          PUSH    BC
1FB5 45       03390          LD      B,L             ;Set up the DEC
1FB6 CDBB18   03400          CALL    @DIRRD
1FB9 2803     03410          JR      Z,OPEN10        ;Jump if no error
1FBB C1       03420          POP     BC              ;  else pop returns
1FBC E1       03430          POP     HL
1FBD C9       03440          RET                     ;  & exit
              03450 ;
              03460 ;        Verify that directory entry is this file
              03470 ;
1FBE E5       03480 OPEN10   PUSH    HL
1FBF C5       03490          PUSH    BC              ;Save drive (reg C)
              03500 ;
              03510 ;        If bit 7 is set, it denotes an extended
              03520 ;        directory entry which does not include
              03530 ;        the filename. Go to next HIT entry if set
              03540 ;
1FC0 CB7E     03550          BIT     7,(HL)          ;Test for FXDE
1FC2 20E5     03560          JR      NZ,OPEN8        ;Jump if extended
1FC4 CB66     03570          BIT     4,(HL)          ;If DIR record spare,
1FC6 28E1     03580          JR      Z,OPEN8         ;  continue to search
1FC8 3E05     03590          LD      A,5             ;Point to filename/ext
1FCA 85       03600          ADD     A,L             ;  field in directory
1FCB 6F       03610          LD      L,A
1FCC 11E922   03620          LD      DE,NAME$EXT     ;Point to entered name
1FCF 060B     03630          LD      B,11            ;Init to check 11 chars
1FD1 1A       03640 OPEN11   LD      A,(DE)          ;Verify a match
1FD2 BE       03650          CP      (HL)            ;  or no match
1FD3 20D4     03660          JR      NZ,OPEN8        ;Go to next HIT entry
1FD5 23       03670          INC     HL              ;  if no match; else bump
1FD6 13       03680          INC     DE              ;  pointers & loop
1FD7 10F8     03690          DJNZ    OPEN11
1FD9 C1       03700          POP     BC              ;Matches! get drive #
1FDA 79       03710          LD      A,C             ;  & stuff it
1FDB 32771F   03720          LD      (OPEN2+1),A
```

```
1FDE E1        03730          POP     HL
1FDF F1        03740          POP     AF
1FE0 F1        03750          POP     AF
1FE1 C5        03760          PUSH    BC              ;Save DEC and drive
1FE2 E5        03770          PUSH    HL              ;Save ptr to dir record
1FE3 7E        03780          LD      A,(HL)          ;P/u 1st byte of dir rec
1FE4 32FA22    03790          LD      (DIR$INIT),A    ;Stuff it
1FE7 E607      03800          AND     7               ;Strip all but protection
1FE9 4F        03810          LD      C,A
1FEA 0600      03820          LD      B,0
1FEC 3E10      03830          LD      A,16            ;Point to update password
1FEE 85        03840          ADD     A,L
1FEF 6F        03850          LD      L,A
1FF0 ED5BF622  03860          LD      DE,(PW$HASH2)   ;P/u password hash
1FF4 7E        03870          LD      A,(HL)          ;P/u owner pswd lo order
1FF5 23        03880          INC     HL
1FF6 E5        03890          PUSH    HL
1FF7 66        03900          LD      H,(HL)          ;P/u owner pswd hi order
1FF8 6F        03910          LD      L,A
1FF9 3A7700    03920          LD      A,(NFLAG$)      ;P/u NFLAG$
1FFC CB7F      03930          BIT     7,A             ;Check network active bit
1FFE 2802      03940          JR      Z,USEPWD
2000 54        03950          LD      D,H
2001 5D        03960          LD      E,L
2002 AF        03970 USEPWD   XOR     A               ;Compare password entry
2003 ED52      03980          SBC     HL,DE           ;  with owner password
2005 E1        03990          POP     HL
2006 282D      04000 WASMAT   JR      Z,OPEN16        ;Grant access if match
2008 79        04010          LD      A,C             ;Recover protection
2009 FE07      04020          CP      7               ;Abort if "no access"
200B 280B      04030          JR      Z,OPEN12
200D 23        04040          INC     HL              ;  else point to user
200E 41        04050          LD      B,C             ;  password & xfer prot
200F 7E        04060          LD      A,(HL)          ;P/u user pswd lo order
2010 23        04070          INC     HL
2011 66        04080          LD      H,(HL)          ;P/u user pswd hi order
2012 6F        04090          LD      L,A
2013 AF        04100          XOR     A               ;Check for a match
2014 ED52      04110          SBC     HL,DE
2016 2806      04120          JR      Z,OPEN13        ;Jump if match
               04130 ;
               04140 ;        File is password protected - abort
               04150 ;
2018 E1        04160 OPEN12   POP     HL
2019 C1        04170          POP     BC
201A 3E19      04180          LD      A,25            ;"file access denied"
201C B7        04190          OR      A
201D C9        04200          RET
               04210 ;
               04220 ;        Check if prot is exec only
               04230 ;
201E 79        04240 OPEN13   LD      A,C
201F FE06      04250          CP      6               ;Check for EXEC ONLY
2021 2012      04260          JR      NZ,OPEN16       ;Jump if not
2023 0600      04270 OPEN14   LD      B,0             ;P/u SFLAG$ entry state
2025 CB50      04280          BIT     2,B             ;Did RUN request open?
2027 2807      04290          JR      Z,OPEN15        ;Bypass if NOT from RUN
2029 217C00    04300          LD      HL,SFLAG$
202C CBCE      04310          SET     1,(HL)          ;Show RUN & EXEC file
202E 3E05      04320          LD      A,5             ;Set read access for now
2030 21791A    04330 OPEN15   LD      HL,SET@EXEC     ;Set RST vector to turn
```

```
2Ø33 36C9      Ø434Ø          LD      (HL),ØC9H       ;  off DEBUG
2Ø35 32A32Ø    Ø435Ø OPEN16   LD      (OPNCB1+1),A    ;Stuff access level
2Ø38 E1        Ø436Ø          POP     HL              ;Ptr to direc record
2Ø39 C1        Ø437Ø          POP     BC              ;P/u DEC and drive
               Ø438Ø ;
               Ø439Ø ;        Routine to open up the fcb from the directory
               Ø44ØØ ;        HL => directory record in SBUFF$
               Ø441Ø ;        BC => DEC and drive used for directory read/write
               Ø442Ø ;        IX => pointer to File Control Block
               Ø443Ø ;
2Ø3A FDE5      Ø444Ø OPNCB    PUSH    IY              ;Save IY
2Ø3C E5        Ø445Ø          PUSH    HL              ;Transfer direc record
2Ø3D FDE1      Ø446Ø          POP     IY              ;  ptr to IY
2Ø3F C5        Ø447Ø          PUSH    BC              ;Save DEC and drive
2Ø4Ø CD932Ø    Ø448Ø          CALL    OPNCBØ          ;Create the opened FCB
2Ø43 C1        Ø449Ø          POP     BC
2Ø44 21242Ø    Ø45ØØ          LD      HL,OPEN14+1     ;If from LOAD, don't do
2Ø47 CB46      Ø451Ø          BIT     Ø,(HL)          ;  any further checks
2Ø49 28Ø4      Ø452Ø          JR      Z,OPNEX1
2Ø4B AF        Ø453Ø          XOR     A
2Ø4C FDE1      Ø454Ø OPNEX    POP     IY
2Ø4E C9        Ø455Ø          RET
2Ø4F FDCBØ16E  Ø456Ø OPNEX1   BIT     5,(IY+1)        ;If file already open
2Ø53 28ØF      Ø457Ø          JR      Z,OPNCB8        ;  then set read-only
2Ø55 FDE1      Ø458Ø          POP     IY              ;  & return "file open...
2Ø57 DD7EØ1    Ø459Ø OPNEX2   LD      A,(IX+1)        ;P/u current attributes
2Ø5A E6F8      Ø46ØØ          AND     ØF8H            ;Mask off current prot
2Ø5C F6Ø5      Ø461Ø          OR      5               ;  & replace with READ
2Ø5E DD77Ø1    Ø462Ø          LD      (IX+1),A        ;Reset access to READ
2Ø61 3E29      Ø463Ø          LD      A,41            ;Set "file already open"
2Ø63 C9        Ø464Ø          RET
               Ø465Ø ;
               Ø466Ø ;        If access level is > read, set file open flag in
               Ø467Ø ;        the directory & note close authority in the FCB.
               Ø468Ø ;
2Ø64 DD7EØ1    Ø469Ø OPNCB8   LD      A,(IX+1)        ;P/u FCB access level
2Ø67 E6Ø7      Ø47ØØ          AND     7               ;Mask off other junk
2Ø69 FEØ5      Ø471Ø          CP      5               ;Ck READ, EXEC, NONE
2Ø6B 3Ø17      Ø472Ø          JR      NC,OPNCB1Ø      ;Go if one of the above
2Ø6D 3EØØ      Ø473Ø OPNCB9   LD      A,Ø             ;P/u CKDRV status
2Ø6F Ø7        Ø474Ø          RLCA                    ;Was drive write prot?
2Ø7Ø 381C      Ø475Ø          JR      C,FRCREAD       ;CF = WP
2Ø72 FDCBØ1EE  Ø476Ø          SET     5,(IY+1)        ;Set file open in direc
2Ø76 3A77ØØ    Ø477Ø          LD      A,(NFLAG$)      ;P/u Nflag
2Ø79 CB47      Ø478Ø          BIT     Ø,A             ;Check for function ON
2Ø7B C4Ø318    Ø479Ø          CALL    NZ,@DIRWR       ;Write the directory
2Ø7E 2ØCC      Ø48ØØ          JR      NZ,OPNEX
2Ø8Ø DDCBØØF6  Ø481Ø          SET     6,(IX+Ø)        ;Set close authority
               Ø482Ø ;
               Ø483Ø ;        Ck if passed LRL matches directory
               Ø484Ø ;
2Ø84 DD7EØ9    Ø485Ø OPNCB1Ø  LD      A,(IX+9)        ;P/u LRL from FCB &
2Ø87 FDBEØ4    Ø486Ø          CP      (IY+4)          ;  compare with directory
2Ø8A 3E2A      Ø487Ø          LD      A,42            ;Init "LRL open fault
2Ø8C 18BE      Ø488Ø          JR      OPNEX
               Ø489Ø ;
               Ø49ØØ ;        Disk write protected - Change access to READ
               Ø491Ø ;
2Ø8E CD572Ø    Ø492Ø FRCREAD  CALL    OPNEX2          ;Change access to READ
2Ø91 18F1      Ø493Ø          JR      OPNCB1Ø
               Ø494Ø ;
```

```
                    04950 ;       This routine creates the open file control block
                    04960 ;
2093 EB             04970 OPNCBØ  EX      DE,HL
2094 DDE5           04980         PUSH    IX              ;Transfer fcb pointer
2096 E1             04990         POP     HL
2097 1A             05000         LD      A,(DE)          ;Get DIR+Ø
2098 E62Ø           05010         AND     2ØH             ;Keep "PDS" bit & show
209A F68Ø           05020         OR      8ØH             ;  FCB as open
209C 77             05030         LD      (HL),A          ;Shove into FCB+Ø
209D 23             05040         INC     HL
209E 3AFE22         05050         LD      A,(LREC$)       ;P/u lrl
20A1 B7             05060         OR      A               ;Test for Ø (256)
20A2 3EØØ           05070 OPNCB1  LD      A,Ø             ;Now start byte 2 with
20A4 28Ø2           05080         JR      Z,OPNCB2        ;  that set by "OPEN16"
20A6 F68Ø           05090         OR      8ØH             ;Show sector or byte I/O
20A8 F62Ø           05100 OPNCB2  OR      2ØH             ;Show buffer is empty
                    05110 ;
                    05120 ;       Set bit 3 if filespec ended in an
                    05130 ;       exclamation point. This causes the
                    05140 ;       directory to be updated on every
                    05150 ;       file write where the EOF is extended
                    05160 ;
20AA F6ØØ           05170 OPNCB3  OR      Ø
20AC 77             05180         LD      (HL),A          ;Init FCB+1
20AD 23             05190         INC     HL
20AE AF             05200         XOR     A
20AF 77             05210         LD      (HL),A          ;Init FCB+2 with Ø
20BØ 23             05220         INC     HL
20B1 D5             05230         PUSH    DE              ;Put address of disk I/O
20B2 11ØØØØ         05240 OPNCB4  LD      DE,Ø            ;  buf into FCB+3 & FCB+4
20B5 73             05250         LD      (HL),E
20B6 23             05260         INC     HL
20B7 72             05270         LD      (HL),D
20B8 23             05280         INC     HL
20B9 D1             05290         POP     DE              ;FCB+5 with Ø for
20BA 77             05300         LD      (HL),A          ;  lo order next
20BB 23             05310         INC     HL
20BC 71             05320         LD      (HL),C          ;FCB+6 with drive
20BD 23             05330         INC     HL
20BE 7Ø             05340         LD      (HL),B          ;FCB+7 with DEC
20BF 23             05350         INC     HL
20CØ 13             05360         INC     DE              ;Point to DIR EOF byte
20C1 13             05370         INC     DE
20C2 13             05380         INC     DE
20C3 1A             05390         LD      A,(DE)          ;P/u DIR lo order EOF
20C4 77             05400         LD      (HL),A          ;  & stuff into FCB+8
20C5 23             05410         INC     HL
20C6 13             05420         INC     DE
20C7 3AFE22         05430         LD      A,(LREC$)       ;P/u lrl & stuff
20CA 77             05440         LD      (HL),A          ;  into FCB+9
20CB 23             05450         INC     HL
20CC AF             05460         XOR     A
20CD 77             05470         LD      (HL),A          ;Init FCB+1Ø & FCB+11
20CE 23             05480         INC     HL              ;  with zero for NRN
20CF 77             05490         LD      (HL),A
20DØ 23             05500         INC     HL
20D1 CBE3           05510         SET     4,E             ;Point to file EOF
20D3 Ø1Ø2ØØ         05520         LD      BC,2            ;Move ERN
20D6 EB             05530         EX      DE,HL
20D7 EDBØ           05540         LDIR                    ;  and zero BC reg
20D9 EB             05550         EX      DE,HL
```

```
2ØDA 3EØ5      Ø556Ø          LD      A,5             ;Max 5 extents
2ØDC F5        Ø557Ø          PUSH    AF
2ØDD 1A        Ø558Ø OPNCB5   LD      A,(DE)          ;Move starting track
2ØDE 77        Ø559Ø          LD      (HL),A
2ØDF 23        Ø56ØØ          INC     HL
2ØEØ 13        Ø561Ø          INC     DE
2ØE1 1A        Ø562Ø          LD      A,(DE)          ;Move grans & offset
2ØE2 77        Ø563Ø          LD      (HL),A
2ØE3 23        Ø564Ø          INC     HL
2ØE4 E61F      Ø565Ø          AND     1FH             ;Strip out grans
2ØE6 3C        Ø566Ø          INC     A               ;Bump for zero offset
               Ø567Ø ;
               Ø568Ø ;        Add reg A to reg pair BC
               Ø569Ø ;
2ØE7 81        Ø57ØØ          ADD     A,C             ;Add previous count
2ØE8 4F        Ø571Ø          LD      C,A             ;Update C
2ØE9 3ØØ1      Ø572Ø          JR      NC,$+3          ;Go if no carry to B
2ØEB Ø4        Ø573Ø          INC     B
2ØEC F1        Ø574Ø          POP     AF              ;Recover counter
2ØED 3D        Ø575Ø          DEC     A               ;Decrement loop
2ØEE C8        Ø576Ø          RET     Z               ;Done if moved in 5
2ØEF F5        Ø577Ø          PUSH    AF
2ØFØ 13        Ø578Ø          INC     DE
2ØF1 1A        Ø579Ø          LD      A,(DE)          ;Test for end of extents
2ØF2 FEFE      Ø58ØØ          CP      ØFEH            ;Extent in use?
2ØF4 3ØØ6      Ø581Ø          JR      NC,OPNCB6       ;Jump if not
2ØF6 71        Ø582Ø          LD      (HL),C          ;Stuff # of cumulative
2ØF7 23        Ø583Ø          INC     HL              ;  grans to this
2ØF8 7Ø        Ø584Ø          LD      (HL),B          ;  allocation into FCB
2ØF9 23        Ø585Ø          INC     HL
2ØFA 18E1      Ø586Ø          JR      OPNCB5          ;Loop for next
               Ø587Ø ;
               Ø588Ø ;        Unused extents - put X'FFFF' in remaining fields
               Ø589Ø ;
2ØFC F1        Ø59ØØ OPNCB6   POP     AF              ;Recover counter
2ØFD Ø7        Ø591Ø          RLCA                    ;Make times 4 and
2ØFE Ø7        Ø592Ø          RLCA                    ;  fill remaining
2ØFF 47        Ø593Ø          LD      B,A             ;  extent bytes with
21ØØ 36FF      Ø594Ø OPNCB7   LD      (HL),ØFFH       ;  ØFFH
2102 23        Ø595Ø          INC     HL
2103 1ØFB      Ø596Ø          DJNZ    OPNCB7
2105 C9        Ø597Ø          RET
               Ø598Ø ;
               Ø599Ø ;        INIT a file
               Ø6ØØØ ;         HL => the address of a 256-byte buffer
               Ø6Ø1Ø ;         DE => the address of a 32-byte FCB
               Ø6Ø2Ø ;          B => the logical record length (LREC)
               Ø6Ø3Ø ;
2106 CD6615    Ø6Ø4Ø INIT     CALL    LNKFCB@         ;Link to FCB
2109 32A32Ø    Ø6Ø5Ø          LD      (OPNCB1+1),A    ;Start FCB+1 with Ø
21ØC E5        Ø6Ø6Ø          PUSH    HL
21ØD 217CØØ    Ø6Ø7Ø          LD      HL,SFLAG$       ;Reset called by RUN bit
2110 CB96      Ø6Ø8Ø          RES     2,(HL)
2112 E1        Ø6Ø9Ø          POP     HL
2113 CD441F    Ø61ØØ          CALL    OPEN1           ;Can we "OPEN" the file?
2116 C8        Ø611Ø          RET     Z               ;Return if file existing
2117 FE18      Ø612Ø          CP      24              ;Return if error not
2119 CØ        Ø613Ø          RET     NZ              ;  "file not found"
211A 3E1Ø      Ø614Ø          LD      A,1ØH           ;Set dir rec to show
211C 32FA22    Ø615Ø          LD      (DIR$INIT),A    ;  assigned
211F 3A771F    Ø616Ø          LD      A,(OPEN2+1)     ;P/u the drive entry
```

```
2122 4F       Ø617Ø         LD      C,A
2123 3C       Ø618Ø         INC     A               ;Jump if a drive entry
2124 F5       Ø619Ø         PUSH    AF
2125 2ØØ1     Ø62ØØ         JR      NZ,INIT1        ;  was made
2127 4F       Ø621Ø         LD      C,A
2128 F1       Ø622Ø INIT1   POP     AF              ;Stack integrity
2129 CD7A1E   Ø623Ø         CALL    CKDRV           ;Is this drive available?
212C 2ØØC     Ø624Ø         JR      NZ,INIT2        ;Jump if not
212E 38ØA     Ø625Ø         JR      C,INIT2         ;  or if write protected
213Ø CD9718   Ø626Ø         CALL    @HITRD          ;Read hash index table
2133 CØ       Ø627Ø         RET     NZ              ;Return if read error
2134 CDEE21   Ø628Ø         CALL    SPRHIT          ;Locate spare entry
2137 281F     Ø629Ø         JR      Z,INIT4         ;Jump if space
2139 AF       Ø63ØØ         XOR     A               ;Set status of CKDRV=Z
213A F5       Ø631Ø INIT2   PUSH    AF              ;Save last CKDRV status
213B CD9E1F   Ø632Ø         CALL    TESTDRV
213E 38E8     Ø633Ø         JR      C,INIT1         ;Loop if not at end
214Ø 3A771F   Ø634Ø         LD      A,(OPEN2+1)     ;If drivespec not entered
2143 3C       Ø635Ø         INC     A               ;  then "directory full
2144 2ØØ3     Ø636Ø         JR      NZ,INIT2A
2146 F1       Ø637Ø         POP     AF              ;Stack integrity
2147 18Ø5     Ø638Ø         JR      ERR26
2149 F1       Ø639Ø INIT2A  POP     AF              ;  else if no drive then
214A 2ØØ8     Ø64ØØ         JR      NZ,ERR32        ;  "illegal drive...
214C 38Ø3     Ø641Ø         JR      C,ERR15         ;  else if write protect
214E 3E1A     Ø642Ø ERR26   LD      A,26            ;  "directory space full"
215Ø Ø1       Ø643Ø         DB      1
2151 3EØF     Ø644Ø ERR15   LD      A,15            ;"write protect...
2153 Ø1       Ø645Ø         DB      1
2154 3E2Ø     Ø646Ø ERR32   LD      A,32            ;"Illegal drive...
2156 B7       Ø647Ø         OR      A
2157 C9       Ø648Ø         RET
              Ø649Ø ;
              Ø65ØØ ;       Found a spare HIT entry position
              Ø651Ø ;
2158 45       Ø652Ø INIT4   LD      B,L             ;Save DEC
2159 3AEØ22   Ø653Ø         LD      A,(FILEHASH)    ;P/u filespec hash
215C 77       Ø654Ø         LD      (HL),A          ;  & store in HIT
215D CD9818   Ø655Ø         CALL    @HITWR          ;Write updated HIT
216Ø CCBB18   Ø656Ø         CALL    Z,@DIRRD        ;Read that dir record
2163 CØ       Ø657Ø         RET     NZ              ;Return if read error
2164 E5       Ø658Ø         PUSH    HL
2165 C5       Ø659Ø         PUSH    BC
2166 EB       Ø66ØØ         EX      DE,HL
2167 Ø1Ø5ØØ   Ø661Ø         LD      BC,5            ;Move 1st 5 bytes into
216A 21FA22   Ø662Ø         LD      HL,DIR$INIT     ;  directory record
216D EDBØ     Ø663Ø         LDIR
216F ØE11     Ø664Ø         LD      C,17            ;Move filename & password
2171 21E922   Ø665Ø         LD      HL,NAME$EXT     ;  info into directory
2174 EDBØ     Ø666Ø         LDIR
2176 EB       Ø667Ø         EX      DE,HL
2177 Ø6ØA     Ø668Ø         LD      B,1Ø            ;Put X'FFFF' into 5 ext's
2179 CDØØ21   Ø669Ø INIT5   CALL    OPNCB7          ;4 for the ext's & 1 for
217C C1       Ø67ØØ         POP     BC              ;  starting info
217D CDØ318   Ø671Ø         CALL    @DIRWR          ;Write updated directory
218Ø E1       Ø672Ø         POP     HL
2181 CØ       Ø673Ø         RET     NZ              ;Return if write error
2182 CD3A2Ø   Ø674Ø         CALL    OPNCB           ;  else open the fcb
2185 37       Ø675Ø         SCF                     ;Indicate new file
2186 C9       Ø676Ø         RET
              Ø677Ø ;
```

```
                    Ø678Ø ;       Xfer the file spec to system buffer area
                    Ø679Ø ;
2187 Ø613           Ø68ØØ XFRSPEC LD      B,19
2189 11E122         Ø681Ø         LD      DE,PSWDBUF
218C 3E2Ø           Ø682Ø         LD      A,2ØH           ;Blank out the filename
218E 12             Ø683Ø XSPEC1  LD      (DE),A          ;  field in system buffer
218F 13             Ø684Ø         INC     DE
219Ø 1ØFC           Ø685Ø         DJNZ    XSPEC1
2192 3EFF           Ø686Ø         LD      A,ØFFH          ;Set drive to X'FF' for
2194 32771F         Ø687Ø         LD      (OPEN2+1),A     ;  checking user entry
2197 1EE9           Ø688Ø         LD      E,NAME$EXT&ØFFH ;Xfer file name
2199 CDD221         Ø689Ø         CALL    XSPEC8
219C 4F             Ø69ØØ         LD      C,A
219D 78             Ø691Ø         LD      A,B
219E D6Ø8           Ø692Ø         SUB     8               ;Any valid chars found?
21AØ 2ØØ3           Ø693Ø         JR      NZ,XSPEC3       ;Jump if valid name
                    Ø694Ø ;
                    Ø695Ø ;       Filename was invalid format
                    Ø696Ø ;
21A2 F613           Ø697Ø         OR      19              ;"illegal file name"
21A4 C9             Ø698Ø         RET
                    Ø699Ø ;
                    Ø7ØØØ ;       Continue to check file spec
                    Ø7Ø1Ø ;
21A5 79             Ø7Ø2Ø XSPEC3  LD      A,C
21A6 FE2F           Ø7Ø3Ø         CP      '/'             ;Ext entered?
21A8 1EF1           Ø7Ø4Ø         LD      E,FILE$EXT&ØFFH
21AA Ø6Ø3           Ø7Ø5Ø         LD      B,3
21AC CCD421         Ø7Ø6Ø         CALL    Z,XSPEC8A       ;Xfer the ext
21AF FE2E           Ø7Ø7Ø         CP      '.'             ;Password entered?
21B1 1EE1           Ø7Ø8Ø         LD      E,PSWDBUF&ØFFH
21B3 CCD221         Ø7Ø9Ø         CALL    Z,XSPEC8        ;Xfer the password
21B6 FE3A           Ø71ØØ         CP      ':'             ;Drive entered?
21B8 2ØØD           Ø711Ø         JR      NZ,XSPEC6
21BA 7E             Ø712Ø         LD      A,(HL)          ;P/u drive #
21BB D63Ø           Ø713Ø         SUB     'Ø'             ;Convert to binary
21BD 32771F         Ø714Ø         LD      (OPEN2+1),A     ;Stuff drive #
21CØ E6F8           Ø715Ø         AND     ØF8H            ;Must be <Ø-7>
21C2 3E2Ø           Ø716Ø         LD      A,32            ;"illegal drive #"
21C4 CØ             Ø717Ø         RET     NZ              ;Return error if out
21C5 23             Ø718Ø         INC     HL              ;  of range
21C6 7E             Ø719Ø         LD      A,(HL)          ;Does filespec end in
21C7 D621           Ø72ØØ XSPEC6  SUB     21H             ;  exclamation point?
21C9 3EØ8           Ø721Ø         LD      A,8             ;Init to set bit 3 of
21CB 28Ø1           Ø722Ø         JR      Z,XSPEC7        ;  FCB+1 & jump if "!"
21CD AF             Ø723Ø         XOR     A               ;  else reset if not
21CE 32AB2Ø         Ø724Ø XSPEC7  LD      (OPNCB3+1),A
21D1 C9             Ø725Ø         RET
                    Ø726Ø ;
                    Ø727Ø ;
                    Ø728Ø ;
21D2 Ø6Ø8           Ø729Ø XSPEC8  LD      B,8
21D4 7E             Ø73ØØ XSPEC8A LD      A,(HL)          ;P/u a filespec character
21D5 23             Ø731Ø         INC     HL              ;  & 1st test for A-Z
21D6 18Ø9           Ø732Ø         JR      XSPEC1Ø
21D8 7E             Ø733Ø XSPEC9  LD      A,(HL)          ;P/u a filespec character
21D9 23             Ø734Ø         INC     HL              ;Advance to next one
21DA FE3Ø           Ø735Ø         CP      'Ø'             ;Check for Ø-9
21DC D8             Ø736Ø         RET     C
21DD FE3A           Ø737Ø         CP      '9'+1
21DF 38Ø6           Ø738Ø         JR      C,XSPEC11
```

```
21E1 FE41    07390 XSPEC10 CP      'A'             ;Check for A-Z
21E3 D8      07400         RET     C
21E4 FE5B    07410         CP      'Z'+1
21E6 DØ      07420         RET     NC
21E7 12      07430 XSPEC11 LD      (DE),A          ;Character is valid
21E8 13      07440         INC     DE              ;Advance to next one
21E9 1ØED    07450         DJNZ    XSPEC9          ;  & loop
21EB 7E      07460         LD      A,(HL)          ;P/u following character
21EC 23      07470         INC     HL
21ED C9      07480         RET
             07490 ;
             07500 ;       Routine to find a spare HIT entry
             07510 ;       Calculate the number of directory sectors
             07520 ;       = (#sectors x #heads) - 2 for GAT & HIT
             07530 ;
             07540 SPRHIT
21EE 3EØ7    07550         LD      A,7             ;Get highest # sector
21FØ CD2B1A  07560         CALL    @DCTBYT
21F3 D5      07570         PUSH    DE
21F4 57      07580         LD      D,A             ;Store heads & sectors
21F5 E61F    07590         AND     1FH             ;Rake off # sectors
21F7 5F      07600         LD      E,A             ;  & stuff into E
21F8 1C      07610         INC     E               ;Bump for Ø offset
21F9 AA      07620         XOR     D               ;Recover # heads
21FA Ø7      07630         RLCA                    ;  into bits Ø-2
21FB Ø7      07640         RLCA
21FC Ø7      07650         RLCA
21FD 3C      07660         INC     A               ;Bump for Ø offset
21FE CDØA19  07670         CALL    @MUL8           ;Multiply sectors x heads
22Ø1 5F      07680         LD      E,A             ;Now check double bit
22Ø2 3EØ4    07690         LD      A,4
22Ø4 CD2B1A  07700         CALL    @DCTBYT
22Ø7 CB6F    07710         BIT     5,A             ;Set if 2-sided
22Ø9 7B      07720         LD      A,E
22ØA 28Ø1    07730         JR      Z,ONESID        ;Go if not set else
22ØC 87      07740         ADD     A,A             ;  double value
22ØD D1      07750 ONESID  POP     DE
22ØE D6Ø2    07760         SUB     2               ;Reduce for GAT & HIT
2210 322422  07770         LD      (GSH3+1),A      ;Stuff for compare
             07780 ;
             07790 ;       Search across rows
             07800 ;
2213 2E27    07810         LD      L,27H           ;Try first to use a HIT
2215 CD1B22  07820         CALL    GSHLOOP         ;  past the SYS slots
2218 C8      07830         RET     Z               ;Return if spare found
             07840 ;
2219 2EØ1    07850         LD      L,1             ;Start after DIR slot
221B 2C      07860 GSHLOOP INC     L               ;Step to next
221C 2ØØ2    07870         JR      NZ,GSHTRY       ;Go if not done yet
221E B4      07880         OR      H               ;Set NZ flag
221F C9      07890         RET                     ;Return failure
2220 7D      07900 GSHTRY  LD      A,L             ;Skip unused parts
2221 E61F    07910         AND     1FH
2223 FEØØ    07920 GSH3    CP      Ø               ;Cp with # of dir sectors
2225 7D      07930         LD      A,L
2226 38Ø5    07940         JR      C,GSHOK         ;Go if NOT unused
2228 F61F    07950         OR      1FH             ;Force to end of row
222A 6F      07960         LD      L,A
222B 18EE    07970         JR      GSHLOOP         ;Loop back & ck for end
222D 7E      07980 GSHOK   LD      A,(HL)          ;P/u HIT byte
222E B7      07990         OR      A               ;Free?
```

```
222F C8       Ø8ØØØ          RET     Z               ;Done if so
2230 18E9     Ø8Ø1Ø          JR      GSHLOOP         ;Try next
              Ø8Ø2Ø ;
              Ø8Ø3Ø ;        Routine to rename a filespec/devspec
              Ø8Ø4Ø ;
2232 3E18     Ø8Ø5Ø RENØ     LD      A,18H
2234 32Ø62Ø   Ø8Ø6Ø           LD      (WASMAT),A
2237 B7       Ø8Ø7Ø           OR      A               ;Denote "file not in dir
2238 C9       Ø8Ø8Ø           RET
2239 CD6615   Ø8Ø9Ø RENAME    CALL    LNKFCB@         ;Save regs & link to IX
223C DD7EØØ   Ø81ØØ           LD      A,(IX+Ø)        ;If a device, use the
223F D62A     Ø811Ø           SUB     '*'             ;  "device" routine
2241 2879     Ø812Ø           JR      Z,RENDEV
2243 FEA8     Ø813Ø           CP      'R'!8ØH-'*'     ;Special open condition?
2245 28EB     Ø814Ø           JR      Z,RENØ          ;Go if so
2247 E5       Ø815Ø           PUSH    HL              ;Save new pointer
2248 217CØØ   Ø816Ø           LD      HL,SFLAG$       ;Set don't test flags
224B CBC6     Ø817Ø           SET     Ø,(HL)
224D CD441F   Ø818Ø           CALL    OPEN1           ;Open the "old" spec
225Ø E1       Ø819Ø           POP     HL
2251 CØ       Ø82ØØ           RET     NZ              ;Exit on error
2252 DD7EØ1   Ø821Ø           LD      A,(IX+1)        ;Make sure user has
2255 E6Ø7     Ø822Ø           AND     7               ;  permission to rename
2257 FEØ3     Ø823Ø           CP      3
2259 38Ø4     Ø824Ø           JR      C,REN1
225B 3E25     Ø825Ø           LD      A,25H           ;"Illegal access...
225D B7       Ø826Ø           OR      A
225E C9       Ø827Ø           RET
              Ø828Ø ;
              Ø829Ø ;        User has access to rename - locate drivespec
              Ø83ØØ ;
225F E5       Ø831Ø REN1      PUSH    HL              ;Save start
2260 7E       Ø832Ø REN2      LD      A,(HL)          ;P/u char of new spec
2261 23       Ø833Ø           INC     HL
2262 FEØD     Ø834Ø           CP      CR
2264 28Ø8     Ø835Ø           JR      Z,REN3          ;Go on ENTER
2266 FEØ3     Ø836Ø           CP      3
2268 28Ø4     Ø837Ø           JR      Z,REN3          ;Go on ETX
226A FE3A     Ø838Ø           CP      ':'
226C 2ØF2     Ø839Ø           JR      NZ,REN2         ;Loop on colon
226E 2B       Ø84ØØ REN3      DEC     HL              ;Backup to where the
226F 363A     Ø841Ø           LD      (HL),':'        ;  colon should go
2271 23       Ø842Ø           INC     HL              ;  & force the drivespec
2272 DD7EØ6   Ø843Ø           LD      A,(IX+6)        ;  to the same as "old"
2275 4F       Ø844Ø           LD      C,A             ;Keep drive spec in C
2276 E6Ø7     Ø845Ø           AND     7
2278 C63Ø     Ø846Ø           ADD     A,'Ø'           ;Make it an ASCII digit
227A 77       Ø847Ø           LD      (HL),A
227B 23       Ø848Ø           INC     HL
227C 36ØD     Ø849Ø           LD      (HL),CR
227E DD46Ø7   Ø85ØØ           LD      B,(IX+7)        ;Get DEC
2281 DDE1     Ø851Ø           POP     IX              ;Put "new" FCB into IX
2283 C5       Ø852Ø           PUSH    BC              ;  & save DEC & drive
2284 217CØØ   Ø853Ø           LD      HL,SFLAG$       ;Set don't test flags
2287 CBC6     Ø854Ø           SET     Ø,(HL)
2289 CD441F   Ø855Ø           CALL    OPEN1           ;Open the "new" spec
228C C1       Ø856Ø           POP     BC
228D 2ØØ4     Ø857Ø           JR      NZ,REN4         ;Should error here
228F 3E13     Ø858Ø REN3A     LD      A,19            ;  or else return
2291 B7       Ø859Ø           OR      A               ;  if "new" is existing
2292 C9       Ø86ØØ           RET                     ;  & we opened it
```

```
2293 FE18     Ø861Ø REN4    CP      24              ;If not "file not found"
2295 CØ       Ø862Ø         RET     NZ              ;  then is error
2296 CDBB18   Ø863Ø         CALL    @DIRRD          ;Read "old's" directory
2299 CØ       Ø864Ø         RET     NZ
229A C5       Ø865Ø         PUSH    BC              ;Save drive spec
229B 54       Ø866Ø         LD      D,H             ;Xfer buffer hi
229C 7D       Ø867Ø         LD      A,L
229D C6Ø5     Ø868Ø         ADD     A,5             ;Pt to filename field
229F 5F       Ø869Ø         LD      E,A             ;Set buffer lo
22AØ 21E922   Ø87ØØ         LD      HL,NAME$EXT     ;Point to where the
22A3 Ø1ØBØØ   Ø871Ø         LD      BC,11           ;  new name is stored
22A6 EDBØ     Ø872Ø         LDIR                    ;Move in new name
22A8 C1       Ø873Ø         POP     BC
22A9 CDØ318   Ø874Ø         CALL    @DIRWR          ;Rewrite the directory
22AC CC9718   Ø875Ø         CALL    Z,@HITRD        ;Read the HIT
22AF CØ       Ø876Ø         RET     NZ
22BØ 54       Ø877Ø         LD      D,H             ;Set the buffer hi
22B1 58       Ø878Ø         LD      E,B             ;Set the exact HIT lo
22B2 21E922   Ø879Ø         LD      HL,NAME$EXT     ;This doesn't change C
22B5 CD1E1E   Ø88ØØ         CALL    HASHNAME        ;Hash the new name
22B8 12       Ø881Ø         LD      (DE),A          ;Stuff code into HIT
22B9 C39818   Ø882Ø         JP      @HITWR          ;Rewrite & exit
              Ø883Ø ;
              Ø884Ø ;       Routine to rename a device
              Ø885Ø ;
22BC E5       Ø886Ø RENDEV  PUSH    HL              ;Save new pointer
22BD CD591E   Ø887Ø         CALL    GETDCB          ;Locate old in tables
22CØ DDE1     Ø888Ø         POP     IX              ;Recover pointer to "new"
22C2 CØ       Ø889Ø         RET     NZ              ;Back if not in tables
22C3 7D       Ø89ØØ         LD      A,L
22C4 FE31     Ø891Ø         CP      DCBKL$          ;Ck if protected device
22C6 3E28     Ø892Ø         LD      A,4Ø            ;"Protected system device
22C8 D8       Ø893Ø         RET     C
22C9 DD7EØØ   Ø894Ø         LD      A,(IX+Ø)        ;"new" must be a device
22CC FE2A     Ø895Ø         CP      '*'
22CE 2ØBF     Ø896Ø         JR      NZ,REN3A        ;"illegal file name...
22DØ E5       Ø897Ø         PUSH    HL              ;Save address of "old"
22D1 CD591E   Ø898Ø         CALL    GETDCB          ;Ck if "new" is unused
22D4 E1       Ø899Ø         POP     HL              ;Rcvr address of "old"
22D5 28B8     Ø9ØØØ         JR      Z,REN3A
22D7 Ø1Ø6ØØ   Ø9Ø1Ø         LD      BC,6            ;Point to name field
22DA Ø9       Ø9Ø2Ø         ADD     HL,BC           ;  of "old" device
22DB 73       Ø9Ø3Ø         LD      (HL),E          ;Stuff new name into
22DC 23       Ø9Ø4Ø         INC     HL              ;  device control block
22DD 72       Ø9Ø5Ø         LD      (HL),D
22DE AF       Ø9Ø6Ø         XOR     A               ;Set Z-flag
22DF C9       Ø9Ø7Ø         RET
              Ø9Ø8Ø ;
              Ø9Ø9Ø ;       Parameter storage area
              Ø91ØØ ;
ØØØ1          Ø911Ø FILEHASH        DS      1
ØØØ8          Ø912Ø PSWDBUF DS      8
ØØØ8          Ø913Ø NAME$EXT        DS      8
ØØØ3          Ø914Ø FILE$EXT        DS      3
ØØØ2          Ø915Ø PW$HASH1        DS      2
ØØØ2          Ø916Ø PW$HASH2        DS      2
22F8 ØØØØ     Ø917Ø         DW      Ø               ;ERN init
22FA ØØ       Ø918Ø DIR$INIT        DB      Ø,Ø,Ø,Ø
     ØØ ØØ ØØ
ØØØ1          Ø919Ø LREC$   DS      1
22FF          Ø92ØØ LAST    EQU     $
```

```
                Ø921Ø           IFGT    $,DIRBUF$
                Ø922Ø           ERR     'Module too big'
                Ø923Ø           ENDIF
23FE            Ø924Ø           ORG     MAXCOR$-2
23FE FFØ4       Ø925Ø           DW      LAST-SYS2       ;Overlay length
                Ø926Ø ;
1EØØ            Ø927Ø           END     SYS2
```

$A1	Ø3B7	$A2	Ø3B8	$A3	Ø3B9		
$CKEOF	147Ø	@$SYS	Ø8FØ	@@1	ØØØØ		
@@2	ØØØØ	@@3	ØØØØ	@@4	ØØØØ		
@ABORT	1BØ8	@ADTSK	1CDA	@BANK	Ø877		
@BKSP	1486	@BREAK	196F	@BYTEIO	13ØØ		
@CHNIO	Ø689	@CKBRKC	Ø553	@CKDRV	1993		
@CKEOF	158F	@CKTSK	1CF5	@CLOSE	1999		
@CLS	Ø545	@CMNDI	197E	@CMNDR	197B		
@CTL	Ø623	@DATE	Ø7A8	@DBGHK	199F		
@DCINIT	19CØ	@DCRES	19C4	@DCSTAT	19B5		
@DCTBYT	1A2B	@DEBUG	19AØ	@DECHEX	Ø3E1		
@DIRCYL	18F7	@DIRRD	18BB	@DIRWR	18Ø3		
@DIV16	Ø6E3	@DIV8	1927	@DODIR	19AF		
@DOKEY	19A9	@DSP	Ø642	@DSPLY	Ø52D		
@ERROR	1BØF	@EXIT	1BØB	@FEXT	1984		
@FLAGS	196A	@FNAME	199C	@FRENCH	ØØØØ		
@FSPEC	1981	@GATRD	1874	@GATWR	1875		
@GERMAN	ØØØØ	@GET	Ø638	@GTDCB	199Ø		
@GTDCT	1A1E	@GTMOD	19B2	@HDFMT	19E4		
@HEX16	Ø7BD	@HEX8	Ø7C2	@HEXDEC	Ø6F6		
@HIGH$	1948	@HITRD	1897	@HITWR	1898		
@HZ5Ø	ØØØØ	@ICNFG	ØØ86	@INIT	198D		
@INTL	ØØØØ	@IPL	1BF2	@JCL	Ø63Ø		
@KBD	Ø635	@KEY	Ø628	@KEYIN	Ø585		
@KITSK	ØØ89	@KLTSK	1CDØ	@LOAD	1B38		
@LOC	14B3	@LOF	14DE	@LOGER	Ø5Ø3		
@LOGOT	Ø5ØØ	@MOD2	ØØØØ	@MOD4	FFFF		
@MSG	Ø53Ø	@MUL16	Ø6C9	@MUL8	19ØA		
@NMI	ØØ66	@OPEN	198A	@OPREG	ØØ84		
@PARAM	1987	@PAUSE	Ø382	@PEOF	14A2		
@POSN	1434	@PRINT	Ø528	@PRT	Ø63D		
@PUT	Ø645	@RAMDIR	19AC	@RDHDR	19D8		
@RDSEC	19F4	@RDSSC	18D8	@RDTRK	19EØ		
@READ	1513	@REMOVE	19A6	@RENAME	1996		
@REW	149B	@RMTSK	1CD7	@RPTSK	1CEB		
@RREAD	1473	@RSLCT	19D4	@RSTØØ	ØØØØ		
@RSTØ8	ØØØ8	@RST1Ø	ØØ1Ø	@RST18	ØØ18		
@RST2Ø	ØØ2Ø	@RST28	ØØ28	@RST3Ø	ØØ3Ø		
@RST38	ØØ38	@RSTNMI	ØFE9	@RSTOR	19C8		
@RSTREG	Ø68Ø	@RUN	1B1D	@RWRIT	13AD		
@SEEK	19DØ	@SEEKSC	1421	@SKIP	143Ø		
@SLCT	19BC	@SOUND	Ø392	@STEPI	19CC		
@TIME	Ø78D	@USA	FFFF	@VDCTL	ØB99		
@VDCTL3	ØD38	@VER	156Ø	@VRSEC	19DC		
@WEOF	14EC	@WHERE	1979	@WRITE	1531		
@WRSEC	19E8	@WRSSC	19EC	@WRTRK	19FØ		
@_VDCTL	ØD42	ADDR_2_ROWCOL	ØDF1	AFLAG$	ØØ6A		
AUTO?	1FF1	BAR$	Ø2Ø1	BOOTST$	439D		
BREAK?	1C6Ø	BRKVEC$	1C88	BUR$	Ø2ØØ		
CASHK$	ØA7B	CDCNT	1ED4	CFCB$	ØØEØ		
CFGFCB$	ØØEØ	CFLAG$	ØØ6C	CKDR1	1EC9		
CKDR2	1ED6	CKDR2A	1EDF	CKDR2B	1EE4		
CKDR3	1FØ6	CKDR3A	1FØ7	CKDR4	1F11		
CKDR5	1F14	CKDR7	1F22	CKDR7A	1F23		
CKDRV	1E7A	CKDRV1	1E9D	CKDRV1A	1EA7		
CKMOD@	1A7F	CKOPEN@	1568	CONFIG$	2Ø3F		
CORE$	Ø3ØØ	CR	ØØØD	CRTBGN$	F8ØØ		
CYL_GRN	16AE	D@FBYT8	1A26	DATE$	ØØ33		
DAYTBL$	Ø4C7	DBGSV$	ØØAØ	DCBKL$	ØØ31		
DCT$	Ø47Ø	DCTBYT8@	1A29	DCTFLD@	1A34		

DEV1	1E62	DEV2	1E72	DEV4	1F2A
DEVOPEN	1F26	DFLAG$	ØØ6D	DIR$INIT	22FA
DIRBUF$	23ØØ	DIS_DO_RAM	Ø846	DODATA$	ØB94
DODCB$	Ø21Ø	DO_CONTROL	ØC44	DO_DSPCHAR	ØCB8
DO_INVERT_DIS	ØC8C	DO_INVERT_ENA	ØC89	DO_INVERT_OFF	ØC9B
DO_MASK	ØØØØ	DO_RET	ØBCB	DO_RET1	ØBCC
DO_SCROLL	ØCCE	DO_TABS	ØBEA	DSKTYP$	Ø4CØ
DTPMT$	Ø4C2	DVREND$	ØFF4	DVRHI$	Ø2Ø6
EFLAG$	ØØ6E	ENADIS_DO_RAM	Ø817	ERR15	2151
ERR26	214E	ERR32	2154	EXTDBG$	19A4
FDDINT$	ØØØE	FEMSK$	ØØ6F	FILE$EXT	22F1
FILEHASH	22EØ	FLGTAB$	ØØ6A	FRCREAD	2Ø8E
GETDCB	1E59	GET_@_ROWCOL	ØDAE	GSH3	2223
GSHLOOP	221B	GSHOK	222D	GSHTRY	222Ø
GTDCB	1E5F	HASHNAME	1E1E	HASHPSWD	1E2E
HERTZ$	Ø75Ø	HIGH$	Ø4ØE	HKRES$	1A6C
HNAME1	1E21	HNAME2	1E2A	HPSWD1	1E38
IFLAG$	ØØ72	INBUF$	Ø42Ø	INDEX	1F17
INIT	21Ø6	INIT1	2128	INIT2	213A
INIT2A	2149	INIT4	2158	INIT5	2179
INTIM$	ØØ3C	INTMSK$	ØØ3D	INTRON	1EC3
INTVC$	ØØ3E	JCLCB$	Ø2Ø3	JDCB$	ØØ24
JFCB$	ØØCØ	JLDCB$	Ø23Ø	JRET$	ØØ26
KCK@	Ø7D6	KFLAG$	ØØ74	KIDATA$	Ø8FC
KIDCB$	Ø2Ø8	LAST	22FF	LBANK$	Ø2Ø2
LDRV$	ØØ23	LFLAG$	ØØ75	LNKFCB@	1566
LOW$	ØØ1E	LREC$	22FE	LSVC$	ØØØD
MAXCOR$	24ØØ	MAXDAY$	Ø4Ø1	MINCOR$	3ØØØ
MODOUT$	ØØ76	MONTBL$	Ø4DC	NAME$EXT	22E9
NFLAG$	ØØ77	ONESID	22ØD	OPEN	1F41
OPEN1	1F44	OPEN1Ø	1FBE	OPEN11	1FD1
OPEN12	2Ø18	OPEN13	2Ø1E	OPEN14	2Ø23
OPEN15	2Ø3Ø	OPEN16	2Ø35	OPEN2	1F76
OPEN3	1F7D	OPEN4	1F86	OPEN5	1F92
OPEN6	1F95	OPEN7	1F9A	OPEN8	1FA9
OPEN9	1FB3	OPNCB	2Ø3A	OPNCBØ	2Ø93
OPNCB1	2ØA2	OPNCB1Ø	2Ø84	OPNCB2	2ØA8
OPNCB3	2ØAA	OPNCB4	2ØB2	OPNCB5	2ØDD
OPNCB6	2ØFC	OPNCB7	21ØØ	OPNCB8	2Ø64
OPNCB9	2Ø6D	OPNEX	2Ø4C	OPNEX1	2Ø4F
OPNEX2	2Ø57	OPREG$	ØØ78	OPREG_SV_AREA	Ø86E
OPREG_SV_PTR	Ø835	ORARET@	14DC	OSRLS$	ØØ3B
OSVER$	ØØ85	OVRLY$	ØØ69	PAKNAM$	Ø41Ø
PAUSE@	Ø382	PCSAVE$	Ø7AF	PDRV$	ØØ1B
PHIGH$	ØØ1C	PRDCB$	Ø218	PSWDBUF	22E1
PUTA@DE	ØDCD	PUT_@	ØDCA	PUT_@_ROWCOL	ØDC6
PW$HASH1	22F4	PW$HASH2	22F6	RENØ	2232
REN1	225F	REN2	226Ø	REN3	226E
REN3A	228F	REN4	2293	RENAME	2239
RENDEV	22BC	RFLAG$	ØØ7B	ROWCOL_2_ADDR	ØDDØ
RST38@	1BFF	RSTOR$	Ø4C4	RWRIT@	13A2
S1DCB$	Ø238	SBUFF$	1DØØ	SET@EXEC	1A79
SET_SCROLL	ØCF3	SFCB$	ØØ8C	SFLAG$	ØØ7C
SIDCB$	Ø22Ø	SODCB$	Ø228	SPACE4$	2142
SPRHIT	21EE	STACK$	Ø38Ø	START$	ØØØØ
SVCRET$	ØØØB	SVCTAB$	Ø1ØØ	SYS2	1EØØ
SYSERR$	1B13	TCB$	ØØ4E	TESTDRV	1F9E
TFLAG$	ØØ7D	TIME$	ØØ2D	TIMER$	ØØ2C
TIMSL$	ØØ2B	TIMTSK$	Ø713	TMPMT$	Ø4C3
TRACE_INT	Ø7B1	TYPHK$	ØA8F	TYPTSK$	ØB26
USEPWD	2ØØ2	USTOR$	ØØ13	VFLAG$	ØØ7F

```
WASMAT          2ØØ6 WRINT$          ØØ8Ø XFRSPEC         2187
XSPEC1          218E XSPEC1Ø         21E1 XSPEC11         21E7
XSPEC3          21A5 XSPEC6          21C7 XSPEC7          21CE
XSPEC8          21D2 XSPEC8A         21D4 XSPEC9          21D8
ZERO$           Ø4Ø1 ZEROA@          13AØ
ØØØØØ Total errors
```

NOTES:

NOTES:

SYS3 handles closing a file or device, and restoring original filespec or device spec to the Control Block. During a file close, it also de-allocates space if the ending record number is smaller than it was upon open. It contains the code for the SVCs @CLOSE and @FNAME.

```
                ØØ1ØØ ;SYS3/ASM - LS-DOS 6.2
ØØØØ            ØØ11Ø          TITLE   <SYS3 - LS-DOS 6.2>
                ØØ12Ø ;
                ØØ13Ø *LIST    OFF                     ;Get SYSØ/EQU
                ØØ15Ø *LIST    ON
ØØØA            ØØ16Ø LF       EQU     1Ø
ØØØD            ØØ17Ø CR       EQU     13
                ØØ18Ø ;
ØØØØ            ØØ19Ø *GET     COPYCOM:3               ;Copyright message
                Ø3Ø1Ø ; COPYCOM - File for Copyright COMment block
                Ø3Ø2Ø ;
ØØØØ            Ø3Ø3Ø          COM     '<*(C) 1982,83,84 by LSI*>'
                Ø3Ø4Ø ;
                ØØ2ØØ ;
1EØØ            ØØ21Ø          ORG     1EØØH
                ØØ22Ø ;
1EØØ E67Ø       ØØ23Ø SYS3     AND     7ØH
1EØ2 C8         ØØ24Ø          RET     Z               ;Back on zero entry
1EØ3 FE1Ø       ØØ25Ø          CP      1ØH
1EØ5 28Ø6       ØØ26Ø          JR      Z,CLOSE         ;Jump if close
1EØ7 FE2Ø       ØØ27Ø          CP      2ØH
1EØ9 CA242Ø     ØØ28Ø          JP      Z,FNAME         ;Jump if filespec recover
1EØC C9         ØØ29Ø          RET
1EØD 1A         ØØ3ØØ CLOSE    LD      A,(DE)          ;Test for device
1EØE CB7F       ØØ31Ø          BIT     7,A
1E1Ø CA5D2Ø     ØØ32Ø          JP      Z,CLOSDEV       ;Jump if closing device
1E13 CD6815     ØØ33Ø          CALL    CKOPEN@         ;Test for open file
1E16 DD4EØ6     ØØ34Ø          LD      C,(IX+6)        ;P/u drive #
                ØØ35Ø ;
                ØØ36Ø ;        Special MINI check drive routine
                ØØ37Ø ;
1E19 FDE5       ØØ38Ø          PUSH    IY              ;Save IY
1E1B CD1E1A     ØØ39Ø          CALL    @GTDCT          ;Pick up DCT for drive
1E1E CDD419     ØØ4ØØ CKAGN    CALL    @RSLCT          ;Wait until not busy
1E21 C2882Ø     ØØ41Ø          JP      NZ,HOLDUP       ;Go to error handler
1E24 FDCBØ35E   ØØ42Ø          BIT     3,(IY+3)        ;If hard drive, bypass
1E28 2Ø24       ØØ43Ø          JR      NZ,SAWBLK
1E2A FDCBØ466   ØØ44Ø          BIT     4,(IY+4)        ;If "ALIEN" by pass
1E2E 2Ø1E       ØØ45Ø          JR      NZ,SAWBLK
1E3Ø FDCBØ47E   ØØ46Ø          BIT     7,(IY+4)        ;Ck if CKDRV inhibit
1E34 2Ø18       ØØ47Ø          JR      NZ,SAWBLK       ;Go if so
                ØØ48Ø ;
                ØØ49Ø ;        Test for diskette in drive (no index)
                ØØ5ØØ ;
1E36 D5         ØØ51Ø          PUSH    DE
1E37 FD56Ø5     ØØ52Ø          LD      D,(IY+5)        ;P/u current track
1E3A 1EØØ       ØØ53Ø          LD      E,Ø             ;Set to sector Ø
1E3C CDDØ19     ØØ54Ø          CALL    @SEEK           ;Do a command
1E3F D1         ØØ55Ø          POP     DE
1E4Ø Ø63Ø       ØØ56Ø          LD      B,3ØH           ;Set up count (short)
1E42 CDD419     ØØ57Ø BLACK    CALL    @RSLCT          ;Check for index pulse
1E45 CB4F       ØØ58Ø          BIT     1,A             ;Test index
1E47 28Ø5       ØØ59Ø          JR      Z,SAWBLK        ;Saw black, seems OK
1E49 1ØF7       ØØ6ØØ          DJNZ    BLACK
1E4B C3882Ø     ØØ61Ø          JP      HOLDUP          ;Close fault handler
                ØØ62Ø ;
                ØØ63Ø ;        Diskette is there, let's continue
                ØØ64Ø ;
1E4E FDE1       ØØ65Ø SAWBLK   POP     IY              ;Restore IY
1E5Ø DD46Ø7     ØØ66Ø          LD      B,(IX+7)        ;P/u DEC of fpde
1E53 CDBB18     ØØ67Ø          CALL    @DIRRD          ;Read the directory
```

```
1E56 CØ          ØØ68Ø          RET     NZ              ;Quit if error there
1E57 CB66        ØØ69Ø          BIT     4,(HL)          ;Ck for killed file
1E59 C8          ØØ7ØØ          RET     Z               ;Quit if killed file
1E5A E5          ØØ71Ø          PUSH    HL
1E5B C5          ØØ72Ø          PUSH    BC
1E5C CDA213      ØØ73Ø          CALL    RWRIT@          ;Write last buffer?
1E5F C1          ØØ74Ø          POP     BC
1E6Ø E1          ØØ75Ø          POP     HL
1E61 CØ          ØØ76Ø          RET     NZ              ;Ret on i/o error
1E62 DDCBØØ76    ØØ77Ø          BIT     6,(IX+Ø)        ;If user does not have
1E66 CACD1F      ØØ78Ø          JP      Z,RCVNØ         ;  close authority...
1E69 2C          ØØ79Ø          INC     L               ;  else reset possible
1E6A CBAE        ØØ8ØØ          RES     5,(HL)          ;  file-open bit in DIR+1
1E6C 2C          ØØ81Ø          INC     L               ;Determine if the EOF
1E6D 2C          ØØ82Ø          INC     L               ;  byte has changed
1E6E DD7EØ8      ØØ83Ø          LD      A,(IX+8)        ;P/u EOF byte offset
1E71 E5          ØØ84Ø          PUSH    HL              ;Save ptr to DIR+3
1E72 BE          ØØ85Ø          CP      (HL)
1E73 2Ø14        ØØ86Ø          JR      NZ,CLOS1        ;Go if moved
1E75 3E11        ØØ87Ø          LD      A,11H
1E77 85          ØØ88Ø          ADD     A,L
1E78 6F          ØØ89Ø          LD      L,A
1E79 DD7EØC      ØØ9ØØ          LD      A,(IX+12)       ;P/u lo-order ERN
1E7C BE          ØØ91Ø          CP      (HL)
1E7D 2ØØA        ØØ92Ø          JR      NZ,CLOS1        ;Go if moved
1E7F 2C          ØØ93Ø          INC     L
1E8Ø DD7EØD      ØØ94Ø          LD      A,(IX+13)       ;P/u hi-order ERN
1E83 BE          ØØ95Ø          CP      (HL)
1E84 2ØØ3        ØØ96Ø          JR      NZ,CLOS1        ;Go if moved
1E86 F1          ØØ97Ø          POP     AF
1E87 181A        ØØ98Ø          JR      CLOS2           ;Didn't move
                 ØØ99Ø ;
                 Ø1ØØØ ;        Routine to change the 3-byte EOF marker
                 Ø1Ø1Ø ;
1E89 E1          Ø1Ø2Ø CLOS1    POP     HL              ;Pop DIR+3
1E8A DD7EØ8      Ø1Ø3Ø          LD      A,(IX+8)        ;Xfer the eof offset
1E8D 77          Ø1Ø4Ø          LD      (HL),A
1E8E 3E11        Ø1Ø5Ø          LD      A,11H
1E9Ø 85          Ø1Ø6Ø          ADD     A,L
1E91 6F          Ø1Ø7Ø          LD      L,A
1E92 DD7EØC      Ø1Ø8Ø          LD      A,(IX+12)       ;  and the ERN from the FB
1E95 77          Ø1Ø9Ø          LD      (HL),A
1E96 2C          Ø11ØØ          INC     L
1E97 DD7EØD      Ø111Ø          LD      A,(IX+13)       ;  to the DIR entry
1E9A 77          Ø112Ø          LD      (HL),A
1E9B DDCBØØ56    Ø113Ø          BIT     2,(IX+Ø)        ;If file was updated
1E9F 2Ø1Ø        Ø114Ø          JR      NZ,CLOS3        ;  then update mod date
1EA1 1834        Ø115Ø          JR      CLOS5           ;  else don't
                 Ø116Ø ;
                 Ø117Ø ;        Three-byte EOF marker did not change
                 Ø118Ø ;
1EA3 DDCBØØ56    Ø119Ø CLOS2    BIT     2,(IX+Ø)        ;If file was updated
1EA7 2ØØ8        Ø12ØØ          JR      NZ,CLOS3        ;  then update mod date
1EA9 DDCBØØ76    Ø121Ø          BIT     6,(IX+Ø)        ;If close authority then
1EAD 2Ø28        Ø122Ø          JR      NZ,CLOS5        ;  write back the DIR
1EAF 182C        Ø123Ø          JR      CLOS6           ;  else continue
                 Ø124Ø ;
                 Ø125Ø ;        Routine to insert packed date into directory
                 Ø126Ø ;
1EB1 E5          Ø127Ø CLOS3    PUSH    HL              ;Save ptr to DIR+21
1EB2 7D          Ø128Ø          LD      A,L             ;Pt to start of dir rec
```

```
1EB3 E6EØ      Ø129Ø          AND     ØEØH
1EB5 6F        Ø13ØØ          LD      L,A
1EB6 2C        Ø131Ø          INC     L               ;Pt to DIR+1
1EB7 CBF6      Ø132Ø          SET     6,(HL)          ;Set the MOD flag
1EB9 1133ØØ    Ø133Ø          LD      DE,DATE$        ;Point to year
1EBC 1A        Ø134Ø          LD      A,(DE)          ;If year = Ø, then date
1EBD B7        Ø135Ø          OR      A               ;  is ØØ/ØØ/ØØ
1EBE 28Ø2      Ø136Ø          JR      Z,$+4
1ECØ D65Ø      Ø137Ø          SUB     8Ø              ;Offset from 198Ø
1EC2 C5        Ø138Ø          PUSH    BC
1EC3 47        Ø139Ø          LD      B,A             ;Year-8Ø -> regB
1EC4 13        Ø14ØØ          INC     DE              ;Point to day
1EC5 1A        Ø141Ø          LD      A,(DE)          ;Shift day into 3-7 &
1EC6 Ø7        Ø142Ø          RLCA                    ;  merge the year into
1EC7 Ø7        Ø143Ø          RLCA                    ;  the lo-order bits
1EC8 Ø7        Ø144Ø          RLCA
1EC9 BØ        Ø145Ø          OR      B
1ECA 2C        Ø146Ø          INC     L
1ECB 77        Ø147Ø          LD      (HL),A          ;Store day/year
1ECC 2D        Ø148Ø          DEC     L
1ECD 13        Ø149Ø          INC     DE              ;Point to month
1ECE 1A        Ø15ØØ          LD      A,(DE)
1ECF 47        Ø151Ø          LD      B,A
1EDØ 7E        Ø152Ø          LD      A,(HL)          ;P/u dir byte
1ED1 E6FØ      Ø153Ø          AND     ØFØH            ;Strip old month
1ED3 BØ        Ø154Ø          OR      B               ;Merge month &
1ED4 77        Ø155Ø          LD      (HL),A          ;  update the field
1ED5 C1        Ø156Ø          POP     BC
1ED6 E1        Ø157Ø CLOS4    POP     HL              ;Rcvr DIR+21
1ED7 E5        Ø158Ø CLOS5    PUSH    HL
1ED8 CDØ318    Ø159Ø          CALL    @DIRWR          ;Write back DIR entry
1EDB E1        Ø16ØØ          POP     HL
1EDC CØ        Ø161Ø          RET     NZ
1EDD 2C        Ø162Ø CLOS6    INC     L               ;Pt to DIR+22 which is
1EDE E5        Ø163Ø          PUSH    HL              ;  the 1st extent
1EDF 7D        Ø164Ø          LD      A,L
1EEØ D615      Ø165Ø          SUB     15H             ;Backup to DIR+1
1EE2 6F        Ø166Ø          LD      L,A
1EE3 CB7E      Ø167Ø          BIT     7,(HL)          ;Test if created
1EE5 E1        Ø168Ø          POP     HL
1EE6 C2CD1F    Ø169Ø          JP      NZ,RCVNØ        ;Bypass if created
1EE9 11ØØØØ    Ø17ØØ          LD      DE,Ø            ;Init gran counter
1EEC 7E        Ø171Ø CLOS7    LD      A,(HL)          ;P/u cyl indicator
1EED 2C        Ø172Ø          INC     L               ;Pt to gran alloc
1EEE FEFE      Ø173Ø          CP      ØFEH            ;Extent in use?
1EFØ 3ØØC      Ø174Ø          JR      NC,CLOS8        ;Jump if spare or FXDE
1EF2 7E        Ø175Ø          LD      A,(HL)          ;P/u granule allocation
1EF3 2C        Ø176Ø          INC     L               ;Pt to next extent
1EF4 E61F      Ø177Ø          AND     1FH             ;Strip off # of grans &
1EF6 3C        Ø178Ø          INC     A               ;  adjust for zero offset
1EF7 83        Ø179Ø          ADD     A,E             ;Accumulate the number of
1EF8 5F        Ø18ØØ          LD      E,A             ;  grans in this extent
1EF9 3ØF1      Ø181Ø          JR      NC,CLOS7        ;Any previous quantity
1EFB 14        Ø182Ø          INC     D
1EFC 18EE      Ø183Ø          JR      CLOS7
1EFE 2ØØB      Ø184Ø CLOS8    JR      NZ,CLOS9        ;Found all grans in this
1FØØ 46        Ø185Ø          LD      B,(HL)          ;  extent, ck for FXDE
1FØ1 CDBB18    Ø186Ø          CALL    @DIRRD
1FØ4 CØ        Ø187Ø          RET     NZ
1FØ5 7D        Ø188Ø          LD      A,L             ;Point to extents in FXDE
1FØ6 C616      Ø189Ø          ADD     A,16H
```

```
1FØ8 6F       Ø19ØØ          LD      L,A
1FØ9 18E1     Ø191Ø          JR      CLOS7           ;Go to continue count
              Ø192Ø ;
              Ø193Ø ;        Routine to determine need to deallocate
              Ø194Ø ;
1FØB E5       Ø195Ø CLOS9    PUSH    HL              ;Save ptr to last extent
1FØC DD6EØC   Ø196Ø          LD      L,(IX+12)       ;P/u ending record #
1FØF DD66ØD   Ø197Ø          LD      H,(IX+13)
1F12 3EØ8     Ø198Ø          LD      A,8             ;Get # sectors/gran
1F14 CD2B1A   Ø199Ø          CALL    @DCTBYT
1F17 E61F     Ø2ØØØ          AND     1FH             ;Remove other data
1F19 F5       Ø2Ø1Ø          PUSH    AF              ;Save the #
1F1A 85       Ø2Ø2Ø          ADD     A,L             ;Round up to next
1F1B 6F       Ø2Ø3Ø          LD      L,A             ;  higher gran
1F1C 3ØØ1     Ø2Ø4Ø          JR      NC,CLOS1Ø
1F1E 24       Ø2Ø5Ø          INC     H
1F1F F1       Ø2Ø6Ø CLOS1Ø   POP     AF              ;Rcvr # sectors/gran
1F2Ø 3C       Ø2Ø7Ø          INC     A               ;Adjust for division
1F21 CDE3Ø6   Ø2Ø8Ø          CALL    @DIV16          ;Calculate # grans in use
1F24 AF       Ø2Ø9Ø          XOR     A               ;Subtract the # of grans
1F25 EB       Ø21ØØ          EX      DE,HL           ;  used from the # of
1F26 ED52     Ø211Ø          SBC     HL,DE           ;  grans allocated in the
1F28 EB       Ø212Ø          EX      DE,HL           ;  directory, and move DE
1F29 E1       Ø213Ø          POP     HL              ;Rcvr ptr to last extent
1F2A CACD1F   Ø214Ø          JP      Z,RCVNØ         ;Jump if same quantity
1F2D DACD1F   Ø215Ø          JP      C,RCVNØ         ;Jump if now more
              Ø216Ø ;
              Ø217Ø ;        Need to deallocate space
              Ø218Ø ;
1F3Ø CD7418   Ø219Ø          CALL    @GATRD          ;Read GAT
1F33 CØ       Ø22ØØ          RET     NZ
1F34 1841     Ø221Ø          JR      BAKUP           ;B/u to last used extent
1F36 D5       Ø222Ø CLOS11   PUSH    DE              ;Sv count of excess grans
1F37 7E       Ø223Ø          LD      A,(HL)          ;P/u alloc info
1F38 E6EØ     Ø224Ø          AND     ØEØH            ;Get starting relative
1F3A Ø7       Ø225Ø          RLCA                    ;  gran into reg-E
1F3B Ø7       Ø226Ø          RLCA
1F3C Ø7       Ø227Ø          RLCA
1F3D 5F       Ø228Ø          LD      E,A
1F3E 7E       Ø229Ø          LD      A,(HL)          ;# of contiguous grans
1F3F E61F     Ø23ØØ          AND     1FH             ;Remove unneeded data
1F41 83       Ø231Ø          ADD     A,E             ;Calculate ending
1F42 5F       Ø232Ø          LD      E,A             ;  relative gran #
1F43 3EØ8     Ø233Ø          LD      A,8             ;P/u the # of grans
1F45 CD2B1A   Ø234Ø          CALL    @DCTBYT         ;  per cylinder
1F48 Ø7       Ø235Ø          RLCA
1F49 Ø7       Ø236Ø          RLCA
1F4A Ø7       Ø237Ø          RLCA
1F4B E6Ø7     Ø238Ø          AND     7               ;Move into bits Ø-2
1F4D 3C       Ø239Ø          INC     A               ;Adjust for zero offset
1F4E 57       Ø24ØØ          LD      D,A             ;Save count
1F4F 3EØ4     Ø241Ø          LD      A,4
1F51 CD2B1A   Ø242Ø          CALL    @DCTBYT
1F54 CB6F     Ø243Ø          BIT     5,A             ;2-sided disk?
1F56 7A       Ø244Ø          LD      A,D             ;Rcvr count
1F57 28Ø1     Ø245Ø          JR      Z,$+3           ;Bypass if 1-sided
1F59 Ø7       Ø246Ø          RLCA                    ;Double count
1F5A CD2719   Ø247Ø          CALL    @DIV8           ;A=quotient, E=remainder
1F5D 2D       Ø248Ø          DEC     L               ;Pt to starting cylinder
1F5E 86       Ø249Ø          ADD     A,(HL)          ;Bump cyl pointer by how
1F5F 57       Ø25ØØ          LD      D,A             ;  many excessive cyls to
```

```
1F6Ø E5       Ø251Ø          PUSH    HL              ;  start from the rear
1F61 C5       Ø252Ø          PUSH    BC
1F62 2623     Ø253Ø          LD      H,DIRBUF$<-8    ;Pt to that cyl's GAT
1F64 6A       Ø254Ø          LD      L,D
1F65 46       Ø255Ø          LD      B,(HL)          ;P/u the GAT allocation
1F66 7B       Ø256Ø          LD      A,E
1F67 CD7B2Ø   Ø257Ø          CALL    CALCBIT         ;Deallocate a gran
1F6A 7Ø       Ø258Ø          LD      (HL),B          ;Repl GAT byte
1F6B C1       Ø259Ø          POP     BC
1F6C E1       Ø26ØØ          POP     HL
1F6D 2C       Ø261Ø          INC     L               ;Repoint to alloc info
1F6E 35       Ø262Ø          DEC     (HL)            ;Reduce by 1 gran
1F6F 7E       Ø263Ø          LD      A,(HL)          ;Get info on contig gran
1F7Ø 3C       Ø264Ø          INC     A               ;Adj for zero offset
1F71 E61F     Ø265Ø          AND     1FH             ;Strip unneeded
1F73 D1       Ø266Ø          POP     DE              ;Rcvr excess gran count
1F74 1B       Ø267Ø          DEC     DE              ;  and count down
1F75 2Ø43     Ø268Ø          JR      NZ,CLOS12       ;Go if extent still used
1F77 36FF     Ø269Ø BAKUP    LD      (HL),ØFFH       ;  else extent is spare
1F79 2D       Ø27ØØ          DEC     L
1F7A 36FF     Ø271Ø          LD      (HL),ØFFH
1F7C 2D       Ø272Ø          DEC     L
1F7D 7D       Ø273Ø          LD      A,L             ;Chack if backed all the
1F7E E61F     Ø274Ø          AND     1FH             ;  way thru this entry
1F8Ø FE15     Ø275Ø          CP      15H
1F82 2Ø36     Ø276Ø          JR      NZ,CLOS12       ;Go if not
1F84 AD       Ø277Ø          XOR     L               ;Deallocate this FXDE
1F85 6F       Ø278Ø          LD      L,A
1F86 CB7E     Ø279Ø          BIT     7,(HL)          ;Was it the FPDE?
1F88 283Ø     Ø28ØØ          JR      Z,CLOS12        ;Bypass if FPDE
1F8A 36ØØ     Ø281Ø          LD      (HL),Ø          ;Show dir is spare
1F8C CDØ318   Ø282Ø          CALL    @DIRWR          ;Write back
1F8F CØ       Ø283Ø          RET     NZ
1F9Ø 78       Ø284Ø          LD      A,B             ;P/u deallocated DEC
1F91 E6EØ     Ø285Ø          AND     ØEØH
1F93 3C       Ø286Ø          INC     A               ;Pt to DIR+1
1F94 6F       Ø287Ø          LD      L,A
1F95 7E       Ø288Ø          LD      A,(HL)          ;P/u previous DEC
1F96 32A51F   Ø289Ø          LD      (STUFDEC+1),A   ;Save in instruction
1F99 CD9718   Ø29ØØ          CALL    @HITRD          ;Read the HIT
1F9C CØ       Ø291Ø          RET     NZ
1F9D 68       Ø292Ø          LD      L,B             ;Point to deallocated HIT
1F9E 36ØØ     Ø293Ø          LD      (HL),Ø          ;Deallocate space in HIT
1FAØ CD9818   Ø294Ø          CALL    @HITWR          ;Write back
1FA3 CØ       Ø295Ø          RET     NZ
1FA4 Ø6ØØ     Ø296Ø STUFDEC  LD      B,Ø             ;P/u previous DEC
1FA6 CDBB18   Ø297Ø          CALL    @DIRRD          ;Read its dir entry
1FA9 CØ       Ø298Ø          RET     NZ
1FAA 78       Ø299Ø          LD      A,B
1FAB F61F     Ø3ØØØ          OR      1FH             ;Pt to end of entry
1FAD 6F       Ø3Ø1Ø          LD      L,A
1FAE 36FF     Ø3Ø2Ø          LD      (HL),ØFFH       ;Erase pointer
1FBØ 2D       Ø3Ø3Ø          DEC     L               ;  to deallocated FXDE
1FB1 36FF     Ø3Ø4Ø          LD      (HL),ØFFH
1FB3 2D       Ø3Ø5Ø          DEC     L               ;Point to previous extent
1FB4 E5       Ø3Ø6Ø          PUSH    HL              ;Save pointer
1FB5 CDØ318   Ø3Ø7Ø          CALL    @DIRWR          ;Write back
1FB8 E1       Ø3Ø8Ø          POP     HL
1FB9 CØ       Ø3Ø9Ø          RET     NZ
1FBA 7A       Ø31ØØ CLOS12   LD      A,D             ;Loop if still more to
1FBB B3       Ø311Ø          OR      E               ;  deallocate
```

```
1FBC C2361F   03120          JP      NZ,CLOS11
1FBF CD0318   03130          CALL    @DIRWR
1FC2 2805     03140          JR      Z,CLOS13        ;Go if no write error
1FC4 FE0F     03150          CP      15              ;"write protected...
1FC6 C0       03160          RET     NZ              ;Bad if not
1FC7 1804     03170          JR      RCVN0
              03180 ;
1FC9 CD7518   03190 CLOS13   CALL    @GATWR          ;Write back the altered GAT
1FCC C0       03200          RET     NZ
              03210 ;
              03220 ;        Routine starts to recover file spec
              03230 ;
1FCD DD7E07   03240 RCVN0    LD      A,(IX+7)        ;P/u DEC of FPDE
1FD0 DD4E06   03250          LD      C,(IX+6)        ;P/u drive
1FD3 A8       03260          XOR     B               ;Check if its directory
1FD4 E61F     03270          AND     1FH             ;  record is resident
1FD6 DD4607   03280          LD      B,(IX+7)        ;P/u DEC of FPDE
1FD9 C4BB18   03290          CALL    NZ,@DIRRD       ;Get FPDE dir if needed
1FDC C0       03300          RET     NZ
1FDD DDE5     03310          PUSH    IX              ;Transfer FCB to DE
1FDF D1       03320          POP     DE
1FE0 79       03330 RCVNAM   LD      A,C
1FE1 E607     03340          AND     7               ;Convert drive to ASCII
1FE3 F630     03350          OR      '0'
1FE5 321C20   03360          LD      (RCVN5+1),A
1FE8 261D     03370          LD      H,SBUFF$<-8     ;Pt to DIR+5 (name)
1FEA 78       03380          LD      A,B
1FEB E6E0     03390          AND     0E0H
1FED F605     03400          OR      5
1FEF 6F       03410          LD      L,A
1FF0 E5       03420          PUSH    HL              ;Save name start posn
1FF1 0608     03430          LD      B,8             ;Init 8 chars max
1FF3 7E       03440 RCVN1    LD      A,(HL)          ;Move filename from
1FF4 FE20     03450          CP      ' '             ;  direc to fcb
1FF6 2805     03460          JR      Z,RCVN2
1FF8 12       03470          LD      (DE),A
1FF9 23       03480          INC     HL
1FFA 13       03490          INC     DE
1FFB 10F6     03500          DJNZ    RCVN1           ;Loop up to 8
1FFD E1       03510 RCVN2    POP     HL
1FFE 7D       03520          LD      A,L
1FFF C608     03530          ADD     A,8             ;Pt to extension
2001 6F       03540          LD      L,A
2002 7E       03550          LD      A,(HL)
2003 FE20     03560          CP      ' '
2005 2810     03570          JR      Z,RCVN4         ;Jump if none
2007 3E2F     03580          LD      A,'/'
2009 12       03590          LD      (DE),A          ;Stuff separator into fcb
200A 13       03600          INC     DE
200B 0603     03610          LD      B,3             ;Init 3-char extension
200D 7E       03620 RCVN3    LD      A,(HL)          ;Stuff the ext
200E FE20     03630          CP      ' '             ;  into fcb
2010 2805     03640          JR      Z,RCVN4
2012 12       03650          LD      (DE),A
2013 23       03660          INC     HL
2014 13       03670          INC     DE
2015 10F6     03680          DJNZ    RCVN3
2017 3E3A     03690 RCVN4    LD      A,':'           ;Stuff drive indicator
2019 12       03700          LD      (DE),A
201A 13       03710          INC     DE
201B 3E00     03720 RCVN5    LD      A,0             ;P/u drive in ASCII
```

```
201D 12       03730          LD      (DE),A          ;  & stuff it
201E 13       03740          INC     DE
201F 3E03     03750          LD      A,3             ;Close FCB with ETX
2021 12       03760          LD      (DE),A
2022 AF       03770          XOR     A
2023 C9       03780          RET
              03790 ;
              03800 ;        Routine to recover the filespec
              03810 ;
2024 E5       03820 FNAME    PUSH    HL
2025 D5       03830          PUSH    DE
              03840 ;
              03850 ;        Calculate the number of directory sectors
              03860 ;        = (#sectors x #heads) - 2 for GAT & HIT
              03870 ;
2026 3E07     03880          LD      A,7             ;Get highest # sector
2028 CD2B1A   03890          CALL    @DCTBYT
202B 57       03900          LD      D,A             ;Store heads & sectors
202C E61F     03910          AND     1FH             ;Rake off # sectors
202E 5F       03920          LD      E,A             ;  & stuff into E
202F 1C       03930          INC     E               ;Bump for 0 offset
2030 AA       03940          XOR     D               ;Recover # heads
2031 07       03950          RLCA                    ;  into bits 0-2
2032 07       03960          RLCA
2033 07       03970          RLCA
2034 3C       03980          INC     A               ;Bump for 0 offset
2035 CD0A19   03990          CALL    @MUL8           ;Multiply sectors x heads
2038 5F       04000          LD      E,A             ;Now check double bit
2039 3E04     04010          LD      A,4
203B CD2B1A   04020          CALL    @DCTBYT
203E CB6F     04030          BIT     5,A             ;Set if 2-sided
2040 7B       04040          LD      A,E
2041 2801     04050          JR      Z,ONESID        ;Go if not set else
2043 87       04060          ADD     A,A             ;  double value
2044 D602     04070 ONESID   SUB     2               ;Reduce for GAT & HIT
2046 57       04080          LD      D,A
2047 78       04090          LD      A,B
2048 E61F     04100          AND     1FH             ;Calc req sector #
204A BA       04110          CP      D
204B 3805     04120          JR      C,FNAM1
204D 3E10     04130          LD      A,16            ;"Illegal logical file #
204F B7       04140          OR      A
2050 1808     04150          JR      FNAM2
2052 D1       04160 FNAM1    POP     DE
2053 D5       04170          PUSH    DE
2054 CDBB18   04180          CALL    @DIRRD
2057 CCE01F   04190          CALL    Z,RCVNAM        ;Rcvr the filespec
205A D1       04200 FNAM2    POP     DE
205B E1       04210          POP     HL
205C C9       04220          RET
              04230 ;
              04240 ;        Close a logical device
              04250 ;
205D FE10     04260 CLOSDEV  CP      10H             ;If not open device,
205F 3E26     04270          LD      A,38            ;  return "file not open...
2061 C0       04280          RET     NZ
2062 CD6615   04290          CALL    LNKFCB@         ;Link to FCB
2065 DD4E06   04300          LD      C,(IX+6)        ;Get device name
2068 DD4607   04310          LD      B,(IX+7)
206B DD3600 2A 04320         LD      (IX+0),'*'      ;Stuff device indicator
206F DD7101   04330          LD      (IX+1),C        ;Stuff 1st char of name
```

```
2072 DD7002    04340          LD      (IX+2),B        ;Stuff 2nd char of name
2075 DD360303  04350          LD      (IX+3),3        ;Terminate with ETX
2079 AF        04360          XOR     A
207A C9        04370          RET
               04380 ;
               04390 ;        Calculate GAT bit to deallocate
               04400 ;
207B E607      04410 CALCBIT  AND     7               ;Make binary bit # into
207D 07        04420          RLCA                    ;  the proper RES
207E 07        04430          RLCA                    ;  opcode
207F 07        04440          RLCA
2080 F680      04450          OR      80H
2082 328620    04460          LD      (CALC1+1),A
2085 CB80      04470 CALC1    RES     0,B             ;Reset bit in GAT
2087 C9        04480          RET
               04490 ;
               04500 ;        User removed disk with an open file
               04510 ;
2088 E5        04520 HOLDUP   PUSH    HL
2089 D5        04530          PUSH    DE
208A 21AE20    04540          LD      HL,HOLDUP$      ;Pt to message
208D CD2D05    04550          CALL    @DSPLY          ;Display to console
2090 CD5305    04560          CALL    @CKBRKC         ;Clear out break bit
2093 CD3506    04570 WAITING  CALL    @KBD            ;Scan the keyboard
2096 20FB      04580          JR      NZ,WAITING      ;Keep looking
2098 FE0D      04590          CP      CR              ;Check for <ENTER>
209A 280D      04600          JR      Z,TRYNOW
209C CD5305    04610          CALL    @CKBRKC         ;Check for a break
209F 28F2      04620          JR      Z,WAITING
20A1 D1        04630 ABRT     POP     DE
20A2 E1        04640          POP     HL
20A3 FDE1      04650          POP     IY              ;Restore from above
20A5 3E20      04660          LD      A,32            ;Show illegal drive #
20A7 B7        04670          OR      A               ;Set NZ condition
20A8 C9        04680          RET                     ;Go back now
20A9 D1        04690 TRYNOW   POP     DE
20AA E1        04700          POP     HL
20AB C31E1E    04710          JP      CKAGN           ;Try checking again
20AE 0A        04720 HOLDUP$  DB      LF,'** CLOSE FAULT **  Drive not ready, '
     2A 2A 20 43 4C 4F 53 45
     20 46 41 55 4C 54 20 2A
     2A 20 20 44 72 69 76 65
     20 6E 6F 74 20 72 65 61
     64 79 2C 20
20D3 3C        04730          DB      '<ENTER> to retry, <BREAK> to abort',CR
     45 4E 54 45 52 3E 20 74
     6F 20 72 65 74 72 79 2C
     20 3C 42 52 45 41 4B 3E
     20 74 6F 20 61 62 6F 72
     74 0D
20F6           04740 LAST     EQU     $
               04750          IFGT    $,DIRBUF$
               04760          ERR     'Module too big'
               04770          ENDIF
23FE           04780          ORG     MAXCOR$-2
23FE F602      04790          DW      LAST-SYS3       ;Overlay length
               04800 ;
1E00           04810          END     SYS3
```

$A1	Ø3B7	$A2	Ø3B8	$A3	Ø3B9		
$CKEOF	147Ø	@$SYS	Ø8FØ	@@1	ØØØØ		
@@2	ØØØØ	@@3	ØØØØ	@@4	ØØØØ		
@ABORT	1BØ8	@ADTSK	1CDA	@BANK	Ø877		
@BKSP	1486	@BREAK	196F	@BYTEIO	13ØØ		
@CHNIO	Ø689	@CKBRKC	Ø553	@CKDRV	1993		
@CKEOF	158F	@CKTSK	1CF5	@CLOSE	1999		
@CLS	Ø545	@CMNDI	197E	@CMNDR	197B		
@CTL	Ø623	@DATE	Ø7A8	@DBGHK	199F		
@DCINIT	19CØ	@DCRES	19C4	@DCSTAT	19B5		
@DCTBYT	1A2B	@DEBUG	19AØ	@DECHEX	Ø3E1		
@DIRCYL	18F7	@DIRRD	18BB	@DIRWR	18Ø3		
@DIV16	Ø6E3	@DIV8	1927	@DODIR	19AF		
@DOKEY	19A9	@DSP	Ø642	@DSPLY	Ø52D		
@ERROR	1BØF	@EXIT	1BØB	@FEXT	1984		
@FLAGS	196A	@FNAME	199C	@FRENCH	ØØØØ		
@FSPEC	1981	@GATRD	1874	@GATWR	1875		
@GERMAN	ØØØØ	@GET	Ø638	@GTDCB	199Ø		
@GTDCT	1A1E	@GTMOD	19B2	@HDFMT	19E4		
@HEX16	Ø7BD	@HEX8	Ø7C2	@HEXDEC	Ø6F6		
@HIGH$	1948	@HITRD	1897	@HITWR	1898		
@HZ5Ø	ØØØØ	@ICNFG	ØØ86	@INIT	198D		
@INTL	ØØØØ	@IPL	1BF2	@JCL	Ø63Ø		
@KBD	Ø635	@KEY	Ø628	@KEYIN	Ø585		
@KITSK	ØØ89	@KLTSK	1CDØ	@LOAD	1B38		
@LOC	14B3	@LOF	14DE	@LOGER	Ø5Ø3		
@LOGOT	Ø5ØØ	@MOD2	ØØØØ	@MOD4	FFFF		
@MSG	Ø53Ø	@MUL16	Ø6C9	@MUL8	19ØA		
@NMI	ØØ66	@OPEN	198A	@OPREG	ØØ84		
@PARAM	1987	@PAUSE	Ø382	@PEOF	14A2		
@POSN	1434	@PRINT	Ø528	@PRT	Ø63D		
@PUT	Ø645	@RAMDIR	19AC	@RDHDR	19D8		
@RDSEC	19F4	@RDSSC	18D8	@RDTRK	19EØ		
@READ	1513	@REMOVE	19A6	@RENAME	1996		
@REW	149B	@RMTSK	1CD7	@RPTSK	1CEB		
@RREAD	1473	@RSLCT	19D4	@RSTØØ	ØØØØ		
@RSTØ8	ØØØ8	@RST1Ø	ØØ1Ø	@RST18	ØØ18		
@RST2Ø	ØØ2Ø	@RST28	ØØ28	@RST3Ø	ØØ3Ø		
@RST38	ØØ38	@RSTNMI	ØFE9	@RSTOR	19C8		
@RSTREG	Ø68Ø	@RUN	1B1D	@RWRIT	13AD		
@SEEK	19DØ	@SEEKSC	1421	@SKIP	143Ø		
@SLCT	19BC	@SOUND	Ø392	@STEPI	19CC		
@TIME	Ø78D	@USA	FFFF	@VDCTL	ØB99		
@VDCTL3	ØD38	@VER	156Ø	@VRSEC	19DC		
@WEOF	14EC	@WHERE	1979	@WRITE	1531		
@WRSEC	19E8	@WRSSC	19EC	@WRTRK	19FØ		
@_VDCTL	ØD42	ABRT	2ØA1	ADDR_2_ROWCOL	ØDF1		
AFLAG$	ØØ6A	AUTO?	1FF1	BAKUP	1F77		
BAR$	Ø2Ø1	BLACK	1E42	BOOTST$	439D		
BREAK?	1C6Ø	BRKVEC$	1C88	BUR$	Ø2ØØ		
CALC1	2Ø85	CALCBIT	2Ø7B	CASHK$	ØA7B		
CFCB$	ØØEØ	CFGFCB$	ØØEØ	CFLAG$	ØØ6C		
CKAGN	1E1E	CKMOD@	1A7F	CKOPEN@	1568		
CLOS1	1E89	CLOS1Ø	1F1F	CLOS11	1F36		
CLOS12	1FBA	CLOS13	1FC9	CLOS2	1EA3		
CLOS3	1EB1	CLOS4	1ED6	CLOS5	1ED7		
CLOS6	1EDD	CLOS7	1EEC	CLOS8	1EFE		
CLOS9	1FØB	CLOSDEV	2Ø5D	CLOSE	1EØD		
CONFIG$	2Ø3F	CORE$	Ø3ØØ	CR	ØØØD		
CRTBGN$	F8ØØ	CYL_GRN	16AE	D@FBYT8	1A26		

```
DATE$           ØØ33 DAYTBL$         Ø4C7 DBGSV$          ØØAØ
DCBKL$          ØØ31 DCT$            Ø47Ø DCTBYT8@        1A29
DCTFLD@         1A34 DFLAG$          ØØ6D DIRBUF$         23ØØ
DIS_DO_RAM      Ø846 DODATA$         ØB94 DODCB$          Ø21Ø
DO_CONTROL      ØC44 DO_DSPCHAR      ØCB8 DO_INVERT_DIS   ØC8C
DO_INVERT_ENA   ØC89 DO_INVERT_OFF   ØC9B DO_MASK         ØØØØ
DO_RET          ØBCB DO_RET1         ØBCC DO_SCROLL       ØCCE
DO_TABS         ØBEA DSKTYP$         Ø4CØ DTPMT$          Ø4C2
DVREND$         ØFF4 DVRHI$          Ø2Ø6 EFLAG$          ØØ6E
ENADIS_DO_RAM   Ø817 EXTDBG$         19A4 FDDINT$         ØØØE
FEMSK$          ØØ6F FLGTAB$         ØØ6A FNAM1           2Ø52
FNAM2           2Ø5A FNAME           2Ø24 GET_@_ROWCOL    ØDAE
HERTZ$          Ø75Ø HIGH$           Ø4ØE HKRES$          1A6C
HOLDUP          2Ø88 HOLDUP$         2ØAE IFLAG$          ØØ72
INBUF$          Ø42Ø INTIM$          ØØ3C INTMSK$         ØØ3D
INTVC$          ØØ3E JCLCB$          Ø2Ø3 JDCB$           ØØ24
JFCB$           ØØCØ JLDCB$          Ø23Ø JRET$           ØØ26
KCK@            Ø7D6 KFLAG$          ØØ74 KIDATA$         Ø8FC
KIDCB$          Ø2Ø8 LAST            2ØF6 LBANK$          Ø2Ø2
LDRV$           ØØ23 LF              ØØØA LFLAG$          ØØ75
LNKFCB@         1566 LOW$            ØØ1E LSVC$           ØØØD
MAXCOR$         24ØØ MAXDAY$         Ø4Ø1 MINCOR$         3ØØØ
MODOUT$         ØØ76 MONTBL$         Ø4DC NFLAG$          ØØ77
ONESID          2Ø44 OPREG$          ØØ78 OPREG_SV_AREA   Ø86E
OPREG_SV_PTR    Ø835 ORARET@         14DC OSRLS$          ØØ3B
OSVER$          ØØ85 OVRLY$          ØØ69 PAKNAM$         Ø41Ø
PAUSE@          Ø382 PCSAVE$         Ø7AF PDRV$           ØØ1B
PHIGH$          ØØ1C PRDCB$          Ø218 PUTA@DE         ØDCD
PUT_@           ØDCA PUT_@_ROWCOL    ØDC6 RCVNØ           1FCD
RCVN1           1FF3 RCVN2           1FFD RCVN3           2ØØD
RCVN4           2Ø17 RCVN5           2Ø1B RCVNAM          1FEØ
RFLAG$          ØØ7B ROWCOL_2_ADDR   ØDDØ RST38@          1BFF
RSTOR$          Ø4C4 RWRIT@          13A2 S1DCB$          Ø238
SAWBLK          1E4E SBUFF$          1DØØ SET@EXEC        1A79
SET_SCROLL      ØCF3 SFCB$           ØØ8C SFLAG$          ØØ7C
SIDCB$          Ø22Ø SODCB$          Ø228 SPACE4$         2142
STACK$          Ø38Ø START$          ØØØØ STUFDEC         1FA4
SVCRET$         ØØØB SVCTAB$         Ø1ØØ SYS3            1EØØ
SYSERR$         1B13 TCB$            ØØ4E TFLAG$          ØØ7D
TIME$           ØØ2D TIMER$          ØØ2C TIMSL$          ØØ2B
TIMTSK$         Ø713 TMPMT$          Ø4C3 TRACE_INT       Ø7B1
TRYNOW          2ØA9 TYPHK$          ØA8F TYPTSK$         ØB26
USTOR$          ØØ13 VFLAG$          ØØ7F WAITING         2Ø93
WRINT$          ØØ8Ø ZERO$           Ø4Ø1 ZEROA@          13AØ

1EØØ is the transfer address
ØØØØØ Total errors
```

NOTES:

SYS4 handles the system error routines, either displaying an error message or placing the message in a user specified buffer. Besides the standard error codes, an "extended error" (error number 63) will display an error value placed in the HL register pair. The only SVC handled by SYS4 is @ERROR.

```
                    00100 ;SYS4/ASM - LS-DOS 6.2
0000                00110         TITLE   <SYS4 - LS-DOS 6.2>
000A                00120 LF      EQU     10
000D                00130 CR      EQU     13
                    00140 *LIST   OFF                     ;Get SYS0/EQU
                    00160 *LIST   ON
0000                00170 *GET    COPYCOM:3               ;Copyright message
                    03010 ; COPYCOM - File for Copyright COMment block
                    03020 ;
0000                03030         COM     '<*(C) 1982,83,84 by LSI*>'
                    03040 ;
                    00180 ;
1E00                00190         ORG     1E00H
                    00200 ;
1E00 C3AF1E         00210 SYS4    JP      BEGIN
                    00220 ;
                    00230 ;       Sentence table - Must be totally within one page
                    00240 ;
1E03 01             00250 MSG0    DB      1,2+80H
     82
                    00260 ;               no error
1E05 04             00270 MSG1    DB      4,2,5,6,9+80H
     02 05 06 89
                    00280 ;               parity error during header read
1E0A 08             00290 MSG2    DB      8,2,5,9+80H
     02 05 89
                    00300 ;               seek error during read
1E0E 0B             00310 MSG3    DB      11,7,5,9+80H
     07 05 89
                    00320 ;               lost data during read
1E12 04             00330 MSG4    DB      4,2,5,9+80H
     02 05 89
                    00340 ;               parity error during read
1E16 07             00350 MSG5    DB      7,27,12,44,5,9+80H
     1B 0C 2C 05 89
                    00360 ;               data record not found during read
1E1C 0D             00370 MSG6    DB      13,9,15,7,27+80H
     09 0F 07 9B
                    00380 ;               attempted to read system data record
1E21 0D             00390 MSG7    DB      13,9,14,7,27+80H
     09 0E 07 9B
                    00400 ;               attempted to read locked/deleted data record
1E26 2A             00410 MSG8    DB      42,12,51+0C0H
     0C F3
                    00420 ;               device not available
1E29 04             00430 MSG9    DB      4,2,5,6,10+80H
     02 05 06 8A
                    00440 ;               parity error during header write
1E2E 08             00450 MSG10   DB      8,2,5,10+80H
     02 05 8A
                    00460 ;               seek error during write
1E32 0B             00470 MSG11   DB      11,7,5,10+80H
     07 05 8A
                    00480 ;               lost data during write
1E36 04             00490 MSG12   DB      4,2,5,10+80H
     02 05 8A
                    00500 ;               parity error during write
1E3A 07             00510 MSG13   DB      7,27,12,44,5,10+80H
     1B 0C 2C 05 8A
                    00520 ;               data record not found during write
1E40 0A             00530 MSG14   DB      10,21,18,19,48+80H
```

```
     15 12 13 BØ
                    ØØ54Ø ;                 write fault on disk drive
1E45 ØA             ØØ55Ø MSG15   DB        1Ø,22,19+8ØH
     16 93
                    ØØ56Ø ;                 write protected disk
1E48 17             ØØ57Ø MSG16   DB        23,24,26,25+8ØH
     18 1A 99
                    ØØ58Ø ;                 illegal logical file number
1E4C 1Ø             ØØ59Ø MSG17   DB        16,9,2+8ØH
     Ø9 82
                    ØØ6ØØ ;                 directory read error
1E4F 1Ø             ØØ61Ø MSG18   DB        16,1Ø,2+8ØH
     ØA 82
                    ØØ62Ø ;                 directory write error
1E52 17             ØØ63Ø MSG19   DB        23,26,41+ØCØH
     1A E9
                    ØØ64Ø ;                 illegal file name
1E55 22             ØØ65Ø MSG2Ø   DB        34,9,2+8ØH
     Ø9 82
                    ØØ66Ø ;                 gat read error
1E58 22             ØØ67Ø MSG21   DB        34,1Ø,2+8ØH
     ØA 82
                    ØØ68Ø ;                 gat write error
1E5B 23             ØØ69Ø MSG22   DB        35,9,2+8ØH
     Ø9 82
                    ØØ7ØØ ;                 hit read error
1E5E 23             ØØ71Ø MSG23   DB        35,1Ø,2+8ØH
     ØA 82
                    ØØ72Ø ;                 hit write error
1E61 1A             ØØ73Ø MSG24   DB        26,12,45,16+ØCØH
     ØC 2D DØ
                    ØØ74Ø ;                 file not in directory
1E65 1A             ØØ75Ø MSG25   DB        26,46,49+ØCØH
     2E F1
                    ØØ76Ø ;                 file access denied
1E68 Ø1             ØØ77Ø MSG26   DB        1,16,39,51+ØCØH
     1Ø 27 F3
                    ØØ78Ø ;                 directory space full
1E6C 13             ØØ79Ø MSG27   DB        19,39,47+8ØH
     27 AF
                    ØØ8ØØ ;                 disk space full
1E6F 1C             ØØ81Ø MSG28   DB        28,29,26,32+8ØH
     1D 1A AØ
                    ØØ82Ø ;                 end of file encountered
1E73 1B             ØØ83Ø MSG29   DB        27,25,3Ø,29,31+8ØH
     19 1E 1D 9F
                    ØØ84Ø ;                 record number out of range
1E78 1Ø             ØØ85Ø MSG3Ø   DB        16,47,52,26+8ØH
     2F 34 9A
                    ØØ86Ø ;                 directory full - can't extend file
1E7C 32             ØØ87Ø MSG31   DB        5Ø,12,44+ØCØH
     ØC EC
                    ØØ88Ø ;                 program not found
1E7F 17             ØØ89Ø MSG32   DB        23,48,25+ØCØH
     3Ø D9
                    ØØ9ØØ ;                 illegal drive number
1E82 Ø1             ØØ91Ø MSG33   DB        1,42,39,51+ØCØH
     2A 27 F3
                    ØØ92Ø ;                 no device space available
1E86 26             ØØ93Ø MSG34   DB        38,26,43,2+8ØH
     1A 2B 82
```

```
                    ØØ94Ø ;                 load file format error
1E8A 11             ØØ95Ø MSG35   DB        17,21+8ØH
     95
                    ØØ96Ø ;                 memory fault
1E8C ØD             ØØ97Ø MSG36   DB        13,38,9,4Ø,17+8ØH
     26 Ø9 28 91
                    ØØ98Ø ;                 attempted to load read only memory
1E91 17             ØØ99Ø MSG37   DB        23,46,13,22,26+8ØH
     2E ØD 16 9A
                    Ø1ØØØ ;                 illegal access attempted to protected file
1E96 1A             Ø1Ø1Ø MSG38   DB        26,12,53+ØCØH
     ØC F5
                    Ø1Ø2Ø ;                 file not open
1E99 2A             Ø1Ø3Ø MSG39   DB        42,45,54+8ØH
     2D B6
                    Ø1Ø4Ø ;                 device in use
1E9C 16             Ø1Ø5Ø MSG4Ø   DB        22,15,42+8ØH
     ØF AA
                    Ø1Ø6Ø ;                 protected system device
1E9F 1A             Ø1Ø7Ø MSG41   DB        26,57,53!ØCØH
     39 F5
                    Ø1Ø8Ø ;                 file already open
1EA2 18             Ø1Ø9Ø MSG42   DB        24,27,58,53,21!ØCØH
     1B 3A 35 D5
                    Ø11ØØ ;                 logical record length open fault
1EA7 38             Ø111Ø MSG43   DB        56,2Ø,2!8ØH
     14 82
                    Ø112Ø ;                 SVC parameter error
1EAA 14             Ø113Ø MSG44   DB        2Ø,2!8ØH
     82
                    Ø114Ø ;                 Parameter error
1EAC 25             Ø115Ø MSG45   DB        37,2,33+8ØH
     Ø2 A1
                    Ø116Ø ;                 unknown error code
1EAF E67Ø           Ø117Ø BEGIN   AND       7ØH             ;What's the entry?
1EB1 C8             Ø118Ø         RET       Z               ;Back on zero
1EB2 F5             Ø119Ø         PUSH      AF
1EB3 3AØDØØ         Ø12ØØ         LD        A,(LSVC$)       ;Grab the last SVC
1EB6 325E2Ø         Ø121Ø         LD        (SVSVC+1),A     ;  and store for later
1EB9 F1             Ø122Ø         POP       AF
1EBA 22461F         Ø123Ø         LD        (EXTEND+1),HL   ;Value if extended error
1EBD E3             Ø124Ø         EX        (SP),HL         ;Grab return address
1EBE 22BF1F         Ø125Ø         LD        (ERR7+1),HL     ;  & stuff it
1EC1 E1             Ø126Ø         POP       HL
1EC2 F1             Ø127Ø         POP       AF              ;Pop off the error code
1EC3 E3             Ø128Ø         EX        (SP),HL         ;Get user ret address
1EC4 22241F         Ø129Ø         LD        (USRET+1),HL    ;  for long dsply
1EC7 E3             Ø13ØØ         EX        (SP),HL
1EC8 E5             Ø131Ø         PUSH      HL              ;Save regs
1EC9 D5             Ø132Ø         PUSH      DE
1ECA C5             Ø133Ø         PUSH      BC
1ECB 2AØBØØ         Ø134Ø         LD        HL,(SVCRET$)    ;Grab last SVC return
1ECE 22722Ø         Ø135Ø         LD        (SVRET+1),HL    ;  and save for dsply
1ED1 47             Ø136Ø         LD        B,A
1ED2 3A7CØØ         Ø137Ø         LD        A,(SFLAG$)      ;Test expanded-error
1ED5 E64Ø           Ø138Ø         AND       4ØH             ;  flag bit in system flag
1ED7 A8             Ø139Ø         XOR       B
1ED8 AØ             Ø14ØØ         AND       B
1ED9 47             Ø141Ø         LD        B,A             ;Xfer the result to B
1EDA F5             Ø142Ø         PUSH      AF              ;  & save for later
1EDB E63F           Ø143Ø         AND       3FH             ;Strip all but error #
```

```
1EDD 4F       Ø144Ø          LD      C,A             ;Place error code -> C
1EDE 216CØØ   Ø145Ø          LD      HL,CFLAG$       ;If system error suppress
1EE1 CB76     Ø146Ø          BIT     6,(HL)          ;  flag is set, don't
1EE3 C2B91F   Ø147Ø          JP      NZ,ERR6A        ;  display error message.
1EE6 CB7E     Ø148Ø          BIT     7,(HL)          ;If error-to-buffer is
1EE8 2ØØ5     Ø149Ø          JR      NZ,ERRØ         ;  set, put to user buf
1EEA 11ØØ1D   Ø15ØØ          LD      DE,SBUFF$
1EED 18Ø6     Ø151Ø          JR      ERRØA           ;Branch around force
1EEF CBFØ     Ø152Ø ERRØ     SET     6,B             ;Force buffer to abbrev
1EF1 F1       Ø153Ø          POP     AF
1EF2 CBF7     Ø154Ø          SET     6,A
1EF4 F5       Ø155Ø          PUSH    AF
1EF5 CB7Ø     Ø156Ø ERRØA    BIT     6,B             ;Expanded error display?
1EF7 Ø6ØØ     Ø157Ø          LD      B,Ø
1EF9 2Ø44     Ø158Ø          JR      NZ,ERR2         ;Jump if abbreviated
1EFB C5       Ø159Ø          PUSH    BC
1EFC 21C32Ø   Ø16ØØ          LD      HL,ERRMSG       ;Pt to "< ERRCOD =...
1EFF ØE11     Ø161Ø          LD      C,MLEN          ;  & move to buffer
1FØ1 EDBØ     Ø162Ø          LDIR
1FØ3 C1       Ø163Ø          POP     BC
1FØ4 EB       Ø164Ø          EX      DE,HL           ;Buffer ptr to HL
1FØ5 79       Ø165Ø          LD      A,C             ;Error code to A
1FØ6 362F     Ø166Ø          LD      (HL),2FH        ;Init for digit conv
1FØ8 34       Ø167Ø ERR1     INC     (HL)            ;Bump ASCII digit
1FØ9 D6ØA     Ø168Ø          SUB     1Ø              ;  count by 1Ø
1FØB 3ØFB     Ø169Ø          JR      NC,ERR1         ;Keep bumping 1Ø's digit
1FØD 2C       Ø17ØØ          INC     L               ;Bump buffer ptr
1FØE C63A     Ø171Ø          ADD     A,3AH           ;Convert rmndr to unit's
1F1Ø 77       Ø172Ø          LD      (HL),A          ;  & place in buffer
1F11 2C       Ø173Ø          INC     L               ;Bump to next pos
1F12 362C     Ø174Ø          LD      (HL),','        ;Stuff a comma & bump
1F14 2C       Ø175Ø          INC     L
1F15 362Ø     Ø176Ø          LD      (HL),' '        ;  & a space
1F17 2C       Ø177Ø          INC     L
1F18 EB       Ø178Ø          EX      DE,HL           ;Buffer ptr back to DE
1F19 C5       Ø179Ø          PUSH    BC
1F1A 21D42Ø   Ø18ØØ          LD      HL,ERRMSG1      ;"Returns to X'"
1F1D Ø1ØD ØØ  Ø181Ø          LD      BC,M1LEN
1F2Ø EDBØ     Ø182Ø          LDIR
1F22 EB       Ø183Ø          EX      DE,HL           ;HL back to buffer
1F23 11ØØØØ   Ø184Ø USRET    LD      DE,$-$          ;User ret address
1F26 CDBDØ7   Ø185Ø          CALL    @HEX16
1F29 3E27     Ø186Ø          LD      A,27H           ;"'"
1F2B 77       Ø187Ø          LD      (HL),A
1F2C 23       Ø188Ø          INC     HL
1F2D 36ØA     Ø189Ø          LD      (HL),LF         ;End the line
1F2F 23       Ø19ØØ          INC     HL
1F3Ø C1       Ø191Ø          POP     BC
1F31 CB71     Ø192Ø          BIT     6,C             ;Extended error?
1F33 2ØØ9     Ø193Ø          JR      NZ,ERR6         ;Go if not
1F35 362A     Ø194Ø          LD      (HL),'*'        ;Make long msg look nice
1F37 23       Ø195Ø          INC     HL
1F38 362A     Ø196Ø          LD      (HL),'*'
1F3A 23       Ø197Ø          INC     HL
1F3B 362Ø     Ø198Ø          LD      (HL),' '
1F3D 23       Ø199Ø          INC     HL
1F3E EB       Ø2ØØØ ERR6     EX      DE,HL           ;DE back to nxt buff line
1F3F 79       Ø2Ø1Ø ERR2     LD      A,C
1F4Ø FE3F     Ø2Ø2Ø          CP      63              ;"Extended error"?
1F42 2Ø23     Ø2Ø3Ø          JR      NZ,ERR2A
              Ø2Ø4Ø ;
```

```
                02050 ;       Do extended error only
                02060 ;
1F44 D5         02070         PUSH    DE              ;Save buffer ptr
1F45 110000     02080 EXTEND  LD      DE,$-$          ;Ext. error value fm HL
1F48 21BD20     02090         LD      HL,EXT$ERR+26
1F4B CDBD07     02100         CALL    @HEX16
1F4E 21A320     02110         LD      HL,EXT$ERR      ;Point to error msg
1F51 D1         02120         POP     DE              ;Recvr buffer
1F52 E5         02130         PUSH    HL              ;Save msg start
1F53 C5         02140         PUSH    BC
1F54 012000     02150         LD      BC,M2LEN        ;Len of error
1F57 EDB0       02160         LDIR                    ;Move into buffer
1F59 C1         02170         POP     BC
1F5A 216C00     02180         LD      HL,CFLAG$       ;See if to user buffer
1F5D CB7E       02190         BIT     7,(HL)
1F5F CBBE       02200         RES     7,(HL)          ;Don't logot if so
1F61 E1         02210         POP     HL
1F62 CC0005     02220         CALL    Z,@LOGOT
1F65 1852       02230         JR      ERR6A           ;  and exit
                02240 ;
                02250 ;       Do regular (non-extended) error
                02260 ;
1F67 3E2D       02270 ERR2A   LD      A,45            ;If error code is > 43,
1F69 B9         02280         CP      C               ;  then set to 44 (max)
1F6A D5         02290         PUSH    DE              ;Save ptr to 1st char
1F6B 3001       02300         JR      NC,ERR3
1F6D 4F         02310         LD      C,A
1F6E 217821     02320 ERR3    LD      HL,CODTAB       ;Pt to start of code
1F71 09         02330         ADD     HL,BC           ;  address table & index
1F72 6E         02340         LD      L,(HL)          ;P/u lo-order vector
1F73 261E       02350         LD      H,MSG0<-8       ;Set hi-order vector
                02360 ;
                02370 ;       HL now points to sentence table
                02380 ;
1F75 7E         02390 ERR5    LD      A,(HL)          ;P/u word offset
1F76 E63F       02400         AND     3FH             ;  & strip any flags
1F78 47         02410         LD      B,A             ;Xfer word # to reg B
1F79 E5         02420         PUSH    HL              ;Save sentence pointer
1F7A 21A621     02430         LD      HL,WORDS        ;Dictionary start
1F7D 7E         02440 LP1     LD      A,(HL)          ;Scan through the table
1F7E 07         02450         RLCA                    ;  counting words (bit 7
1F7F 23         02460         INC     HL              ;  denotes word end)
1F80 30FB       02470         JR      NC,LP1          ;  until requested word
1F82 05         02480         DEC     B               ;  is reached
1F83 20F8       02490         JR      NZ,LP1
                02500 ;
                02510 ;       Found the start of the desired word
                02520 ;
1F85 7E         02530 LP2     LD      A,(HL)          ;Transfer the word until
1F86 07         02540         RLCA                    ;  bit 7 set (last char)
1F87 CB3F       02550         SRL     A               ;  while resetting bit-7
1F89 12         02560         LD      (DE),A          ;Stuff letter of word
1F8A 23         02570         INC     HL              ;  & bump pointers
1F8B 13         02580         INC     DE
1F8C 30F7       02590         JR      NC,LP2
1F8E 3E20       02600         LD      A,' '           ;Move a space into buffer
1F90 12         02610         LD      (DE),A
1F91 13         02620         INC     DE
1F92 E1         02630         POP     HL              ;Rcvr ptr to sentence
1F93 7E         02640         LD      A,(HL)          ;P/u this word byte
1F94 23         02650         INC     HL
```

```
1F95 07       02660          RLCA                    ;Was this the last word?
1F96 30DD     02670          JR      NC,ERR5         ;Loop if still more to go
1F98 E3       02680          EX      (SP),HL         ;Get ptr to 1st char
1F99 7E       02690          LD      A,(HL)
1F9A CBAF     02700          RES     5,A             ;Set it to UC
1F9C 77       02710          LD      (HL),A
1F9D E1       02720          POP     HL              ;Get back sentence ptr
1F9E F1       02730          POP     AF              ;Rcvr error code
1F9F F5       02740          PUSH    AF
1FA0 E5       02750          PUSH    HL              ;Save sentence ptr
1FA1 3E0D     02760          LD      A,CR
1FA3 12       02770          LD      (DE),A          ;Stuff end-of-line
1FA4 216C00   02780          LD      HL,CFLAG$       ;If to user buffer,
1FA7 CB7E     02790          BIT     7,(HL)          ;  then don't LOGOT
1FA9 CBBE     02800          RES     7,(HL)
1FAB 211D01   02810          LD      HL,SBUFF$       ;Display the line
1FAE CC0005   02820          CALL    Z,@LOGOT
1FB1 E1       02830          POP     HL
1FB2 F1       02840          POP     AF              ;Rcvr word index
1FB3 F5       02850          PUSH    AF
1FB4 CB77     02860          BIT     6,A             ;Test if a disk error
1FB6 CCC41F   02870          CALL    Z,DSPSPEC       ;Get filespec if it is
1FB9 F1       02880 ERR6A    POP     AF
1FBA C1       02890          POP     BC
1FBB D1       02900          POP     DE
1FBC E1       02910          POP     HL
1FBD B7       02920          OR      A               ;Ret to user if bit 7
1FBE FA0000   02930 ERR7     JP      M,0             ;  of error code is set
1FC1 C3081B   02940          JP      @ABORT          ;  else abort
              02950 ;
              02960 ;        Routine to display the filespec
              02970 ;
1FC4 DDE5     02980 DSPSPEC  PUSH    IX
1FC6 DD2A2400 02990          LD      IX,(JDCB$)      ;P/u FCB vector
1FCA 2B       03000          DEC     HL
1FCB CB76     03010          BIT     6,(HL)
1FCD 2030     03020          JR      NZ,DSPC2
1FCF DD4E06   03030          LD      C,(IX+6)        ;Device 1st char or drive
1FD2 DD4607   03040          LD      B,(IX+7)        ;Device 2nd char or DEC
1FD5 DDCB007E 03050          BIT     7,(IX+0)        ;Test if file or device
1FD9 205D     03060          JR      NZ,RCVSPEC      ;Jump if it is a file
1FDB 212821   03070          LD      HL,OPN$DCB
1FDE 79       03080 DSPC1    LD      A,C             ;Possible devspec, 1st char
1FDF FE41     03090          CP      'A'
1FE1 3815     03100          JR      C,DCBUNK        ;C=do unknown
1FE3 FE5B     03110          CP      'Z'+1
1FE5 3011     03120          JR      NC,DCBUNK       ;Again, go if bad
1FE7 78       03130          LD      A,B             ;Check 2nd character
1FE8 FE30     03140          CP      '0'
1FEA 380C     03150          JR      C,DCBUNK
1FEC FE5B     03160          CP      'Z'+1
1FEE 3008     03170          JR      NC,DCBUNK
1FF0 ED433A21 03180          LD      (OPN$DCB+18),BC ;Stuff the device name
1FF2          03190 DSPC1A   EQU     $-2
1FF4 DDE1     03200          POP     IX
1FF6 185F     03210          JR      RSPC6           ;Go display it
              03220 ;
1FF8 213D21   03230 DCBUNK   LD      HL,UNK$TYP
1FFB DDE1     03240          POP     IX
1FFD 1858     03250          JR      RSPC6
              03260 ;
```

```
1FFF DD4E01   03270 DSPC2   LD      C,(IX+1)        ;P/u 1st char or vector
2002 DD4602   03280         LD      B,(IX+2)        ;P/u 2nd char or vector
2005 DD7E00   03290         LD      A,(IX+0)
2008 21EB20   03300         LD      HL,DEV$NAM
200B 22F21F   03310         LD      (DSPC1A),HL     ;Change dsply message
200E 21E120   03320         LD      HL,DEV$EQ
2011 FE2A     03330         CP      '*'             ;If '*', go to device
2013 28C9     03340         JR      Z,DSPC1
2015 DDE5     03350         PUSH    IX              ;  else assume file
2017 E1       03360         POP     HL
2018 11F520   03370         LD      DE,FILE$EQ+7    ;Init "<file=...
201B 0618     03380         LD      B,24            ;Max filespec
201D 7E       03390 DSPC3   LD      A,(HL)          ;P/u file spec char
201E FE03     03400         CP      3               ;ETX?
2020 2811     03410         JR      Z,DSPC3A
2022 FE0D     03420         CP      CR              ;EOL?
2024 280D     03430         JR      Z,DSPC3A
2026 B7       03440         OR      A
2027 280A     03450         JR      Z,DSPC3A        ;Zero ok terminator, too.
2029 CD8320   03460         CALL    CHKASC          ;Check if an ASCII char
202C 38CA     03470         JR      C,DCBUNK        ;  and abort if not
202E 12       03480         LD      (DE),A
202F 13       03490         INC     DE
2030 23       03500         INC     HL
2031 10EA     03510         DJNZ    DSPC3           ;Loop until end
2033 21EE20   03520 DSPC3A  LD      HL,FILE$EQ
2036 181A     03530         JR      RSPC5
              03540 ;
              03550 ;       Routine to get recover the filespec
              03560 ;
2038 79       03570 RCVSPEC LD      A,C
2039 C630     03580         ADD     A,30H           ;Conv drive # to decimal
203B FE30     03590         CP      '0'             ;Valid drive?
203D 38B9     03600         JR      C,DCBUNK
203F FE38     03610         CP      '8'
2041 30B5     03620         JR      NC,DCBUNK
2043 321D21   03630         LD      (OPN$FCB+16),A
2046 78       03640         LD      A,B             ;Dec into A
2047 212421   03650         LD      HL,OPN$FCB+23   ;Pt into msg string
204A CDC207   03660         CALL    @HEX8           ;  and convert it.
204D EB       03670         EX      DE,HL           ;DE back to buff end
204E 210D21   03680         LD      HL,OPN$FCB
2051 13       03690         INC     DE
2052 3E0D     03700 RSPC5   LD      A,CR            ;Close with EOL
2054 12       03710         LD      (DE),A
2055 DDE1     03720         POP     IX
2057 CD0005   03730 RSPC6   CALL    @LOGOT          ;Log it
              03740 ;
              03750 ;       Build the SVC info line
              03760 ;
205A 117221   03770         LD      DE,LILBUF       ;Tempy for hexdec
205D 3E00     03780 SVSVC   LD      A,$-$           ;P/u the stored last svc
205F 6F       03790         LD      L,A
2060 2600     03800         LD      H,0             ;  into HL for conv
2062 CDF606   03810         CALL    @HEXDEC
2065 115821   03820         LD      DE,SVC$NUM+11
2068 CD9420   03830         CALL    EDEC
206B 3E03     03840         LD      A,3             ;Then put in ETX
206D 12       03850         LD      (DE),A
              03860 ;
206E 216C21   03870         LD      HL,SVC$RET+16   ;Now, do last svc return
```

```
2071 110000    03880 SVRET   LD      DE,$-$
2074 CDBD07    03890         CALL    @HEX16
2077 214D21    03900         LD      HL,SVC$NUM
207A CD0005    03910         CALL    @LOGOT
207D 215C21    03920         LD      HL,SVC$RET
2080 C30005    03930         JP      @LOGOT          ;Log it
               03940 ;
               03950 ;       Routine to check for valid chars
               03960 ;
2083 7E        03970 CHKASC  LD      A,(HL)          ;Xfer until 1st space
2084 FE2E      03980         CP      '.'
2086 D8        03990         RET     C               ;CF on ret = bad char
2087 FE3B      04000         CP      ':'+1
2089 3002      04010         JR      NC,CKASC1
208B 1805      04020         JR      CKASC2
208D FE41      04030 CKASC1  CP      'A'
208F D8        04040         RET     C
2090 FE5B      04050         CP      'Z'+1
2092 3F        04060 CKASC2  CCF
2093 C9        04070         RET
               04080 ;
2094 217221    04090 EDEC    LD      HL,LILBUF       ;Pt to conved decimal num.
2097 7E        04100 ED1     LD      A,(HL)
2098 B7        04110         OR      A
2099 C8        04120         RET     Z
209A FE20      04130         CP      ' '
209C 23        04140         INC     HL
209D 28F8      04150         JR      Z,ED1
209F 12        04160         LD      (DE),A          ;Store valid digit
20A0 13        04170         INC     DE
20A1 18F4      04180         JR      ED1
               04190 ;
               04200 ;
               04210 ;
20A3 2A        04220 EXT$ERR DB      '** Extended error, HL = X',27H,'xxxx',27H,CR
     2A 20 45 78 74 65 6E 64
     65 64 20 65 72 72 6F 72
     2C 20 48 4C 20 3D 20 58
     27 78 78 78 78 27 0D
0020           04230 M2LEN   EQU     $-EXT$ERR
20C3 0A        04240 ERRMSG  DB      LF,'** Error code = '
     2A 2A 20 45 72 72 6F 72
     20 63 6F 64 65 20 3D 20
0011           04250 MLEN    EQU     $-ERRMSG
20D4 52        04260 ERRMSG1 DB      'Returns to X',27H
     65 74 75 72 6E 73 20 74
     6F 20 58 27
000D           04270 M1LEN   EQU     $-ERRMSG1
20E1 44        04280 DEV$EQ  DB      'Device = *'
     65 76 69 63 65 20 3D 20
     2A
20EB 58        04290 DEV$NAM DB      'XX',CR
     58 0D
20EE 46        04300 FILE$EQ DB      'File = NNNNNNNN/EEE.PPPPPPPP:D',CR
     69 6C 65 20 3D 20 4E 4E
     4E 4E 4E 4E 4E 4E 2F 45
     45 45 2E 50 50 50 50 50
     50 50 50 3A 44 0D
210D 4F        04310 OPN$FCB DB      'Open FCB, Drive=n, DEC=   ',CR
     70 65 6E 20 46 43 42 2C
     20 44 72 69 76 65 3D 6E
```

```
     2C 2Ø 44 45 43 3D 2Ø 2Ø
     2Ø ØD
2128 4F       Ø432Ø OPN$DCB DB      'Open DCB, Device=*xx',CR
     7Ø 65 6E 2Ø 44 43 42 2C
     2Ø 44 65 76 69 63 65 3D
     2A 78 78 ØD
213D 55       Ø433Ø UNK$TYP DB      'Unknown FCB/DCB',CR
     6E 6B 6E 6F 77 6E 2Ø 46
     43 42 2F 44 43 42 ØD
214D 4C       Ø434Ø SVC$NUM DB      'Last SVC = nnn',3
     61 73 74 2Ø 53 56 43 2Ø
     3D 2Ø 6E 6E 6E Ø3
215C 2C       Ø435Ø SVC$RET DB      ', Returned to X',27H,'xxxx',27H,CR
     2Ø 52 65 74 75 72 6E 65
     64 2Ø 74 6F 2Ø 58 27 78
     78 78 78 27 ØD
              Ø436Ø ;
ØØØ5          Ø437Ø LILBUF  DS      5
2177 ØØ       Ø438Ø         DB      Ø
              Ø439Ø ;
              Ø44ØØ ;       Table points to low-order bytes of messages
              Ø441Ø ;
2178 Ø3       Ø442Ø CODTAB  DB      MSGØ&ØFFH,MSG1&ØFFH,MSG2&ØFFH,MSG3&ØFFH
     Ø5 ØA ØE
217C 12       Ø443Ø         DB      MSG4&ØFFH,MSG5&ØFFH,MSG6&ØFFH
     16 1C
217F 21       Ø444Ø         DB      MSG7&ØFFH,MSG8&ØFFH,MSG9&ØFFH
     26 29
2182 2E       Ø445Ø         DB      MSG1Ø&ØFFH,MSG11&ØFFH,MSG12&ØFFH,MSG13&ØFFH
     32 36 3A
2186 4Ø       Ø446Ø         DB      MSG14&ØFFH,MSG15&ØFFH,MSG16&ØFFH,MSG17&ØFFH
     45 48 4C
218A 4F       Ø447Ø         DB      MSG18&ØFFH,MSG19&ØFFH,MSG2Ø&ØFFH,MSG21&ØFFH
     52 55 58
218E 5B       Ø448Ø         DB      MSG22&ØFFH,MSG23&ØFFH,MSG24&ØFFH,MSG25&ØFFH
     5E 61 65
2192 68       Ø449Ø         DB      MSG26&ØFFH,MSG27&ØFFH,MSG28&ØFFH,MSG29&ØFFH
     6C 6F 73
2196 78       Ø45ØØ         DB      MSG3Ø&ØFFH,MSG31&ØFFH,MSG32&ØFFH,MSG33&ØFFH
     7C 7F 82
219A 86       Ø451Ø         DB      MSG34&ØFFH,MSG35&ØFFH,MSG36&ØFFH,MSG37&ØFFH
     8A 8C 91
219E 96       Ø452Ø         DB      MSG38&ØFFH,MSG39&ØFFH,MSG4Ø&ØFFH,MSG41&ØFFH
     99 9C 9F
21A2 A2       Ø453Ø         DB      MSG42&ØFFH,MSG43&ØFFH,MSG44&ØFFH,MSG45&ØFFH
     A7 AA AC
              Ø454Ø ;
              Ø455Ø ;       Word dictionary
              Ø456Ø ;
21A6 D2       Ø457Ø WORDS   DB      'R'!8ØH                 ;Start table with bit 7
21A7 6E       Ø458Ø         DB      'n','o'!8ØH             ;1
     EF
21A9 65       Ø459Ø         DB      'erro','r'!8ØH          ;2
     72 72 6F F2
21AE EF       Ø46ØØ         DB      'o'!8ØH                 ;3 extra word
21AF 7Ø       Ø461Ø         DB      'parit','y'!8ØH         ;4
     61 72 69 74 F9
21B5 64       Ø462Ø         DB      'durin','g'!8ØH         ;5
     75 72 69 6E E7
21BB 68       Ø463Ø         DB      'heade','r'!8ØH         ;6
     65 61 64 65 F2
```

```
21C1 64          Ø464Ø          DB      'dat','a'!8ØH           ;7
     61 74 E1
21C5 73          Ø465Ø          DB      'see','k'!8ØH           ;8
     65 65 EB
21C9 72          Ø466Ø          DB      'rea','d'!8ØH           ;9
     65 61 E4
21CD 77          Ø467Ø          DB      'writ','e'!8ØH          ;1Ø
     72 69 74 E5
21D2 6C          Ø468Ø          DB      'los','t'!8ØH           ;11
     6F 73 F4
21D6 6E          Ø469Ø          DB      'no','t'!8ØH            ;12
     6F F4
21D9 61          Ø47ØØ          DB      'attempted t','o'!8ØH   ;13
     74 74 65 6D 7Ø 74 65 64
     2Ø 74 EF
21E5 6C          Ø471Ø          DB      'locked/delete','d'!8ØH ;14
     6F 63 6B 65 64 2F 64 65
     6C 65 74 65 E4
21F3 73          Ø472Ø          DB      'syste','m'!8ØH         ;15
     79 73 74 65 ED
21F9 64          Ø473Ø          DB      'director','y'!8ØH      ;16
     69 72 65 63 74 6F 72 F9
22Ø2 6D          Ø474Ø          DB      'memor','y'!8ØH         ;17
     65 6D 6F 72 F9
22Ø8 6F          Ø475Ø          DB      'o','n'!8ØH             ;18
     EE
22ØA 64          Ø476Ø          DB      'dis','k'!8ØH           ;19
     69 73 EB
22ØE 7Ø          Ø477Ø          DB      'paramete','r'!8ØH      ;2Ø
     61 72 61 6D 65 74 65 F2
2217 66          Ø478Ø          DB      'faul','t'!8ØH          ;21
     61 75 6C F4
221C 7Ø          Ø479Ø          DB      'protecte','d'!8ØH      ;22
     72 6F 74 65 63 74 65 E4
2225 69          Ø48ØØ          DB      'illega','l'!8ØH        ;23
     6C 6C 65 67 61 EC
222C 6C          Ø481Ø          DB      'logica','l'!8ØH        ;24
     6F 67 69 63 61 EC
2233 6E          Ø482Ø          DB      'numbe','r'!8ØH         ;25
     75 6D 62 65 F2
2239 66          Ø483Ø          DB      'fil','e'!8ØH           ;26
     69 6C E5
223D 72          Ø484Ø          DB      'recor','d'!8ØH         ;27
     65 63 6F 72 E4
2243 65          Ø485Ø          DB      'en','d'!8ØH            ;28
     6E E4
2246 6F          Ø486Ø          DB      'o','f'!8ØH             ;29
     E6
2248 6F          Ø487Ø          DB      'ou','t'!8ØH            ;3Ø
     75 F4
224B 72          Ø488Ø          DB      'rang','e'!8ØH          ;31
     61 6E 67 E5
225Ø 65          Ø489Ø          DB      'encountere','d'!8ØH    ;32
     6E 63 6F 75 6E 74 65 72
     65 E4
225B 63          Ø49ØØ          DB      'cod','e'!8ØH           ;33
     6F 64 E5
225F 47          Ø491Ø          DB      'GA','T'!8ØH            ;34
     41 D4
2262 48          Ø492Ø          DB      'HI','T'!8ØH            ;35
     49 D4
```

```
2265 F9          Ø493Ø          DB      'y'!8ØH                 ;36
2266 75          Ø494Ø          DB      'unknow','n'!8ØH        ;37
     6E 6B 6E 6F 77 EE
226D 6C          Ø495Ø          DB      'loa','d'!8ØH           ;38
     6F 61 E4
2271 73          Ø496Ø          DB      'spac','e'!8ØH          ;39
     7Ø 61 63 E5
2276 6F          Ø497Ø          DB      'onl','y'!8ØH           ;4Ø
     6E 6C F9
227A 6E          Ø498Ø          DB      'nam','e'!8ØH           ;41
     61 6D E5
227E 64          Ø499Ø          DB      'devic','e'!8ØH         ;42
     65 76 69 63 E5
2284 66          Ø5ØØØ          DB      'forma','t'!8ØH         ;43
     6F 72 6D 61 F4
228A 66          Ø5Ø1Ø          DB      'foun','d'!8ØH          ;44
     6F 75 6E E4
228F 69          Ø5Ø2Ø          DB      'i','n'!8ØH             ;45
     EE
2291 61          Ø5Ø3Ø          DB      'acces','s'!8ØH         ;46
     63 63 65 73 F3
2297 66          Ø5Ø4Ø          DB      'ful','l'!8ØH           ;47
     75 6C EC
229B 64          Ø5Ø5Ø          DB      'driv','e'!8ØH          ;48
     72 69 76 E5
22AØ 64          Ø5Ø6Ø          DB      'denie','d'!8ØH         ;49
     65 6E 69 65 E4
22A6 7Ø          Ø5Ø7Ø          DB      'progra','m'!8ØH        ;5Ø
     72 6F 67 72 61 ED
22AD 61          Ø5Ø8Ø          DB      'availabl','e'!8ØH      ;51
     76 61 69 6C 61 62 6C E5
22B6 2D          Ø5Ø9Ø          DB      '- can''t exten','d'!8ØH        ;52
     2Ø 63 61 6E 27 74 2Ø 65
     78 74 65 6E E4
22C4 6F          Ø51ØØ          DB      'ope','n'!8ØH           ;53
     7Ø 65 EE
22C8 75          Ø511Ø          DB      'us','e'!8ØH            ;54
     73 E5
22CB 6F          Ø512Ø          DB      'o','r'!8ØH             ;55
     F2
22CD 53          Ø513Ø          DB      'SV','C'!8ØH            ;56
     56 C3
22DØ 61          Ø514Ø          DB      'alread','y'!8ØH        ;57
     6C 72 65 61 64 F9
22D7 6C          Ø515Ø          DB      'lengt','h'!8ØH         ;58
     65 6E 67 74 E8
22DD             Ø516Ø LAST     EQU     $
                 Ø517Ø          IFGT    $,DIRBUF$
                 Ø518Ø          ERR     'Module too big'
                 Ø519Ø          ENDIF
23FE             Ø52ØØ          ORG     MAXCOR$-2
23FE DDØ4        Ø521Ø          DW      LAST-SYS4       ;Overlay length
                 Ø522Ø ;
1EØØ             Ø523Ø          END     SYS4
```

$A1	Ø3B7	$A2	Ø3B8	$A3	Ø3B9		
$CKEOF	147Ø	@$SYS	Ø8FØ	@@1	ØØØØ		
@@2	ØØØØ	@@3	ØØØØ	@@4	ØØØØ		
@ABORT	1BØ8	@ADTSK	1CDA	@BANK	Ø877		
@BKSP	1486	@BREAK	196F	@BYTEIO	13ØØ		
@CHNIO	Ø689	@CKBRKC	Ø553	@CKDRV	1993		
@CKEOF	158F	@CKTSK	1CF5	@CLOSE	1999		
@CLS	Ø545	@CMNDI	197E	@CMNDR	197B		
@CTL	Ø623	@DATE	Ø7A8	@DBGHK	199F		
@DCINIT	19CØ	@DCRES	19C4	@DCSTAT	19B5		
@DCTBYT	1A2B	@DEBUG	19AØ	@DECHEX	Ø3E1		
@DIRCYL	18F7	@DIRRD	18BB	@DIRWR	18Ø3		
@DIV16	Ø6E3	@DIV8	1927	@DODIR	19AF		
@DOKEY	19A9	@DSP	Ø642	@DSPLY	Ø52D		
@ERROR	1BØF	@EXIT	1BØB	@FEXT	1984		
@FLAGS	196A	@FNAME	199C	@FRENCH	ØØØØ		
@FSPEC	1981	@GATRD	1874	@GATWR	1875		
@GERMAN	ØØØØ	@GET	Ø638	@GTDCB	199Ø		
@GTDCT	1A1E	@GTMOD	19B2	@HDFMT	19E4		
@HEX16	Ø7BD	@HEX8	Ø7C2	@HEXDEC	Ø6F6		
@HIGH$	1948	@HITRD	1897	@HITWR	1898		
@HZ5Ø	ØØØØ	@ICNFG	ØØ86	@INIT	198D		
@INTL	ØØØØ	@IPL	1BF2	@JCL	Ø63Ø		
@KBD	Ø635	@KEY	Ø628	@KEYIN	Ø585		
@KITSK	ØØ89	@KLTSK	1CDØ	@LOAD	1B38		
@LOC	14B3	@LOF	14DE	@LOGER	Ø5Ø3		
@LOGOT	Ø5ØØ	@MOD2	ØØØØ	@MOD4	FFFF		
@MSG	Ø53Ø	@MUL16	Ø6C9	@MUL8	19ØA		
@NMI	ØØ66	@OPEN	198A	@OPREG	ØØ84		
@PARAM	1987	@PAUSE	Ø382	@PEOF	14A2		
@POSN	1434	@PRINT	Ø528	@PRT	Ø63D		
@PUT	Ø645	@RAMDIR	19AC	@RDHDR	19D8		
@RDSEC	19F4	@RDSSC	18D8	@RDTRK	19EØ		
@READ	1513	@REMOVE	19A6	@RENAME	1996		
@REW	149B	@RMTSK	1CD7	@RPTSK	1CEB		
@RREAD	1473	@RSLCT	19D4	@RSTØØ	ØØØØ		
@RSTØ8	ØØØ8	@RST1Ø	ØØ1Ø	@RST18	ØØ18		
@RST2Ø	ØØ2Ø	@RST28	ØØ28	@RST3Ø	ØØ3Ø		
@RST38	ØØ38	@RSTNMI	ØFE9	@RSTOR	19C8		
@RSTREG	Ø68Ø	@RUN	1B1D	@RWRIT	13AD		
@SEEK	19DØ	@SEEKSC	1421	@SKIP	143Ø		
@SLCT	19BC	@SOUND	Ø392	@STEPI	19CC		
@TIME	Ø78D	@USA	FFFF	@VDCTL	ØB99		
@VDCTL3	ØD38	@VER	156Ø	@VRSEC	19DC		
@WEOF	14EC	@WHERE	1979	@WRITE	1531		
@WRSEC	19E8	@WRSSC	19EC	@WRTRK	19FØ		
@_VDCTL	ØD42	ADDR_2_ROWCOL	ØDF1	AFLAG$	ØØ6A		
AUTO?	1FF1	BAR$	Ø2Ø1	BEGIN	1EAF		
BOOTST$	439D	BREAK?	1C6Ø	BRKVEC$	1C88		
BUR$	Ø2ØØ	CASHK$	ØA7B	CFCB$	ØØEØ		
CFGFCB$	ØØEØ	CFLAG$	ØØ6C	CHKASC	2Ø83		
CKASC1	2Ø8D	CKASC2	2Ø92	CKMOD@	1A7F		
CKOPEN@	1568	CODTAB	2178	CONFIG$	2Ø3F		
CORE$	Ø3ØØ	CR	ØØØD	CRTBGN$	F8ØØ		
CYL_GRN	16AE	D@FBYT8	1A26	DATE$	ØØ33		
DAYTBL$	Ø4C7	DBGSV$	ØØAØ	DCBKL$	ØØ31		
DCBUNK	1FF8	DCT$	Ø47Ø	DCTBYT8@	1A29		
DCTFLD@	1A34	DEV$EQ	2ØE1	DEV$NAM	2ØEB		
DFLAG$	ØØ6D	DIRBUF$	23ØØ	DIS_DO_RAM	Ø846		
DODATA$	ØB94	DODCB$	Ø21Ø	DO_CONTROL	ØC44		

DO_DSPCHAR	ØCB8	DO_INVERT_DIS	ØC8C	DO_INVERT_ENA	ØC89
DO_INVERT_OFF	ØC9B	DO_MASK	ØØØØ	DO_RET	ØBCB
DO_RET1	ØBCC	DO_SCROLL	ØCCE	DO_TABS	ØBEA
DSKTYP$	Ø4CØ	DSPC1	1FDE	DSPC1A	1FF2
DSPC2	1FFF	DSPC3	2Ø1D	DSPC3A	2Ø33
DSPSPEC	1FC4	DTPMT$	Ø4C2	DVREND$	ØFF4
DVRHI$	Ø2Ø6	ED1	2Ø97	EDEC	2Ø94
EFLAG$	ØØ6E	ENADIS_DO_RAM	Ø817	ERRØ	1EEF
ERRØA	1EF5	ERR1	1FØ8	ERR2	1F3F
ERR2A	1F67	ERR3	1F6E	ERR5	1F75
ERR6	1F3E	ERR6A	1FB9	ERR7	1FBE
ERRMSG	2ØC3	ERRMSG1	2ØD4	EXT$ERR	2ØA3
EXTDBG$	19A4	EXTEND	1F45	FDDINT$	ØØØE
FEMSK$	ØØ6F	FILE$EQ	2ØEE	FLGTAB$	ØØ6A
GET_@_ROWCOL	ØDAE	HERTZ$	Ø75Ø	HIGH$	Ø4ØE
HKRES$	1A6C	IFLAG$	ØØ72	INBUF$	Ø42Ø
INTIM$	ØØ3C	INTMSK$	ØØ3D	INTVC$	ØØ3E
JCLCB$	Ø2Ø3	JDCB$	ØØ24	JFCB$	ØØCØ
JLDCB$	Ø23Ø	JRET$	ØØ26	KCK@	Ø7D6
KFLAG$	ØØ74	KIDATA$	Ø8FC	KIDCB$	Ø2Ø8
LAST	22DD	LBANK$	Ø2Ø2	LDRV$	ØØ23
LF	ØØØA	LFLAG$	ØØ75	LILBUF	2172
LNKFCB@	1566	LOW$	ØØ1E	LP1	1F7D
LP2	1F85	LSVC$	ØØØD	M1LEN	ØØØD
M2LEN	ØØ2Ø	MAXCOR$	24ØØ	MAXDAY$	Ø4Ø1
MINCOR$	3ØØØ	MLEN	ØØ11	MODOUT$	ØØ76
MONTBL$	Ø4DC	MSGØ	1EØ3	MSG1	1EØ5
MSG1Ø	1E2E	MSG11	1E32	MSG12	1E36
MSG13	1E3A	MSG14	1E4Ø	MSG15	1E45
MSG16	1E48	MSG17	1E4C	MSG18	1E4F
MSG19	1E52	MSG2	1EØA	MSG2Ø	1E55
MSG21	1E58	MSG22	1E5B	MSG23	1E5E
MSG24	1E61	MSG25	1E65	MSG26	1E68
MSG27	1E6C	MSG28	1E6F	MSG29	1E73
MSG3	1EØE	MSG3Ø	1E78	MSG31	1E7C
MSG32	1E7F	MSG33	1E82	MSG34	1E86
MSG35	1E8A	MSG36	1E8C	MSG37	1E91
MSG38	1E96	MSG39	1E99	MSG4	1E12
MSG4Ø	1E9C	MSG41	1E9F	MSG42	1EA2
MSG43	1EA7	MSG44	1EAA	MSG45	1EAC
MSG5	1E16	MSG6	1E1C	MSG7	1E21
MSG8	1E26	MSG9	1E29	NFLAG$	ØØ77
OPN$DCB	2128	OPN$FCB	21ØD	OPREG$	ØØ78
OPREG_SV_AREA	Ø86E	OPREG_SV_PTR	Ø835	ORARET@	14DC
OSRLS$	ØØ3B	OSVER$	ØØ85	OVRLY$	ØØ69
PAKNAM$	Ø41Ø	PAUSE@	Ø382	PCSAVE$	Ø7AF
PDRV$	ØØ1B	PHIGH$	ØØ1C	PRDCB$	Ø218
PUTA@DE	ØDCD	PUT_@	ØDCA	PUT_@_ROWCOL	ØDC6
RCVSPEC	2Ø38	RFLAG$	ØØ7B	ROWCOL_2_ADDR	ØDDØ
RSPC5	2Ø52	RSPC6	2Ø57	RST38@	1BFF
RSTOR$	Ø4C4	RWRIT@	13A2	S1DCB$	Ø238
SBUFF$	1DØØ	SET@EXEC	1A79	SET_SCROLL	ØCF3
SFCB$	ØØ8C	SFLAG$	ØØ7C	SIDCB$	Ø22Ø
SODCB$	Ø228	SPACE4$	2142	STACK$	Ø38Ø
START$	ØØØØ	SVC$NUM	214D	SVC$RET	215C
SVCRET$	ØØØB	SVCTAB$	Ø1ØØ	SVRET	2Ø71
SVSVC	2Ø5D	SYS4	1EØØ	SYSERR$	1B13
TCB$	ØØ4E	TFLAG$	ØØ7D	TIME$	ØØ2D
TIMER$	ØØ2C	TIMSL$	ØØ2B	TIMTSK$	Ø713
TMPMT$	Ø4C3	TRACE_INT	Ø7B1	TYPHK$	ØA8F
TYPTSK$	ØB26	UNK$TYP	213D	USRET	1F23

```
USTOR$          ØØ13 VFLAG$          ØØ7F WORDS          21A6
WRINT$          ØØ8Ø ZERO$           Ø4Ø1 ZEROA@         13AØ

1EØØ is the transfer address
ØØØØØ Total errors
```

NOTES:

SYS5 is the primary system debugger. It is activated by the system, by the break key, or by the SVC @DEBUG. During assembly, SYS5 is cross referenced to produce an EQUate file used in assembling SYS9, the extended debugger.

```
                00100 ;SYS5/ASM - LS-DOS 6.2
0000            00110         TITLE   <SYS5 - LS-DOS 6.2>
                00120 *LIST   OFF                     ;Get SYSØ/EQU
                00140 *LIST   ON
0000            00150 *GET    COPYCOM:3               ;Copyright message
                03010 ; COPYCOM - File for Copyright COMment block
                03020 ;
0000            03030         COM     '<*(C) 1982,83,84 by LSI*>'
                03040 ;
                00160 ;
0000            00170 *GET    SYS5A:3
                03050 ;SYS5A/ASM - LS-DOS 6.2
                03060 ;
00A0            03070         ORG     0A0H
                03080 ;
                03090 ;       References to save area in lowcore
                03100 ;
0001            03110 SAVONE  DS      1
0001            03120 SAVTWO  DS      1
0001            03130         DS      1               ;Space for saved byte (1)
0002            03140 NXTADR  DS      2
0001            03150 NXTBYT  DS      1
0002            03160 DSPADR  DS      2
0002            03170 AFREG   DS      2               ;AF  Register save area
0002            03180         DS      2               ;BC
0002            03190         DS      2               ;DE
0002            03200 HLREG   DS      2               ;HL
0008            03210         DS      8               ;AF', BC', DE', HL'
0002            03220 IXREG   DS      2               ;IX
0002            03230 IYREG   DS      2               ;IY
0001            03240 SPREG   DS      1               ;SP
0001            03250 REGSAV  DS      1
0002            03260 PCREG   DS      2               ;PC
                03270 ;
1E00            03280         ORG     1E00H
                03290 ;
1E00 E670       03300 SYS5    AND     70H             ;If entry = Ø, return
1E02 C8         03310         RET     Z
1E03 F1         03320         POP     AF              ;Discard return to SYSØ
1E04 F1         03330         POP     AF              ;Get original reg-AF
1E05 F5         03340         PUSH    AF
1E06 FDE5       03350         PUSH    IY              ;Save remaining regs
1E08 DDE5       03360         PUSH    IX
1E0A 08         03370         EX      AF,AF'
1E0B D9         03380         EXX
1E0C E5         03390         PUSH    HL
1E0D D5         03400         PUSH    DE
1E0E C5         03410         PUSH    BC
1E0F F5         03420         PUSH    AF
1E10 08         03430         EX      AF,AF'
1E11 D9         03440         EXX
1E12 E5         03450         PUSH    HL
1E13 D5         03460         PUSH    DE
1E14 C5         03470         PUSH    BC
1E15 F5         03480         PUSH    AF
1E16 210000     03490         LD      HL,0
1E19 39         03500         ADD     HL,SP           ;Place SP address into HL
1E1A 11A800     03510         LD      DE,AFREG
1E1D 011800     03520         LD      BC,24           ;Move the 24 bytes saved
1E20 EDB0       03530         LDIR
1E22 22BC00     03540         LD      (SPREG),HL
```

```
1E25 F9       Ø355Ø          LD      SP,HL
1E26 2ABEØØ   Ø356Ø          LD      HL,(PCREG)
1E29 2B       Ø357Ø          DEC     HL
1E2A 7E       Ø358Ø          LD      A,(HL)          ;P/u the byte at PC
1E2B FEF7     Ø359Ø          CP      ØF7H            ;  & check for breakpoint
1E2D 2ØØ3     Ø36ØØ          JR      NZ,$?1          ;Go if not a breakpoint
1E2F 22BEØØ   Ø361Ø          LD      (PCREG),HL
              Ø362Ø ;
              Ø363Ø ;        This next routine picks up the data stored in the
              Ø364Ø ;        instruction storage areas used to hold the
              Ø365Ø ;        address & byte of the inserted RST's used to
              Ø366Ø ;        control the single step mode. If the address
              Ø367Ø ;        save area is zero, then an RST was not inserted.
              Ø368Ø ;        Two areas are needed because DEBUG inserts
              Ø369Ø ;        RST 48's at both CALL origin & destination.
              Ø37ØØ ;
1E32 21AØØØ   Ø371Ø $?1      LD      HL,SAVONE
1E35 Ø6Ø2     Ø372Ø          LD      B,2             ;Set up loop for 2 areas
1E37 AF       Ø373Ø $?2      XOR     A               ;Clear register A & flags
1E38 5E       Ø374Ø          LD      E,(HL)          ;P/u the next 2 bytes
1E39 77       Ø375Ø          LD      (HL),A          ;  (where an address
1E3A 23       Ø376Ø          INC     HL              ;  would be stored) while
1E3B 56       Ø377Ø          LD      D,(HL)          ;  simultaneously setting
1E3C 77       Ø378Ø          LD      (HL),A          ;  the save area to zero
1E3D 23       Ø379Ø          INC     HL
1E3E 7B       Ø38ØØ          LD      A,E             ;Ck if the area was zero
1E3F B2       Ø381Ø          OR      D
1E4Ø 28Ø7     Ø382Ø          JR      Z,$?3           ;If zero, no RST entry
1E42 1A       Ø383Ø          LD      A,(DE)          ;Address save <> zero,
1E43 FEF7     Ø384Ø          CP      ØF7H            ;  ck byte for RST 48
1E45 2ØØ2     Ø385Ø          JR      NZ,$?3
1E47 7E       Ø386Ø          LD      A,(HL)          ;  Was RST 48, restore
1E48 12       Ø387Ø          LD      (DE),A          ;  the program byte
1E49 23       Ø388Ø $?3      INC     HL
1E4A 1ØEB     Ø389Ø          DJNZ    $?2             ;Loop thru 2 save areas
1E4C ED7BBCØØ Ø39ØØ CMND     LD      SP,(SPREG)      ;Set up the stack
1E5Ø CDD51E   Ø391Ø          CALL    WRREGS          ;  & display normal CRT
1E53 21ØØ1Ø   Ø392Ø          LD      HL,16<8!Ø       ;Move cursor to 16,Ø
1E56 Ø6Ø3     Ø393Ø          LD      B,3             ;Command
1E58 3EØF     Ø394Ø          LD      A,15            ;Svc @VDCTL
1E5A EF       Ø395Ø          RST     28H             ;Set cursor
1E5B CDC921   Ø396Ø          CALL    INPUT@          ;Get command
1E5E FE67     Ø397Ø          CP      'g'             ;Goto AAAA,(BBBB(,CCCC))
1E6Ø CA821F   Ø398Ø          JP      Z,CMD_G
1E63 214C1E   Ø399Ø          LD      HL,CMND         ;Set up a return branch
1E66 E5       Ø4ØØØ          PUSH    HL
1E67 FE73     Ø4Ø1Ø          CP      's'             ;Set CRT to full screen?
1E69 2832     Ø4Ø2Ø          JR      Z,CMD_S
1E6B FE3B     Ø4Ø3Ø          CP      ';'             ;Inc CRT one page?
1E6D 2842     Ø4Ø4Ø          JR      Z,CMD_INC
1E6F FE2D     Ø4Ø5Ø          CP      '-'             ;Dec CRT one page?
1E71 2856     Ø4Ø6Ø          JR      Z,CMD_DEC
1E73 FE6F     Ø4Ø7Ø          CP      'o'             ;Out to DOS
1E75 2857     Ø4Ø8Ø          JR      Z,CMD_O
1E77 FE63     Ø4Ø9Ø          CP      'c'             ;Single step with CALL?
1E79 28Ø6     Ø41ØØ          JR      Z,CMD_C
1E7B FE64     Ø411Ø          CP      'd'             ;Display AAAA <space>
1E7D 282C     Ø412Ø          JR      Z,CMD_D
1E7F FE69     Ø413Ø          CP      'i'             ;Single step?
1E81 CA8B2Ø   Ø414Ø CMD_C    JP      Z,CMD_CI
1E84 FE61     Ø415Ø          CP      'a'             ;ASCII modify memory?
```

```
1E86 CAD61F   04160          JP      Z,CMD_AH
1E89 FE68     04170          CP      'h'             ;Hex modify memory AAAA?
1E8B CAD61F   04180          JP      Z,CMD_AH
1E8E FE72     04190          CP      'r'             ;Modify reg pair RP DDDD?
1E90 CA3F20   04200          JP      Z,CMD_R
1E93 FE75     04210          CP      'u'             ;Dynamic display update?
1E95 280A     04220          JR      Z,CMD_U
1E97 FE78     04230          CP      'x'             ;Display register format?
1E99 C23F22   04240          JP      NZ,BLOCK        ;Try extra commands
              04250 ;
              04260 ;        Command X - Normal display mode
              04270 ;
1E9C AF       04280 CMD_X    XOR     A
1E9D 32A100   04290 CMD_S    LD      (SAVTWO),A      ;Show not full screen
1EA0 C9       04300          RET
              04310 ;
              04320 ;        Command U - Continuously update display
              04330 ;
1EA1 CD3506   04340 CMD_U    CALL    @KBD            ;Scan keyboard
1EA4 B7       04350          OR      A               ;Character entered?
1EA5 C0       04360          RET     NZ              ;Return to CMND if so
1EA6 CDD51E   04370          CALL    WRREGS          ;  else refresh display
1EA9 18F6     04380          JR      CMD_U           ;  & loop
              04390 ;
              04400 ;        Command D - Display memory at address NNNN
              04410 ;
1EAB CDE421   04420 CMD_D    CALL    HEXIN@
1EAE C8       04430          RET     Z               ;Ret to CMND if no char
1EAF 1814     04440          JR      $?6             ;  else set DSPADR to
              04450                                  ;  new address in HL
              04460 ;
              04470 ;        Command ; - Increment memory display one block
              04480 ;
1EB1 014000   04490 CMD_INC  LD      BC,64           ;Init for 64-byte block
1EB4 2AA600   04500 $?4      LD      HL,(DSPADR)     ;P/u current display addr
1EB7 3AA100   04510          LD      A,(SAVTWO)      ; =0 -> Normal disp mode
              04520                                  ;<>0 -> Full disp mode
1EBA B7       04530          OR      A
1EBB 2807     04540          JR      Z,$?5
1EBD 0E00     04550          LD      C,0             ;Zero out low order to
              04560                                  ;  provide inc or dec of
              04570                                  ;  256 bytes (full disp)
1EBF 78       04580          LD      A,B             ;B=00 -> inc 1 page,
1EC0 B7       04590          OR      A               ;  make BC = 256
1EC1 2001     04600          JR      NZ,$?5          ;B=FF -> Dec 1 page,
1EC3 04       04610          INC     B               ;  just add
1EC4 09       04620 $?5      ADD     HL,BC           ;HL now points to
1EC5 22A600   04630 $?6      LD      (DSPADR),HL     ;  new display address
1EC8 C9       04640          RET
              04650 ;
              04660 ;        Command - - Decrement memory display 1 block
              04670 ;
1EC9 01C0FF   04680 CMD_DEC  LD      BC,0FFC0H       ;Init to 64-byte dec
1ECC 18E6     04690          JR      $?4
              04700 ;
              04710 ;        Command O - Exit to DOS
              04720 ;
1ECE CDC921   04730 CMD_O    CALL    INPUT@          ;Fetch valid terminator
1ED1 D0       04740          RET     NC              ;Back if bad char
1ED2 C30B1B   04750          JP      @EXIT           ;Else exit to DOS
              04760 ;
```

```
                    Ø477Ø ;       Register display routine
                    Ø478Ø ;
                    Ø479Ø WRREGS
1ED5 3E1C           Ø48ØØ         LD      A,1CH           ;Home the cursor
1ED7 CD42Ø6         Ø481Ø         CALL    @DSP
                    Ø482Ø         IF      @MOD4
1EDA 3EØF           Ø483Ø         LD      A,15            ;Turn off the cursor
1EDC CD42Ø6         Ø484Ø         CALL    @DSP
                    Ø485Ø         ENDIF
1EDF 3AA1ØØ         Ø486Ø         LD      A,(SAVTWO)      ;Ø = Normal display mode
1EE2 B7             Ø487Ø         OR      A               ;<> Ø = Full display mode
1EE3 2Ø68           Ø488Ø         JR      NZ,FULDSP       ;No reg display if FULL
1EE5 21A8ØØ         Ø489Ø         LD      HL,AFREG        ;Pt to register save area
1EE8 E5             Ø49ØØ         PUSH    HL
1EE9 21561F         Ø491Ø         LD      HL,REGTBL       ;Pt to reg symbol table
1EEC Ø6ØC           Ø492Ø         LD      B,12            ;Init for 12 registers
1EEE CD3522         Ø493Ø $?8     CALL    WR3BYT          ;Write 3-character symbol
1EF1 E3             Ø494Ø         EX      (SP),HL         ;Exchange reg save ptr
1EF2 5E             Ø495Ø         LD      E,(HL)          ;Place reg value -> DE
1EF3 23             Ø496Ø         INC     HL
1EF4 56             Ø497Ø         LD      D,(HL)
1EF5 23             Ø498Ø         INC     HL              ;Place next reg save
1EF6 E5             Ø499Ø         PUSH    HL              ;  pointer on the stack
1EF7 EB             Ø5ØØØ         EX      DE,HL           ;Reg value -> HL
1EF8 3E3D           Ø5Ø1Ø         LD      A,'='
1EFA CD42Ø6         Ø5Ø2Ø         CALL    @DSP
1EFD CD3122         Ø5Ø3Ø         CALL    WRSPA@
1FØØ 7C             Ø5Ø4Ø         LD      A,H             ;Write hi-order byte
1FØ1 CD2E22         Ø5Ø5Ø         CALL    WRHEX
1FØ4 7D             Ø5Ø6Ø         LD      A,L             ;Write lo-order byte
1FØ5 CD2E22         Ø5Ø7Ø         CALL    WRHEX
1FØ8 78             Ø5Ø8Ø         LD      A,B             ;Get loop counter &
1FØ9 E6ØB           Ø5Ø9Ø         AND     ØBH             ;  ck if 12 => AF pair
1FØB FEØ8           Ø51ØØ         CP      Ø8H             ;  or if 8 => AF' pair
1FØD 2Ø1C           Ø511Ø         JR      NZ,NOFLG        ;Bypass if not flag reg
1FØF 4D             Ø512Ø         LD      C,L             ;Transfer 'F' reg to C &
1F1Ø C5             Ø513Ø         PUSH    BC              ;  save the loop counter
1F11 217A1F         Ø514Ø         LD      HL,FLGTBL       ;Pt to flag synbol table
1F14 Ø6Ø8           Ø515Ø         LD      B,8             ;Init for 8 bits
1F16 CB21           Ø516Ø $?9     SLA     C               ;Shift a bit into carry
1F18 7E             Ø517Ø         LD      A,(HL)          ;P/u flag table character
1F19 38Ø2           Ø518Ø         JR      C,$?1Ø          ;Use table char if bit on
1F1B 3E2D           Ø519Ø         LD      A,'-'           ;  else use a dash
1F1D CD42Ø6         Ø52ØØ $?1Ø    CALL    @DSP
1F2Ø 23             Ø521Ø         INC     HL              ;Next flag table char
1F21 1ØF3           Ø522Ø         DJNZ    $?9             ;Loop for 8 flag bits
1F23 C1             Ø523Ø         POP     BC              ;Get main loop counter
1F24 3EFD           Ø524Ø         LD      A,61+ØCØH       ;Tab 6Ø to put cursor
1F26 CD42Ø6         Ø525Ø         CALL    @DSP            ;  on next line
1F29 18Ø3           Ø526Ø         JR      $?11
1F2B CD7321         Ø527Ø NOFLG   CALL    WRMEM
1F2E E1             Ø528Ø $?11    POP     HL              ;Get next reg save ptr
1F2F E3             Ø529Ø         EX      (SP),HL         ;Exc with next reg symbol
1F3Ø 1ØBC           Ø53ØØ         DJNZ    $?8             ;Loop end
1F32 E1             Ø531Ø         POP     HL              ;Get reg save ptr (fini)
1F33 2AA6ØØ         Ø532Ø         LD      HL,(DSPADR)     ;P/u memory disp address
1F36 Ø6Ø4           Ø533Ø         LD      B,4             ;Init for 4 lines
1F38 3EC6           Ø534Ø $?12    LD      A,6+ØCØH        ;Tab 6 spaces
1F3A CD42Ø6         Ø535Ø         CALL    @DSP
1F3D CD1522         Ø536Ø         CALL    WR2HEX@         ;Write the memory address
1F4Ø CD3122         Ø537Ø         CALL    WRSPA@
```

```
1F43 CD7321    Ø538Ø          CALL    WRMEM           ;Write a line of memory
1F46 1ØFØ      Ø539Ø          DJNZ    $?12            ;Loop until 4 or 16
1F48 3E1F      Ø54ØØ          LD      A,1FH           ;Clear to end-of-frame
1F4A C342Ø6    Ø541Ø          JP      @DSP
1F4D 2AA6ØØ    Ø542Ø FULDSP   LD      HL,(DSPADR)     ;P/u display address
1F5Ø 2EØØ      Ø543Ø          LD      L,Ø             ;Round to multiple of 256
1F52 Ø61Ø      Ø544Ø          LD      B,16            ;Init for 16 lines
1F54 18E2      Ø545Ø          JR      $?12
               Ø546Ø ;
               Ø547Ø ;        Register symbol table
               Ø548Ø ;
1F56 61        Ø549Ø REGTBL   DB      'af bc de hl af''bc''de''hl''ix iy sp pc '
     66 2Ø 62 63 2Ø 64 65 2Ø
     68 6C 2Ø 61 66 27 62 63
     27 64 65 27 68 6C 27 69
     78 2Ø 69 79 2Ø 73 7Ø 2Ø
     7Ø 63 2Ø
               Ø55ØØ ;
               Ø551Ø ;        Flag register bit symbol table
               Ø552Ø ;
1F7A 53        Ø553Ø FLGTBL   DB      'SZ1H1PNC'
     5A 31 48 31 5Ø 4E 43
               Ø554Ø ;
               Ø555Ø ;        Command G - Go to memory address NNNN,
               Ø556Ø ;         Optional breakpoints
               Ø557Ø ;
1F82 Ø6Ø2      Ø558Ø CMD_G    LD      B,2             ;Init for maximum of
1F84 11A5ØØ    Ø559Ø          LD      DE,NXTBYT       ;  two breakpoints
1F87 CDE421    Ø56ØØ          CALL    HEXIN@          ;Get exec address
1F8A 28Ø3      Ø561Ø          JR      Z,$?13          ;Go on end
1F8C 22BEØØ    Ø562Ø          LD      (PCREG),HL      ;  else save new start
1F8F 38ØA      Ø563Ø $?13     JR      C,$?14          ;Go if <ENTER> used
1F91 CDE421    Ø564Ø          CALL    HEXIN@          ;Get a breakpoint
1F94 F5        Ø565Ø          PUSH    AF
1F95 C4C51F    Ø566Ø          CALL    NZ,$?17         ;Set if brkpt entered
1F98 F1        Ø567Ø          POP     AF
1F99 1ØF4      Ø568Ø          DJNZ    $?13
               Ø569Ø $?14
1F9B AF        Ø57ØØ          XOR     A
1F9C 329F19    Ø571Ø          LD      (@DBGHK),A      ;Init DEBUG on
               Ø572Ø ;
               Ø573Ø ;        This next section of code picks up the register
               Ø574Ø ;        save area, pushes the save area onto the stack,
               Ø575Ø ;        then pops out into the correct reg assignments.
               Ø576Ø ;
1F9F 21BDØØ    Ø577Ø $?15     LD      HL,REGSAV       ;End of reg save area
1FA2 Ø6ØB      Ø578Ø          LD      B,11            ;Init for 11 regs
1FA4 56        Ø579Ø $?16     LD      D,(HL)
1FA5 2B        Ø58ØØ          DEC     HL
1FA6 5E        Ø581Ø          LD      E,(HL)
1FA7 2B        Ø582Ø          DEC     HL
1FA8 D5        Ø583Ø          PUSH    DE
1FA9 1ØF9      Ø584Ø          DJNZ    $?16
1FAB F1        Ø585Ø          POP     AF              ;Now pop the registers
1FAC C1        Ø586Ø          POP     BC
1FAD D1        Ø587Ø          POP     DE
1FAE E1        Ø588Ø          POP     HL
1FAF Ø8        Ø589Ø          EX      AF,AF'
1FBØ D9        Ø59ØØ          EXX
1FB1 F1        Ø591Ø          POP     AF
1FB2 C1        Ø592Ø          POP     BC
```

```
1FB3 D1       Ø593Ø          POP     DE
1FB4 E1       Ø594Ø          POP     HL
1FB5 Ø8       Ø595Ø          EX      AF,AF'
1FB6 D9       Ø596Ø          EXX
1FB7 DDE1     Ø597Ø          POP     IX
1FB9 FDE1     Ø598Ø          POP     IY
1FBB E1       Ø599Ø          POP     HL
1FBC F9       Ø6ØØØ          LD      SP,HL
1FBD 2ABEØØ   Ø6Ø1Ø          LD      HL,(PCREG)      ;Init the branch address
1FCØ E5       Ø6Ø2Ø          PUSH    HL
1FC1 2AAEØØ   Ø6Ø3Ø          LD      HL,(HLREG)
1FC4 C9       Ø6Ø4Ø          RET                     ;Go to branch
              Ø6Ø5Ø ;
              Ø6Ø6Ø ;        This next routine will insert an RST 48 inst into
              Ø6Ø7Ø ;        the target of a single-step or breakpoint
              Ø6Ø8Ø ;        providing the target address is a RAM location.
              Ø6Ø9Ø ;        If it is, the target byte and its address are
              Ø61ØØ ;        saved in one of the instruction save areas.
              Ø611Ø ;        If the target address is ROM or nonexistent, a
              Ø612Ø ;        branch to command INPUT routine is taken instead
              Ø613Ø ;        of the pending operation.
              Ø614Ø ;
1FC5 7E       Ø615Ø $?17     LD      A,(HL)          ;Save byte of next inst
1FC6 12       Ø616Ø          LD      (DE),A
1FC7 1B       Ø617Ø          DEC     DE
1FC8 3EF7     Ø618Ø          LD      A,ØF7H          ;Insert RST 48 into
1FCA 77       Ø619Ø          LD      (HL),A          ;  next INST address
1FCB BE       Ø62ØØ          CP      (HL)            ;Ck if RAM/ROM/no memory
1FCC C2321E   Ø621Ø          JP      NZ,$?1          ;Go to command if not RAM
1FCF 7C       Ø622Ø          LD      A,H             ;Is RAM, save address of
1FDØ 12       Ø623Ø          LD      (DE),A          ;  insertion into buffer
1FD1 1B       Ø624Ø          DEC     DE              ;  pointed to by DE, DE-1
1FD2 7D       Ø625Ø          LD      A,L
1FD3 12       Ø626Ø          LD      (DE),A
1FD4 1B       Ø627Ø          DEC     DE
1FD5 C9       Ø628Ø          RET
              Ø629Ø ;
              Ø63ØØ ;        Commands A & H - Modify address NNNN to XX
              Ø631Ø ;         <SPACE> increments address
              Ø632Ø ;
1FD6 32AØØØ   Ø633Ø CMD_AH   LD      (SAVONE),A      ;Save entry condition
1FD9 2AA3ØØ   Ø634Ø          LD      HL,(NXTADR)     ;Default to current mod addr
1FDC CDE421   Ø635Ø          CALL    HEXIN@
1FDF 22A3ØØ   Ø636Ø $?18     LD      (NXTADR),HL     ;Adjust addr for mod
1FE2 D8       Ø637Ø          RET     C               ;Return on <ENTER>
1FE3 E5       Ø638Ø          PUSH    HL
1FE4 CDD51E   Ø639Ø          CALL    WRREGS
1FE7 21ØØØD   Ø64ØØ          LD      HL,13<8!Ø       ;Cursor to 13,Ø
1FEA Ø6Ø3     Ø641Ø          LD      B,3
1FEC 3EØF     Ø642Ø          LD      A,15            ;SVC @VDCTL set cursor
1FEE EF       Ø643Ø          RST     28H
1FEF 2AA3ØØ   Ø644Ø          LD      HL,(NXTADR)     ;P/u mod address again
1FF2 CD1522   Ø645Ø          CALL    WR2HEX@         ;Write the address & save
1FF5 E5       Ø646Ø          PUSH    HL              ;  the mod addr again
1FF6 21ØØØE   Ø647Ø          LD      HL,14<8!Ø       ;Cursor to 14,Ø
1FF9 Ø6Ø3     Ø648Ø          LD      B,3
1FFB 3EØF     Ø649Ø          LD      A,15            ;SVC @VDCTL set cursor
1FFD EF       Ø65ØØ          RST     28H
1FFE E1       Ø651Ø          POP     HL              ;Recover mod addr
1FFF CD132Ø   Ø652Ø          CALL    AHDSP
2ØØ2 3E2D     Ø653Ø          LD      A,'-'
```

```
2004 CD4206   06540          CALL    @DSP
2007 D1       06550          POP     DE              ;Recover mod addr in DE
2008 CD2920   06560          CALL    AHGET
200B EB       06570          EX      DE,HL           ;Switch mod addr/value
200C 2801     06580          JR      Z,$?19          ;Bypass change on <SPACE>
200E 73       06590          LD      (HL),E          ;Insert new val in memory
200F D8       06600 $?19     RET     C               ;To CMND on non-digit
2010 23       06610          INC     HL              ;  else increment address
2011 18CC     06620          JR      $?18            ;  pointer & loop
2013 3AA000   06630 AHDSP    LD      A,(SAVONE)
2016 FE61     06640          CP      'a'
2018 C21122   06650          JP      NZ,WR1HEX@      ;Write (HL) & bump H
201B 7E       06660 DSPASC@  LD      A,(HL)          ;Else write in ASCII
201C FE20     06670          CP      20H             ;Convert non-displayable
201E 3804     06680          JR      C,TYP3          ;  values to '.'
2020 FEC0     06690          CP      0C0H
2022 3802     06700          JR      C,TYP4
2024 3E2E     06710 TYP3     LD      A,'.'
2026 C34206   06720 TYP4     JP      @DSP
2029 3AA000   06730 AHGET    LD      A,(SAVONE)
202C FE61     06740          CP      'a'
202E C2E421   06750          JP      NZ,HEXIN@
2031 E5       06760 GETASC@  PUSH    HL              ;Provide lower/upper
2032 21D621   06770          LD      HL,INPUC@+1     ;  case entry in type
2035 366F     06780          LD      (HL),6FH        ;  by modifying sys5 code
2037 CDC921   06790          CALL    INPUT@
203A 36EF     06800          LD      (HL),0EFH       ;Restore the UC -> lc
203C E1       06810          POP     HL              ;  conversion
203D 6F       06820          LD      L,A
203E C9       06830          RET
              06840 ;
              06850 ;        Command R - Load register pair RP with NNNN
              06860 ;
203F CDC921   06870 CMD_R    CALL    INPUT@          ;Get 1st symbol char
2042 C8       06880          RET     Z               ;Return if end
2043 4F       06890          LD      C,A             ;  else save char in C
2044 CDC921   06900          CALL    INPUT@          ;Get 2nd symbol char
2047 C8       06910          RET     Z               ;Return if end
2048 57       06920          LD      D,A             ;  else save char in D
2049 1E20     06930          LD      E,' '           ;Init for space
204B CDC921   06940          CALL    INPUT@          ;Get 3rd symbol char
204E D8       06950          RET     C               ;Return on end
204F 2806     06960          JR      Z,$?20          ;Bypass if not primed
2051 5F       06970          LD      E,A             ;  else put "'" into E
2052 CDC921   06980          CALL    INPUT@          ;Ck for space separator
2055 C0       06990          RET     NZ              ;Return if none
2056 D8       07000          RET     C
2057 21561F   07010 $?20     LD      HL,REGTBL       ;Register symbol table
205A 060C     07020          LD      B,12            ;Init for 12 registers
205C 7E       07030 $?21     LD      A,(HL)          ;Match first symbol?
205D B9       07040          CP      C
205E 2806     07050          JR      Z,$?24          ;If a match, test 2nd
2060 23       07060          INC     HL              ;  else pt to next reg
2061 23       07070 $?22     INC     HL
2062 23       07080 $?23     INC     HL
2063 10F7     07090          DJNZ    $?21            ;Loop for 12 regs
2065 C9       07100          RET                     ;Return if no match
2066 23       07110 $?24     INC     HL              ;Pt to 2nd table char
2067 7E       07120          LD      A,(HL)          ;  & p/u the symbol
2068 BA       07130          CP      D               ;Ck the 2nd char input
2069 20F6     07140          JR      NZ,$?22         ;-> next if no match
```

```
2Ø6B 23        Ø715Ø          INC     HL              ;Match, ck 3rd reg symbol
2Ø6C 7E        Ø716Ø          LD      A,(HL)          ;P/u the 3rd table symbol
2Ø6D BB        Ø717Ø          CP      E               ;  & compare with input
2Ø6E 2ØF2      Ø718Ø          JR      NZ,$?23         ;-> next if no match
2Ø7Ø 3E18      Ø719Ø          LD      A,18H           ;Convert counter to index
2Ø72 9Ø        Ø72ØØ          SUB     B               ;  into reg save area
2Ø73 9Ø        Ø721Ø          SUB     B
2Ø74 4F        Ø722Ø          LD      C,A             ;Index into BC
2Ø75 Ø6ØØ      Ø723Ø          LD      B,Ø
2Ø77 21A8ØØ    Ø724Ø          LD      HL,AFREG        ;Start of reg save area
2Ø7A Ø9        Ø725Ø          ADD     HL,BC           ;Add index to get pointer
2Ø7B E5        Ø726Ø          PUSH    HL              ;Save the pointer
2Ø7C 3E1E      Ø727Ø          LD      A,1EH           ;Erase to end-of-line
2Ø7E CD42Ø6    Ø728Ø          CALL    @DSP
2Ø81 D1        Ø729Ø          POP     DE              ;Recover pointer
2Ø82 CDE421    Ø73ØØ          CALL    HEXIN@          ;Read in the new value
2Ø85 C8        Ø731Ø          RET     Z               ;No update if none
2Ø86 EB        Ø732Ø          EX      DE,HL           ;Exchg value/pointer
2Ø87 73        Ø733Ø          LD      (HL),E          ;Insert new value into
2Ø88 23        Ø734Ø          INC     HL              ;  register save area
2Ø89 72        Ø735Ø          LD      (HL),D
2Ø8A C9        Ø736Ø          RET
               Ø737Ø ;
               Ø738Ø ;        Command I - Step one instruction at a time
               Ø739Ø ;
2Ø8B F5        Ø74ØØ CMD_CI   PUSH    AF              ;Save whether I or C
2Ø8C ED5BBEØØ  Ø741Ø          LD      DE,(PCREG)      ;Point to inst address
2Ø9Ø 1A        Ø742Ø          LD      A,(DE)          ;  & get it
2Ø91 215721    Ø743Ø          LD      HL,XY_TAB       ;IX,IY Table
2Ø94 FEDD      Ø744Ø          CP      ØDDH            ;Is inst an IX?
2Ø96 28ØE      Ø745Ø          JR      Z,$?25
2Ø98 FEFD      Ø746Ø          CP      ØFDH            ;Is inst an IY?
2Ø9A 28ØA      Ø747Ø          JR      Z,$?25
2Ø9C 211F21    Ø748Ø          LD      HL,OP_TAB       ;All X IX, IY, & ED
2Ø9F FEED      Ø749Ø          CP      ØEDH            ;Is inst an ED?
2ØA1 2ØØ6      Ø75ØØ          JR      NZ,$?26
2ØA3 215Ø21    Ø751Ø          LD      HL,ED_TAB       ;ED Table
2ØA6 13        Ø752Ø $?25     INC     DE              ;Get next byte for
2ØA7 1A        Ø753Ø          LD      A,(DE)          ;  IX, IY, and ED inst
2ØA8 1B        Ø754Ø          DEC     DE              ;Reset ptr to 1st byte
2ØA9 4F        Ø755Ø $?26     LD      C,A             ;Inst byte to reg C
               Ø756Ø ;
               Ø757Ø ;        This next section of code determines the length
               Ø758Ø ;        of all instructions and whether they
               Ø759Ø ;        are CALLs, JumPs, or RETurns.
               Ø76ØØ ;
2ØAA 7E        Ø761Ø $?27     LD      A,(HL)          ;P/u table value &
2ØAB A1        Ø762Ø          AND     C               ;  strip off certain bits
2ØAC 23        Ø763Ø          INC     HL              ;Pt to table code
2ØAD BE        Ø764Ø          CP      (HL)            ;If a match, the inst is
2ØAE 23        Ø765Ø          INC     HL              ;  fully decoded as to
2ØAF 28Ø6      Ø766Ø          JR      Z,$?28          ;  length & type by the
2ØB1 23        Ø767Ø          INC     HL              ;  next byte
2ØB2 7E        Ø768Ø          LD      A,(HL)          ;Ck for table end
2ØB3 FE Ø5     Ø769Ø          CP      5
2ØB5 3ØF3      Ø77ØØ          JR      NC,$?27
2ØB7 7E        Ø771Ø $?28     LD      A,(HL)          ;Get control/length byte
2ØB8 47        Ø772Ø          LD      B,A             ;  into reg B
2ØB9 E6ØF      Ø773Ø          AND     ØFH             ;Strip off the control
2ØBB 6F        Ø774Ø          LD      L,A             ;Put length into reg L
2ØBC 26ØØ      Ø775Ø          LD      H,Ø             ;Zero out reg H
```

```
20BE 19       07760          ADD     HL,DE           ;Next address into HL
20BF D5       07770          PUSH    DE              ;This addr in DE saved
20C0 11A500   07780          LD      DE,NXTBYT       ;Buffer area
20C3 CDC51F   07790          CALL    $?17            ;Insert RST 48 if RAM
20C6 E1       07800          POP     HL              ;Get this inst address
20C7 78       07810          LD      A,B             ;Get control/length byte
20C8 E6F0     07820          AND     0F0H            ;Strip off length
20CA 282A     07830          JR      Z,$?29          ;Go if regular inst
20CC 23       07840          INC     HL
20CD FE20     07850          CP      20H
20CF 3846     07860          JR      C,$?34          ;Branch if 'JP (HL)'
20D1 2838     07870          JR      Z,$?33          ;Go if 'JP (IX/IY)'
20D3 FE40     07880          CP      40H
20D5 382B     07890          JR      C,$?32          ;Go if 'JR' or 'DJNZ'
20D7 2823     07900          JR      Z,$?31          ;Branch if 'JP' inst
20D9 FE60     07910          CP      60H
20DB 381C     07920          JR      C,$?30          ;Branch if 'RET' inst
20DD 2812     07930          JR      Z,$?28A         ;Branch if CALL inst
20DF 79       07940          LD      A,C             ;  else calc target of
20E0 E638     07950          AND     38H             ;  the RST inst
20E2 6F       07960          LD      L,A
20E3 2600     07970          LD      H,0
20E5 F1       07980          POP     AF              ;Rcvr entry command
20E6 FE63     07990          CP      'c'
20E8 280C     08000          JR      Z,$?29          ;Go in "call" mode
20EA 7D       08010          LD      A,L             ;Must check RST for
20EB FE28     08020          CP      5<3             ;  40, 48, 56 inhibit
20ED 3007     08030          JR      NC,$?29         ;Convert to CALL
20EF 1829     08040          JR      $?35            ;  else single step
20F1 F1       08050 $?28A    POP     AF              ;Recover entry command
20F2 FE69     08060          CP      'i'             ;Was command an 'I'
20F4 2806     08070          JR      Z,$?31          ;Go for 'CALLs' if 'I'
20F6 C39F1F   08080 $?29     JP      $?15            ;Go for 'CALLs' if 'C'
20F9 2ABC00   08090 $?30     LD      HL,(SPREG)      ;RET inst, p/u RET addr
20FC 7E       08100 $?31     LD      A,(HL)          ;JP inst, p/u jump addr &
20FD 23       08110          INC     HL              ;  insert into reg HL
20FE 66       08120          LD      H,(HL)
20FF 6F       08130          LD      L,A
2100 1818     08140          JR      $?35
2102 4E       08150 $?32     LD      C,(HL)          ;'JR' or 'DJNZ', get 'E'
2103 79       08160          LD      A,C             ;Make A=0 if C is
2104 07       08170          RLCA                    ;  positive, else make
2105 9F       08180          SBC     A,A             ;  A=FF for negative
2106 47       08190          LD      B,A             ;Put -> B, FF if 'E' neg
2107 23       08200          INC     HL              ;  or 0 if 'E' pos.
2108 09       08210          ADD     HL,BC           ;Add the displacement
2109 180F     08220          JR      $?35
210B 2AB800   08230 $?33     LD      HL,(IXREG)      ;Init for JP (IX)
210E CB69     08240          BIT     5,C             ;Test inst for DD/FD
2110 2808     08250          JR      Z,$?35          ;Bit 5 off = DD
2112 2ABA00   08260          LD      HL,(IYREG)      ;JP (IY), p/u jump addr
2115 1803     08270          JR      $?35
2117 2AAE00   08280 $?34     LD      HL,(HLREG)      ;JP (HL), p/u jump addr
211A CDC51F   08290 $?35     CALL    $?17
211D 18D7     08300          JR      $?29
              08310 ;
              08320 ;        The next three tables are used to determine the
              08330 ;        length & instruction type for all instructions
              08340 ;        used in the single-step mode. Table format uses
              08350 ;        three bytes for each decoding process. The 1st
              08360 ;        byte is ANDed with the inst byte to strip off
```

```
                    ;       selected bits and include others. The result is
                    ;       compared to the next table byte (test byte) for
                    ;       a match. If matched, then the inst byte has been
                    ;       identified as to its class & length. The 3rd byte
                    ;       denotes the class and length as follows:
                    ; High order nybble
                    ;         Ø = Regular instruction
                    ;         1 = JP (HL) instruction
                    ;         2 = JP (IX) or JP (IY) instruction
                    ;         3 = JR or DJNZ instructions
                    ;         4 = JP instructions
                    ;         5 = RET instructions
                    ;         6 = CALL instructions
                    ;         7 = RST instructions
                    ; Low order nybble = the length
                    ;       The last byte of each table is the length of
                    ;       all other instructions.
                    ;
                    ;       Table for regular instructions (no IX, IY, ED)
                    ;
211F C7             OP_TAB  DB      ØC7H,ØCØH,51H   ;C8, D8, E8, F8
     CØ 51
2122 FF             Ø858Ø           DB      ØFFH,ØC9H,51H   ;C9
     C9 51
2125 FF             Ø859Ø           DB      ØFFH,ØE9H,11H   ;E9
     E9 11
2128 CF             Ø86ØØ           DB      ØCFH,Ø1H,Ø3H    ;Ø1, 11, 21, 31
     Ø1 Ø3
212B E7             Ø861Ø           DB      ØE7H,22H,3      ;22, 2A, 32, 3A
     22 Ø3
212E C7C2           Ø862Ø           DW      ØC2C7H          ;C2, CA, D2, DA, E2, EA,
213Ø 43             Ø863Ø           DB      43H             ;  F2, FA
2131 FF             Ø864Ø           DB      ØFFH,ØC3H,43H   ;C3
     C3 43
2134 C7C4           Ø865Ø           DW      ØC4C7H          ;C4, CC, D4, DC, E4, EC,
2136 63             Ø866Ø           DB      63H             ;  F4, FC
2137 FF             Ø867Ø           DB      ØFFH,ØCDH,63H   ;CD
     CD 63
213A C7Ø6           Ø868Ø           DW      Ø6C7H           ;Ø6, ØE, 16, 1E, 26, 2E,
213C Ø2             Ø869Ø           DB      Ø2H             ;  36, 3E
213D F7             Ø87ØØ           DB      ØF7H,ØD3H,Ø2    ;D3, DB
     D3 Ø2
214Ø C7C6           Ø871Ø           DW      ØC6C7H          ;C6, CE, D6, DE, E6, EE,
2142 Ø2             Ø872Ø           DB      Ø2H             ;  F6, FE
2143 FF             Ø873Ø           DB      ØFFH,ØCBH,2     ;All CB instructions
     CB Ø2
2146 F7             Ø874Ø           DB      ØF7H,1ØH,32H    ;1Ø, 18
     1Ø 32
2149 E7             Ø875Ø           DB      ØE7H,2ØH,32H    ;2Ø, 28, 3Ø, 38
     2Ø 32
214C C7             Ø876Ø           DB      ØC7H,ØC7H,71H   ;RST instructions
     C7 71
214F Ø1             Ø877Ø           DB      1               ;All others are 1-byte
                    Ø878Ø ;
                    Ø879Ø ;       Next table is for ED - extended instructions
                    Ø88ØØ ;
215Ø C7             Ø881Ø ED_TAB    DB      ØC7H,43H,Ø4H    ;43, 4B, 53, 5B, 73, 7B
     43 Ø4
2153 F7             Ø882Ø           DB      ØF7H,45H,52H    ;45, 4D
     45 52
2156 Ø2             Ø883Ø           DB      2               ;All other ED are 2-byte
```

```
                    Ø884Ø ;
                    Ø885Ø ;       IX, IY Index instructions table
                    Ø886Ø ;
2157 FE             Ø887Ø XY_TAB  DB      ØFEH,34H,Ø3     ;34, 35
     34 Ø3
215A CØ             Ø888Ø         DB      ØCØH,4ØH,Ø3     ;4X, 5X, 6X, 7X (X = Ø-F)
     4Ø Ø3
215D CØ             Ø889Ø         DB      ØCØH,8ØH,Ø3     ;8X, 9X, AX, BX (X= Ø-F)
     8Ø Ø3
216Ø FF             Ø89ØØ         DB      ØFFH,21H,Ø4     ;21
     21 Ø4
2163 FF             Ø891Ø         DB      ØFFH,22H,Ø4     ;22
     22 Ø4
2166 FF             Ø892Ø         DB      ØFFH,2AH,Ø4     ;2A
     2A Ø4
2169 FF             Ø893Ø         DB      ØFFH,36H,Ø4     ;36
     36 Ø4
216C FF             Ø894Ø         DB      ØFFH,ØCBH,Ø4    ;CB
     CB Ø4
216F FF             Ø895Ø         DB      ØFFH,ØE9H,22H   ;E9
     E9 22
2172 Ø2             Ø896Ø         DB      Ø2H             ;All others are 2-bytes
                    Ø897Ø ;
                    Ø898Ø ;       Routine to display memory on CRT screen
                    Ø899Ø ;
2173 C5             Ø9ØØØ WRMEM   PUSH    BC              ;Save main counter 4/16
2174 3E3D           Ø9Ø1Ø         LD      A,'='
2176 CD42Ø6         Ø9Ø2Ø         CALL    @DSP
2179 3C             Ø9Ø3Ø         INC     A               ;'>'
217A CD42Ø6         Ø9Ø4Ø         CALL    @DSP
217D Ø61Ø           Ø9Ø5Ø         LD      B,16            ;Init for 16 lines
217F E5             Ø9Ø6Ø         PUSH    HL              ;Save memory pointer
218Ø CDA421         Ø9Ø7Ø $?36    CALL    GRPHIC          ;Ck if need graphic bars
2183 CD1122         Ø9Ø8Ø         CALL    WR1HEX@         ;Call on HEX display only
2186 1ØF8           Ø9Ø9Ø         DJNZ    $?36            ;Loop until full line
2188 E1             Ø91ØØ         POP     HL              ;Rcvr memory pointer
                    Ø911Ø ;
                    Ø912Ø ;       Now write the line in ASCII
                    Ø913Ø ;
2189 CD3122         Ø914Ø         CALL    WRSPA@
218C Ø61Ø           Ø915Ø         LD      B,16
218E CDC321         Ø916Ø $?37    CALL    $?41            ;Space after 8th
2191 7E             Ø917Ø         LD      A,(HL)          ;P/u the byte -> reg A
2192 FE2Ø           Ø918Ø         CP      2ØH             ;Repl controls with '.'
2194 3804           Ø919Ø         JR      C,$?38
2196 FECØ           Ø92ØØ         CP      ØCØH            ;Tabs/specials with '.'
2198 3802           Ø921Ø         JR      C,$?39
219A 3E2E           Ø922Ø $?38    LD      A,'.'
219C CD42Ø6         Ø923Ø $?39    CALL    @DSP
219F 23             Ø924Ø         INC     HL              ;Bump memory address
21AØ 1ØEC           Ø925Ø         DJNZ    $?37
21A2 C1             Ø926Ø         POP     BC              ;Get line counter
21A3 C9             Ø927Ø         RET
                    Ø928Ø ;
                    Ø929Ø ;       This routine determines if the vertical graphic
                    Ø93ØØ ;       bars should be surrounding the current character
                    Ø931Ø ;
21A4 ED5BA3ØØ       Ø932Ø GRPHIC  LD      DE,(NXTADR)     ;P/u modification address
21A8 13             Ø933Ø         INC     DE              ;  & increment it
21A9 E5             Ø934Ø         PUSH    HL              ;Save current memory
21AA AF             Ø935Ø         XOR     A               ;  display address
```

```
21AB ED52      Ø936Ø          SBC     HL,DE           ;Ck if mod addr=disp addr
               Ø937Ø          IF      @MOD4
21AD 3E95      Ø938Ø          LD      A,95H           ;Graphic left bar
               Ø939Ø          ENDIF
               Ø94ØØ          IF      @MOD2
               Ø941Ø          LD      A,15H
               Ø942Ø          ENDIF
21AF 28ØE      Ø943Ø          JR      Z,$?4Ø          ;Insert graphic if equal
21B1 CDC321    Ø944Ø          CALL    $?41            ;Not =, insert space if
21B4 23        Ø945Ø          INC     HL              ;  between pos 8 & 9
21B5 7D        Ø946Ø          LD      A,L             ;Result is zero if next
21B6 B4        Ø947Ø          OR      H               ;  char address is also
               Ø948Ø                                  ;  the display address
21B7 E1        Ø949Ø          POP     HL              ;Get current mem disp adr
               Ø95ØØ          IF      @MOD4
21B8 3EAA      Ø951Ø          LD      A,ØAAH          ;Graphic right bar output
21BA CA42Ø6    Ø952Ø          JP      Z,@DSP          ;Go if yes
21BD 18Ø8      Ø953Ø          JR      $?42            ;  else continue
               Ø954Ø          ENDIF
               Ø955Ø          IF      @MOD2
               Ø956Ø          JR      NZ,$?42         ;Go if not
               Ø957Ø          XOR     A               ;  lead in
               Ø958Ø          CALL    @DSP            ;Init video lead in
               Ø959Ø          LD      A,15H
               Ø96ØØ          JP      @DSP            ;  and display
               Ø961Ø          ENDIF
21BF           Ø962Ø $?4Ø     EQU     $
               Ø963Ø          IF      @MOD2
               Ø964Ø          PUSH    AF
               Ø965Ø          XOR     A
               Ø966Ø          CALL    @DSP            ;Lead in code
               Ø967Ø          POP     AF              ;Restore
               Ø968Ø          ENDIF
21BF CD42Ø6    Ø969Ø          CALL    @DSP            ;Display char
21C2 E1        Ø97ØØ          POP     HL              ;Recover current display
21C3 78        Ø971Ø $?41     LD      A,B             ;  address & output a
21C4 FEØ8      Ø972Ø          CP      8               ;  space if between the
21C6 CØ        Ø973Ø          RET     NZ              ;  8th & 9th bytes
21C7 1868      Ø974Ø $?42     JR      WRSPA@          ;  else just return
               Ø975Ø ;
               Ø976Ø ;        This routine will return with zero flag set
               Ø977Ø ;        on entry of a comma or a SPACE. Entry of <ENTER>
               Ø978Ø ;        will set carry flag and return
               Ø979Ø ;
21C9 D5        Ø98ØØ INPUT@   PUSH    DE
21CA CD28Ø6    Ø981Ø $?43     CALL    @KEY
21CD FEØD      Ø982Ø          CP      ØDH             ;ENTER?
21CF 281Ø      Ø983Ø          JR      Z,$?44
21D1 FE2Ø      Ø984Ø          CP      2ØH             ;Get another char if
21D3 38F5      Ø985Ø          JR      C,$?43          ;  entry was control
21D5 CBEF      Ø986Ø INPUC@   SET     5,A             ;Cvrt UC to lc
21D7 CD42Ø6    Ø987Ø          CALL    @DSP            ;Not control, disp it
21DA D1        Ø988Ø          POP     DE
21DB FE2C      Ø989Ø          CP      ','             ;Return with zero flag
21DD C8        Ø99ØØ          RET     Z               ;  set if a comma
21DE FE2Ø      Ø991Ø          CP      ' '             ;Return with zero flag
21EØ C9        Ø992Ø          RET                     ;  set if <SPACE>
21E1 D1        Ø993Ø $?44     POP     DE
21E2 37        Ø994Ø          SCF                     ;<ENTER> will set
21E3 C9        Ø995Ø          RET                     ;  the carry flag
               Ø996Ø ;
```

```
                    Ø997Ø ;       This routine will read in digits
                    Ø998Ø ;       and convert them to binary
                    Ø999Ø ;
21E4 CDC921         1ØØØØ HEXIN@  CALL    INPUT@          ;Get char and return on
21E7 C8             1ØØ1Ø         RET     Z               ;  SPACE, COMMA, or ENTER
21E8 21ØØØØ         1ØØ2Ø         LD      HL,Ø            ;Init value to zero
21EB CDØØ22         1ØØ3Ø $?45    CALL    CVB             ;Convert to binary if ok
21EE DA4C1E         1ØØ4Ø         JP      C,CMND          ;  else back on bad digit
21F1 29             1ØØ5Ø         ADD     HL,HL           ;Multiply current value
21F2 29             1ØØ6Ø         ADD     HL,HL           ;  by 16 and insert the
21F3 29             1ØØ7Ø         ADD     HL,HL           ;  new digit into the
21F4 29             1ØØ8Ø         ADD     HL,HL           ;  lo-order nybble of L
21F5 B5             1ØØ9Ø         OR      L
21F6 6F             1Ø1ØØ         LD      L,A
21F7 CDC921         1Ø11Ø         CALL    INPUT@          ;Get another character
21FA 2ØEF           1Ø12Ø         JR      NZ,$?45         ;Go if not separator
21FC 1F             1Ø13Ø         RRA                     ;Force <ENTER> to set
21FD CE81           1Ø14Ø         ADC     A,81H           ;  the carry flag
21FF C9             1Ø15Ø         RET
                    1Ø16Ø ;
                    1Ø17Ø ;       Routine to convert expected ASCII hex digit to
                    1Ø18Ø ;       its binary value. Set Carry-flag on bad digit
                    1Ø19Ø ;
2200 D630           1Ø2ØØ CVB     SUB     '0'             ;Convert digit to binary
2202 D8             1Ø21Ø         RET     C               ;Error if < '0'
2203 C6C9           1Ø22Ø         ADD     A,ØC9H          ;Ck for > F (46H-30H=16H)
                    1Ø23Ø                                 ;  (16H + E9H = FFH)
2205 D8             1Ø24Ø         RET     C               ;Error if > ASCII 'F'
2206 C606           1Ø25Ø         ADD     A,6             ;(E9H-EFH) to (EFH-05H)
2208 3803           1Ø26Ø         JR      C,ATOF          ;Carry denotes was <A-F>
220A C627           1Ø27Ø         ADD     A,27H           ;(EFH-FFH) to (F6H-06H)
220C D8             1Ø28Ø         RET     C               ;Error if (3AH-3FH/:-?)
220D C60A           1Ø29Ø ATOF    ADD     A,ØAH           ;(00D-06D) to (10D-16D)
                    1Ø3ØØ                                 ;  or (F6H-FFH) to (Ø-9)
220F B7             1Ø31Ø         OR      A               ;Set zero flag on zero
2210 C9             1Ø32Ø         RET
                    1Ø33Ø ;
                    1Ø34Ø ;       Routine to write one byte as two hex digits
                    1Ø35Ø ;
2211 7E             1Ø36Ø WR1HEX@ LD      A,(HL)
2212 23             1Ø37Ø         INC     HL
2213 18Ø5           1Ø38Ø         JR      CV2HEX@
                    1Ø39Ø ;
                    1Ø4ØØ ;       Routine to write 2 bytes (HL) as 4 hex digits
                    1Ø41Ø ;
2215 7C             1Ø42Ø WR2HEX@ LD      A,H
2216 CD1A22         1Ø43Ø         CALL    CV2HEX@
2219 7D             1Ø44Ø         LD      A,L
                    1Ø45Ø ;
                    1Ø46Ø ;       Routine converts a byte to 2 hex digits
                    1Ø47Ø ;
221A F5             1Ø48Ø CV2HEX@ PUSH    AF              ;Save the byte in A
221B 1F             1Ø49Ø         RRA                     ;Move hi-order
221C 1F             1Ø5ØØ         RRA                     ;  into lo-order
221D 1F             1Ø51Ø         RRA
221E 1F             1Ø52Ø         RRA
221F CD2322         1Ø53Ø         CALL    $?46            ;Strip off hi-order
                    1Ø54Ø                                 ;  & convert to ASCII
2222 F1             1Ø55Ø         POP     AF              ;Recover the byte
2223 E60F           1Ø56Ø $?46    AND     0FH             ;Strip off hi-order
                    1Ø57Ø                                 ;  & convert to ASCII
```

```
2225 C690      10580          ADD     A,90H
2227 27        10590          DAA
2228 CE40      10600          ADC     A,40H
222A 27        10610          DAA
222B C34206    10620 $?47     JP      @DSP
               10630 ;
               10640 ;        Miscellaneous routines
               10650 ;
222E CD1A22    10660 WRHEX    CALL    CV2HEX@
2231 3E20      10670 WRSPA@   LD      A,20H
2233 18F6      10680          JR      $?47
               10690 ;
2235 CD3B22    10700 WR3BYT   CALL    $?48
2238 CD3B22    10710          CALL    $?48
223B 7E        10720 $?48     LD      A,(HL)
223C 23        10730          INC     HL
223D 18EC      10740          JR      $?47
               10750 ;
               10760 ;        Command B - Block move
               10770 ;
223F FE62      10780 BLOCK    CP      'b'
2241 2048      10790          JR      NZ,FILL
2243 2AA600    10800          LD      HL,(DSPADR)     ;'b'lock move s,d,len
2246 CDE421    10810          CALL    HEXIN@          ;Default to display addd
2249 D8        10820          RET     C               ;Back on <ENTER>
224A 22A600    10830          LD      (DSPADR),HL     ;Save start addr
224D 2008      10840          JR      NZ,BLO1         ;Go if start entered
224F CD1522    10850          CALL    WR2HEX@         ;  else show default
2252 3E2C      10860          LD      A,','
2254 CD4206    10870          CALL    @DSP
2257 2AA300    10880 BLO1     LD      HL,(NXTADR)     ;Default next address
225A CDE421    10890          CALL    HEXIN@
225D 22A300    10900          LD      (NXTADR),HL     ;Save dest address
2260 200A      10910          JR      NZ,BLO2         ;Go if entered
2262 F5        10920          PUSH    AF
2263 CD1522    10930          CALL    WR2HEX@         ;  else show default
2266 3E2C      10940          LD      A,','
2268 CD4206    10950          CALL    @DSP
226B F1        10960          POP     AF
226C 210001    10970 BLO2     LD      HL,256          ;Default length to 256
226F 3805      10980          JR      C,BLO3          ;Go if <ENTER> used prev.
2271 CDE421    10990          CALL    HEXIN@          ;Get new length
2274 2005      11000          JR      NZ,BLO4         ;Go if entered
2276 E5        11010 BLO3     PUSH    HL
2277 CD1522    11020          CALL    WR2HEX@         ;  else dsply default
227A E1        11030          POP     HL
227B 44        11040 BLO4     LD      B,H             ;Length to BC
227C 4D        11050          LD      C,L
227D 2AA600    11060          LD      HL,(DSPADR)     ;Set source
2280 ED5BA300  11070          LD      DE,(NXTADR)     ;  and dest
2284 EDB0      11080          LDIR
2286 ED53A300  11090          LD      (NXTADR),DE     ;Set new mod addr
228A C9        11100          RET
               11110 ;
               11120 ;        'f'ill aaaa,bbbb,cc
               11130 ;
228B FE66      11140 FILL     CP      'f'
228D 201C      11150          JR      NZ,JUMP
228F CDE421    11160          CALL    HEXIN@          ;Get starting address
2292 C8        11170          RET     Z
2293 E5        11180          PUSH    HL              ;Save starting address
```

```
2294 CDE421   11190          CALL    HEXIN@          ;Get ending address
2297 E3       11200          EX      (SP),HL         ;Place ending into BC
2298 C1       11210          POP     BC              ;  & starting into HL
2299 C8       11220          RET     Z
229A E5       11230          PUSH    HL              ;Save starting again
229B CDE421   11240          CALL    HEXIN@          ;Get fill character
229E 5D       11250          LD      E,L             ;Save fill in E
229F E1       11260          POP     HL              ;Recover starting addr
22AØ C8       11270          RET     Z
22A1 AF       11280          XOR     A               ;Clear the C-flag
22A2 E5       11290 FIL1     PUSH    HL
22A3 ED42     11300          SBC     HL,BC
22A5 E1       11310          POP     HL
22A6 DØ       11320          RET     NC              ;Return when start = end
22A7 73       11330          LD      (HL),E          ;Stuff char into memory
22A8 23       11340          INC     HL
22A9 18F7     11350          JR      FIL1
              11360 ;
              11370 ;        'j'ump over next instruction
              11380 ;
22AB FE6A     11390 JUMP     CP      'j'
22AD 2ØØ8     11400          JR      NZ,QUERY
22AF 2ABEØØ   11410          LD      HL,(PCREG)      ;Get current PC location
22B2 23       11420          INC     HL              ;  and increment it
22B3 22BEØØ   11430          LD      (PCREG),HL
22B6 C9       11440          RET
              11450 ;
              11460 ;        'q'uery ii - 'q'uery oo,dd
              11470 ;        input/output to port
              11480 ;
22B7 FE71     11490 QUERY    CP      'q'
22B9 2Ø24     11500          JR      NZ,DISKIO
22BB 3E1E     11510          LD      A,1EH           ;Clear to end of line
22BD CD42Ø6   11520          CALL    @DSP
22CØ CDE421   11530          CALL    HEXIN@          ;Get port number
22C3 C8       11540          RET     Z               ;Back if no value
22C4 4D       11550          LD      C,L
22C5 38Ø7     11560          JR      C,QUE1          ;If <ENTER>, do input
22C7 CDE421   11570          CALL    HEXIN@          ;Get byte to output
22CA C8       11580          RET     Z               ;Quit if none
22CB ED69     11590          OUT     (C),L           ;Do the output
22CD C9       11600          RET
22CE 3E3D     11610 QUE1     LD      A,'='           ;Dsply separator
22DØ CD42Ø6   11620          CALL    @DSP
22D3 ED78     11630          IN      A,(C)           ;Read the port and
22D5 CD1A22   11640          CALL    CV2HEX@         ;  dsply the value
22D8 C3C921   11650          JP      INPUT@
              11660 ;
              11670 ;        If a command is entered and not found in SYS5,
              11680 ;        SYS9 will be searched if the extended debugger
              11690 ;        is active.
              11700 ;
22DB 2AA419   11710 EXTDBG   LD      HL,(EXTDBG$)    ;Try extended debug
22DE E9       11720          JP      (HL)
              11730 ;
              11740 ;        disk i/o - d,c,s,r/w/*,addr,lth
              11750 ;
22DF D63Ø     11760 DISKIO   SUB     3ØH             ;Cvrt drive to binary
22E1 FE8      11770          CP      8               ;Check on max drive
22E3 3ØF6     11780          JR      NC,EXTDBG       ;Exit if not <Ø-7>
22E5 4F       11790          LD      C,A             ;Xfer drive # to reg C
```

```
22E6 CD1E1A   11800          CALL   @GTDCT          ;  & get the DCT
22E9 FD7E07   11810          LD     A,(IY+7)       ;Get sectors/cyl & heads
22EC E6E0     11820          AND    0E0H           ;Remove sectors/cyl
22EE 07       11830          RLCA                  ;  & keep # of heads
22EF 07       11840          RLCA                  ;Shift into bits 0-2
22F0 07       11850          RLCA
22F1 3C       11860          INC    A              ;Adj for zero offset
22F2 47       11870          LD     B,A
22F3 FD7E07   11880          LD     A,(IY+7)       ;# of sectors per cyl
22F6 E61F     11890          AND    1FH            ;Remove heads
22F8 3C       11900          INC    A              ;Adj for zero offset
22F9 67       11910          LD     H,A
22FA AF       11920          XOR    A              ;Accumulate total # of
22FB 84       11930 DIS1     ADD    A,H            ;Sectors per cyl
22FC 10FD     11940          DJNZ   DIS1
22FE FDCB046E 11950          BIT    5,(IY+4)       ;Test if 2-sided drive
2302 2801     11960          JR     Z,DIS2
2304 87       11970          ADD    A,A            ;Times 2 if 2-sided
2305 32A200   11980 DIS2     LD     (SAVTWO+1),A   ;Save sectors per cyl
2308 3E1E     11990          LD     A,1EH          ;Clear to end of line
230A CD4206   12000          CALL   @DSP
230D CDC921   12010          CALL   INPUT@         ;Input CYL #
2310 D8       12020          RET    C
2311 CDE421   12030          CALL   HEXIN@
2314 D8       12040          RET    C
2315 55       12050          LD     D,L            ;Cylinder entered?
2316 2003     12060          JR     NZ,DIS3
2318 FD5609   12070          LD     D,(IY+9)       ;P/u directory cyl
231B CDE421   12080 DIS3     CALL   HEXIN@
231E 5D       12090          LD     E,L            ;Sector entered?
231F 3E01     12100          LD     A,1            ;Init to 1 sector i/o
2321 2005     12110          JR     NZ,DIS4
2323 1E00     12120          LD     E,0            ;Default to sector 0
2325 3AA200   12130          LD     A,(SAVTWO+1)   ;Default to total sectors
2328 32A500   12140 DIS4     LD     (NXTBYT),A
232B D8       12150          RET    C
232C CDC921   12160          CALL   INPUT@         ;Get I/O direction (RW*)
232F D8       12170          RET    C
2330 47       12180          LD     B,A            ;Save i/o char in B
2331 CDC921   12190          CALL   INPUT@         ;Get buffer i/o address
2334 D8       12200          RET    C
2335 CDE421   12210          CALL   HEXIN@
2338 E5       12220          PUSH   HL             ;Save buffer address
2339 380B     12230          JR     C,DIS6
233B E5       12240          PUSH   HL
233C CDE421   12250          CALL   HEXIN@         ;Sector count entered?
233F 7D       12260          LD     A,L
2340 E1       12270          POP    HL
2341 2803     12280          JR     Z,DIS6         ;Go if no sector count
2343 32A500   12290          LD     (NXTBYT),A     ;Else update count
2346 78       12300 DIS6     LD     A,B            ;P/u i/o direction
2347 FE72     12310          CP     'r'            ;Read?
2349 2830     12320          JR     Z,DIS9
234B FE77     12330          CP     'w'            ;Write?
234D 2847     12340          JR     Z,DIS10
234F FE2A     12350          CP     '*'            ;Write to directory?
2351 284A     12360          JR     Z,DIS11
2353 24       12370 DIS7     INC    H              ;Bump up a buffer page
2354 1C       12380          INC    E              ;Bump sector number
2355 3AA200   12390          LD     A,(SAVTWO+1)   ;P/u max # sectors
2358 3D       12400          DEC    A              ;Compare max to where
```

```
2359 BB       12410          CP      E               ;  we are
235A 3003     12420          JR      NC,DIS8         ;Jump if more on cyl
235C 1E00     12430          LD      E,Ø             ;Reset sector # to Ø
235E 14       12440          INC     D               ;Bump cylinder
235F 3AA500   12450 DIS8     LD      A,(NXTBYT)      ;Reduce i/o sector count
2362 3D       12460          DEC     A
2363 32A500   12470          LD      (NXTBYT),A
2366 20DE     12480          JR      NZ,DIS6         ;Loop if not through
2368 E1       12490 DIS8A    POP     HL              ;Rcvr buffer start addr
2369 78       12500          LD      A,B             ;P/u i/o direction
236A FE72     12510          CP      'r'             ;Read?
236C C0       12520          RET     NZ              ;Ret if not read
236D 2E00     12530          LD      L,Ø             ;Reset memory buffer ptr
236F 22A600   12540          LD      (DSPADR),HL     ;  to display the 1st
2372 22A300   12550          LD      (NXTADR),HL     ;  sector read
2375 3E73     12560          LD      A,'s'           ;Set full screen mode
2377 32A100   12570          LD      (SAVTWO),A
237A C9       12580          RET
              12590 ;
237B          12600 DIS9     EQU     $
237B E5       12610          PUSH    HL
237C D5       12620          PUSH    DE
237D C5       12630          PUSH    BC
237E 54       12640          LD      D,H             ;Pass buffer to DE
237F 5D       12650          LD      E,L
2380 13       12660          INC     DE              ;Start +1
2381 3600     12670          LD      (HL),Ø          ;Clear a byte
2383 01FF00   12680          LD      BC,255          ;Length -1
2386 EDB0     12690          LDIR                    ;Clear buffer
2388 C1       12700          POP     BC              ;Unstack
2389 D1       12710          POP     DE
238A E1       12720          POP     HL
              12730 ;
238B CDF419   12740          CALL    @RDSEC          ;Read the sector
238E 28C3     12750          JR      Z,DIS7          ;Loop on read ok
2390 FE06     12760          CP      6               ;  or directory read
2392 28BF     12770          JR      Z,DIS7
2394 180C     12780          JR      DIS12           ;  else error
2396 CDE819   12790 DIS10    CALL    @WRSEC          ;Write sector
2399 28B8     12800          JR      Z,DIS7          ;Loop on write ok
239B 1805     12810          JR      DIS12
239D CDEC19   12820 DIS11    CALL    @WRSSC          ;Write system sector
23A0 28B1     12830          JR      Z,DIS7          ;Loop on write prot ok
              12840 ;
              12850 ;        disk i/o error output display routine
              12860 ;
23A2 D5       12870 DIS12    PUSH    DE              ;Save track & sector
23A3 F5       12880          PUSH    AF              ;Save error code
23A4 CD3122   12890          CALL    WRSPA@          ;Output a space
23A7 3E2A     12900          LD      A,'*'
23A9 CD4206   12910          CALL    @DSP            ;  followed by asterisk
23AC F1       12920          POP     AF
23AD CD1A22   12930          CALL    CV2HEX@         ;Write error #
23B0 3E2A     12940          LD      A,'*'
23B2 CD4206   12950          CALL    @DSP            ;  followed by space
23B5 CDC921   12960          CALL    INPUT@          ;Continue?
23B8 D1       12970          POP     DE              ;Rcvr track/sector
23B9 3098     12980          JR      NC,DIS7         ;Loop unless <ENTER>
23BB 18AB     12990          JR      DIS8A           ;Exit on <ENTER>
23BD          13000 LAST     EQU     $
              13010          IFGT    LAST,MAXCOR$-2
```

```
                13Ø2Ø           ERR     'Module too big'
                13Ø3Ø           ENDIF
23FE            13Ø4Ø           ORG     MAXCOR$-2
23FE BDØ5       13Ø5Ø           DW      LAST-SYS5       ;Overlay size
                ØØ18Ø ;
1EØØ            ØØ19Ø           END     SYS5
```

$?1	1E32	$?1Ø	1F1D
$?11	1F2E	$?12	1F38
$?13	1F8F	$?14	1F9B
$?15	1F9F	$?16	1FA4
$?17	1FC5	$?18	1FDF
$?19	1E37	$?2	2ØØF
$?2Ø	2Ø57	$?21	2Ø5C
$?22	2Ø61	$?23	2Ø62
$?24	2Ø66	$?25	2ØA6
$?26	2ØA9	$?27	2ØAA
$?28	2ØB7	$?28A	2ØF1
$?29	2ØF6	$?3	1E49
$?3Ø	2ØF9	$?31	2ØFC
$?32	21Ø2	$?33	21ØB
$?34	2117	$?35	211A
$?36	218Ø	$?37	218E
$?38	219A	$?39	219C
$?4	1EB4	$?4Ø	21BF
$?41	21C3	$?42	21C7
$?43	21CA	$?44	21E1
$?45	21EB	$?46	2223
$?47	222B	$?48	223B
$?5	1EC4	$?6	1EC5
$?8	1EEE	$?9	1F16
$A1	Ø3B7	$A2	Ø3B8
$A3	Ø3B9	$CKEOF	147Ø
@$SYS	Ø8FØ	@@1	ØØØØ
@@2	ØØØØ	@@3	ØØØØ
@@4	ØØØØ	@ABORT	1BØ8
@ADTSK	1CDA	@BANK	Ø877
@BKSP	1486	@BREAK	196F
@BYTEIO	13ØØ	@CHNIO	Ø689
@CKBRKC	Ø553	@CKDRV	1993
@CKEOF	158F	@CKTSK	1CF5
@CLOSE	1999	@CLS	Ø545
@CMNDI	197E	@CMNDR	197B
@CTL	Ø623	@DATE	Ø7A8
@DBGHK	199F	@DCINIT	19CØ
@DCRES	19C4	@DCSTAT	19B5
@DCTBYT	1A2B	@DEBUG	19AØ
@DECHEX	Ø3E1	@DIRCYL	18F7
@DIRRD	18BB	@DIRWR	18Ø3
@DIV16	Ø6E3	@DIV8	1927
@DODIR	19AF	@DOKEY	19A9
@DSP	Ø642	@DSPLY	Ø52D
@ERROR	1BØF	@EXIT	1BØB
@FEXT	1984	@FLAGS	196A
@FNAME	199C	@FRENCH	ØØØØ
@FSPEC	1981	@GATRD	1874
@GATWR	1875	@GERMAN	ØØØØ
@GET	Ø638	@GTDCB	199Ø
@GTDCT	1A1E	@GTMOD	19B2
@HDFMT	19E4	@HEX16	Ø7BD
@HEX8	Ø7C2	@HEXDEC	Ø6F6
@HIGH$	1948	@HITRD	1897
@HITWR	1898	@HZ5Ø	ØØØØ
@ICNFG	ØØ86	@INIT	198D
@INTL	ØØØØ	@IPL	1BF2
@JCL	Ø63Ø	@KBD	Ø635
@KEY	Ø628	@KEYIN	Ø585
@KITSK	ØØ89	@KLTSK	1CDØ
@LOAD	1B38	@LOC	14B3
@LOF	14DE	@LOGER	Ø5Ø3
@LOGOT	Ø5ØØ	@MOD2	ØØØØ
@MOD4	FFFF	@MSG	Ø53Ø
@MUL16	Ø6C9	@MUL8	19ØA
@NMI	ØØ66	@OPEN	198A
@OPREG	ØØ84	@PARAM	1987
@PAUSE	Ø382	@PEOF	14A2
@POSN	1434	@PRINT	Ø528
@PRT	Ø63D	@PUT	Ø645
@RAMDIR	19AC	@RDHDR	19D8
@RDSEC	19F4	@RDSSC	18D8
@RDTRK	19EØ	@READ	1513
@REMOVE	19A6	@RENAME	1996
@REW	149B	@RMTSK	1CD7
@RPTSK	1CEB	@RREAD	1473
@RSLCT	19D4	@RST0Ø	ØØØØ
@RST08	ØØØ8	@RST1Ø	ØØ1Ø
@RST18	ØØ18	@RST2Ø	ØØ2Ø
@RST28	ØØ28	@RST3Ø	ØØ3Ø
@RST38	ØØ38	@RSTNMI	ØFE9
@RSTOR	19C8	@RSTREG	Ø68Ø
@RUN	1B1D	@RWRIT	13AD
@SEEK	19DØ	@SEEKSC	1421
@SKIP	143Ø	@SLCT	19BC
@SOUND	Ø392	@STEPI	19CC
@TIME	Ø78D	@USA	FFFF
@VDCTL	ØB99	@VDCTL3	ØD38
@VER	156Ø	@VRSEC	19DC

@WEOF	14EC	@WHERE	1979	@WRITE	1531		
@WRSEC	19E8	@WRSSC	19EC	@WRTRK	19FØ		
@_VDCTL	ØD42	ADDR_2_ROWCOL	ØDF1	AFLAG$	ØØ6A		
AFREG	ØØA8	AHDSP	2Ø13	AHGET	2Ø29		
ATOF	22ØD	AUTO?	1FF1	BAR$	Ø2Ø1		
BLO1	2257	BLO2	226C	BLO3	2276		
BLO4	227B	BLOCK	223F	BOOTST$	439D		
BREAK?	1C6Ø	BRKVEC$	1C88	BUR$	Ø2ØØ		
CASHK$	ØA7B	CFCB$	ØØEØ	CFGFCB$	ØØEØ		
CFLAG$	ØØ6C	CKMOD@	1A7F	CKOPEN@	1568		
CMD_AH	1FD6	CMD_C	1E81	CMD_CI	2Ø8B		
CMD_D	1EAB	CMD_DEC	1EC9	CMD_G	1F82		
CMD_INC	1EB1	CMD_O	1ECE	CMD_R	2Ø3F		
CMD_S	1E9D	CMD_U	1EA1	CMD_X	1E9C		
CMND	1E4C	CONFIG$	2Ø3F	CORE$	Ø3ØØ		
CRTBGN$	F8ØØ	CV2HEX@	221A	CVB	22ØØ		
CYL_GRN	16AE	D@FBYT8	1A26	DATE$	ØØ33		
DAYTBL$	Ø4C7	DBGSV$	ØØAØ	DCBKL$	ØØ31		
DCT$	Ø47Ø	DCTBYT8@	1A29	DCTFLD@	1A34		
DFLAG$	ØØ6D	DIRBUF$	23ØØ	DIS1	22FB		
DIS1Ø	2396	DIS11	239D	DIS12	23A2		
DIS2	23Ø5	DIS3	231B	DIS4	2328		
DIS6	2346	DIS7	2353	DIS8	235F		
DIS8A	2368	DIS9	237B	DISKIO	22DF		
DIS_DO_RAM	Ø846	DODATA$	ØB94	DODCB$	Ø21Ø		
DO_CONTROL	ØC44	DO_DSPCHAR	ØCB8	DO_INVERT_DIS	ØC8C		
DO_INVERT_ENA	ØC89	DO_INVERT_OFF	ØC9B	DO_MASK	ØØØØ		
DO_RET	ØBCB	DO_RET1	ØBCC	DO_SCROLL	ØCCE		
DO_TABS	ØBEA	DSKTYP$	Ø4CØ	DSPADR	ØØA6		
DSPASC@	2Ø1B	DTPMT$	Ø4C2	DVREND$	ØFF4		
DVRHI$	Ø2Ø6	ED_TAB	215Ø	EFLAG$	ØØ6E		
ENADIS_DO_RAM	Ø817	EXTDBG	22DB	EXTDBG$	19A4		
FDDINT$	ØØØE	FEMSK$	ØØ6F	FIL1	22A2		
FILL	228B	FLGTAB$	ØØ6A	FLGTBL	1F7A		
FULDSP	1F4D	GETASC@	2Ø31	GET_@_ROWCOL	ØDAE		
GRPHIC	21A4	HERTZ$	Ø75Ø	HEXIN@	21E4		
HIGH$	Ø4ØE	HKRES$	1A6C	HLREG	ØØAE		
IFLAG$	ØØ72	INBUF$	Ø42Ø	INPUC@	21D5		
INPUT@	21C9	INTIM$	ØØ3C	INTMSK$	ØØ3D		
INTVC$	ØØ3E	IXREG	ØØB8	IYREG	ØØBA		
JCLCB$	Ø2Ø3	JDCB$	ØØ24	JFCB$	ØØCØ		
JLDCB$	Ø23Ø	JRET$	ØØ26	JUMP	22AB		
KCK@	Ø7D6	KFLAG$	ØØ74	KIDATA$	Ø8FC		
KIDCB$	Ø2Ø8	LAST	23BD	LBANK$	Ø2Ø2		
LDRV$	ØØ23	LFLAG$	ØØ75	LNKFCB@	1566		
LOW$	ØØ1E	LSVC$	ØØØD	MAXCOR$	24ØØ		
MAXDAY$	Ø4Ø1	MINCOR$	3ØØØ	MODOUT$	ØØ76		
MONTBL$	Ø4DC	NFLAG$	ØØ77	NOFLG	1F2B		
NXTADR	ØØA3	NXTBYT	ØØA5	OPREG$	ØØ78		
OPREG_SV_AREA	Ø86E	OPREG_SV_PTR	Ø835	OP_TAB	211F		
ORARET@	14DC	OSRLS$	ØØ3B	OSVER$	ØØ85		
OVRLY$	ØØ69	PAKNAM$	Ø41Ø	PAUSE@	Ø382		
PCREG	ØØBE	PCSAVE$	Ø7AF	PDRV$	ØØ1B		
PHIGH$	ØØ1C	PRDCB$	Ø218	PUTA@DE	ØDCD		
PUT_@	ØDCA	PUT_@_ROWCOL	ØDC6	QUE1	22CE		
QUERY	22B7	REGSAV	ØØBD	REGTBL	1F56		
RFLAG$	ØØ7B	ROWCOL_2_ADDR	ØDDØ	RST38@	1BFF		
RSTOR$	Ø4C4	RWRIT@	13A2	S1DCB$	Ø238		
SAVONE	ØØAØ	SAVTWO	ØØA1	SBUFF$	1DØØ		
SET@EXEC	1A79	SET_SCROLL	ØCF3	SFCB$	ØØ8C		
SFLAG$	ØØ7C	SIDCB$	Ø22Ø	SODCB$	Ø228		

```
SPACE4$         2142 SPREG          ØØBC STACK$         Ø38Ø
START$          ØØØØ SVCRET$        ØØØB SVCTAB$        Ø1ØØ
SYS5            1EØØ SYSERR$        1B13 TCB$           ØØ4E
TFLAG$          ØØ7D TIME$          ØØ2D TIMER$         ØØ2C
TIMSL$          ØØ2B TIMTSK$        Ø713 TMPMT$         Ø4C3
TRACE_INT       Ø7B1 TYP3           2Ø24 TYP4           2Ø26
TYPHK$          ØA8F TYPTSK$        ØB26 USTOR$         ØØ13
VFLAG$          ØØ7F WR1HEX@        2211 WR2HEX@        2215
WR3BYT          2235 WRHEX          222E WRINT$         ØØ8Ø
WRMEM           2173 WRREGS         1ED5 WRSPA@         2231
XY_TAB          2157 ZERO$          Ø4Ø1 ZEROA@         13AØ

1EØØ is the transfer address
ØØØØØ Total errors
```

NOTES:

NOTES:

SYS9 is the extended system debugger. It resides in high memory, and handles additional functions that require more memory than was available for SYS5, which had to reside in the overlay region. There are no SVCs directly handled by SYS9.

```
                00100 ;SYS9/ASM - LS-DOS 6.2
0000            00110         TITLE   <SYS9 - LS-DOS 6.2>
                00120 ;
                00130 *LIST   OFF                     ;Get SYS5/EQU
                00150 *LIST   ON
0000            00160 *GET    COPYCOM:3               ;Copyright message
                03740 ; COPYCOM - File for Copyright COMment block
                03750 ;
0000            03760         COM     '<*(C) 1982,83,84 by LSI*>'
                03770 ;
00A0            00170         ORG     0A0H
                00180 ;
0001            00190 SAVONE  DS      1
0001            00200 SAVTWO  DS      1
0001            00210         DS      1               ;Space for saved byte (1)
0002            00220 NXTADR  DS      2
0001            00230 NXTBYT  DS      1
0002            00240 DSPADR  DS      2
0006            00250 AFREG   DS      6               ;AF, BC, DE
0002            00260 HLREG   DS      2               ;HL
0008            00270         DS      8               ;AF', BC', DE', HL'
0002            00280 IXREG   DS      2               ;IX
0002            00290 IYREG   DS      2               ;IY
0001            00300 SPREG   DS      1               ;SP
0001            00310 REGSAV  DS      1
0002            00320 PCREG   DS      2               ;PC
                00330 ;
1E00            00340         ORG     1E00H
                00350 ;
1E00 E670       00360 SYS9    AND     70H
1E02 C8         00370         RET     Z               ;Back on zero entry
1E03 2AA419     00380         LD      HL,(EXTDBG$)    ;P/u hook address
1E06 AF         00390         XOR     A               ;See if already resident
1E07 1124EB     00400         LD      DE,-ORARET@
1E0A ED5A       00410         ADC     HL,DE           ;ADD does not effect Z
1E0C C0         00420         RET     NZ              ;Ret if resident already
1E0D 2A0E04     00430         LD      HL,(HIGH$)      ;Change high$ to provide
1E10 222B1E     00440         LD      (DEBUGE+2),HL   ;Stuff last byte used
1E13 01560200   00450         LD      BC,LAST-DEBUGE  ;Room for relocating
1E16 AF         00460         XOR     A               ;  this module to high
1E17 ED42       00470         SBC     HL,BC
1E19 220E04     00480         LD      (HIGH$),HL
1E1C 23         00490         INC     HL              ;Pt to new entry point
1E1D E5         00500         PUSH    HL              ;Save it for later
1E1E EB         00510         EX      DE,HL           ;Move extended debug
1E1F 21291E     00520         LD      HL,DEBUGE       ;  up to top of core
1E22 EDB0       00530         LDIR
1E24 E1         00540         POP     HL              ;Rcvr pointer to ept
1E25 22A419     00550         LD      (EXTDBG$),HL    ;  & reset sysres vector
1E28 C9         00560         RET
                00570 ;
                00580 ;       Start of extended debug utility
                00590 ;
1E29 180D       00600 DEBUGE  JR      NEXT
1E2B 0000       00610         DW      $-$
1E2D 06         00620         DB      6,'EXTDBG'
     45 58 54 44 42 47
1E34 0000       00630         DW      0,0
     0000
                00640 ;
                00650 ;       'n'ext aaaa - position to next relative block
```

```
                    00660 ;       used in stepping through a program file
                    00670 ;       dumped to core in load module format
                    00680 ;
1E38 FE3E           00690 NEXT    CP      'n'-'0'
1E3A 201D           00700         JR      NZ,ENTER
1E3C 2AA300         00710         LD      HL,(NXTADR)     ;Init if no further input
1E3F CDE421         00720         CALL    HEXIN@          ;Arg aaaa entered?
1E42 23             00730         INC     HL              ;Bump from type to length
1E43 1600           00740         LD      D,0
1E45 5E             00750         LD      E,(HL)          ;P/u block length
1E46 7B             00760         LD      A,E
1E47 FE03           00770         CP      3               ;Len= 0,1,2?
1E49 3001           00780         JR      NC,NEX1         ;If len=0,1,2 (256-8),
1E4B 14             00790         INC     D               ;  next block is +257-259
1E4C 13             00800 NEX1    INC     DE              ;Bump by one for len byte
1E4D 19             00810         ADD     HL,DE           ;Add length to index
1E4E 22A300         00820         LD      (NXTADR),HL     ;Next block
1E51 7D             00830         LD      A,L             ;Now set up the display
1E52 E6C0           00840         AND     0C0H            ;Address
1E54 6F             00850         LD      L,A
1E55 22A600         00860         LD      (DSPADR),HL
1E58 C9             00870         RET
                    00880 ;
                    00890 ;       Enter hex data into memory
                    00900 ;
1E59 FE35           00910 ENTER   CP      'e'-'0'         ;'e'nter <addr>
1E5B 202F           00920         JR      NZ,LOCATE
1E5D 2AA300         00930         LD      HL,(NXTADR)     ;Pt to current address
1E60 CDE421         00940         CALL    HEXIN@          ;Get new address to enter
1E63 22A300         00950         LD      (NXTADR),HL
1E66 D8             00960         RET     C               ;Back on <ENTER>
1E67 2006           00970         JR      NZ,ENT1         ;Go if new addr
1E69 CD1522         00980         CALL    WR2HEX@         ;  else dsply default
1E6C CD3122         00990         CALL    WRSPA@
1E6F 3E1E           01000 ENT1    LD      A,1EH           ;Clear the line
1E71 CD4206         01010         CALL    @DSP
1E74 CD1122         01020 ENT2    CALL    WR1HEX@         ;Set up the display
1E77 2B             01030         DEC     HL
1E78 3E2D           01040         LD      A,'-'
1E7A CD4206         01050         CALL    @DSP
1E7D EB             01060         EX      DE,HL
1E7E CDE421         01070         CALL    HEXIN@          ;Get the modify info
1E81 EB             01080         EX      DE,HL
1E82 2801           01090         JR      Z,ENT3          ;No change if no new data
1E84 73             01100         LD      (HL),E          ;  else update byte
1E85 D8             01110 ENT3    RET     C               ;Back if <ENTER> pressed
1E86 23             01120         INC     HL
1E87 22A300         01130         LD      (NXTADR),HL     ;Index to next address
1E8A 18E8           01140         JR      ENT2
                    01150 ;
                    01160 ;       'l'ocate aaaa,dd
                    01170 ;
1E8C FE3C           01180 LOCATE  CP      'l'-'0'
1E8E 2043           01190         JR      NZ,TYPE
1E90 2AA300         01200         LD      HL,(NXTADR)     ;Default current address
1E93 23             01210         INC     HL
1E94 CDE421         01220         CALL    HEXIN@          ;Prompt new address
1E97 22A300         01230         LD      (NXTADR),HL
1E9A 200E           01240         JR      NZ,LOC1         ;Go if new addr
1E9C F5             01250         PUSH    AF              ;Save flags
1E9D CD1522         01260         CALL    WR2HEX@         ;Display default
```

```
1EA0 3E2C     01270          LD      A,','
1EA2 CD4206   01280          CALL    @DSP
1EA5 F1       01290          POP     AF
1EA6 3AA500   01300          LD      A,(NXTBYT)      ;P/u default byte
1EA9 6F       01310          LD      L,A
1EAA 380B     01320 LOC1     JR      C,LOC2          ;Go if <ENTER> used
1EAC CDE421   01330          CALL    HEXIN@          ;  else get new byte
1EAF 2806     01340          JR      Z,LOC2          ;Go if none entered
1EB1 7D       01350          LD      A,L
1EB2 32A500   01360          LD      (NXTBYT),A      ;  else set byte to find
1EB5 1804     01370          JR      LOC3
1EB7 7D       01380 LOC2     LD      A,L             ;Display byte info
1EB8 CD1A22   01390          CALL    CV2HEX@
1EBB 2AA300   01400 LOC3     LD      HL,(NXTADR)     ;Set up for search
1EBE 3AA500   01410          LD      A,(NXTBYT)
1EC1 010000   01420          LD      BC,0            ;Set loop to 64K
1EC4 EDB1     01430          CPIR                    ;Find a match
1EC6 C0       01440          RET     NZ              ;Back if none
1EC7 2B       01450          DEC     HL
1EC8 22A300   01460          LD      (NXTADR),HL     ;Store new mod addr
1ECB 7D       01470          LD      A,L
1ECC E6C0     01480          AND     0C0H
1ECE 6F       01490          LD      L,A
1ECF 22A600   01500          LD      (DSPADR),HL
1ED2 C9       01510          RET
              01520 ;
              01530 ;        't'ype aaaa - type ascii into memory
              01540 ;
1ED3 FE44     01550 TYPE     CP      't'-'0'
1ED5 2030     01560          JR      NZ,VERIFY
1ED7 2AA300   01570          LD      HL,(NXTADR)     ;Default current address
1EDA CDE421   01580          CALL    HEXIN@          ;Prompt for new address
1EDD 22A300   01590          LD      (NXTADR),HL
1EE0 D8       01600          RET     C               ;Back on <ENTER>
1EE1 2003     01610          JR      NZ,TYP1         ;Go if new addr
1EE3 CD1522   01620          CALL    WR2HEX@         ;  else dsply default
1EE6 3E1E     01630 TYP1     LD      A,1EH           ;Clear to end of line
1EE8 CD4206   01640          CALL    @DSP
1EEB CD3122   01650 TYP2     CALL    WRSPA@
1EEE CD1B20   01660          CALL    DSPASC@         ;Display current contents
1EF1 3E2D     01670          LD      A,'-'
1EF3 CD4206   01680          CALL    @DSP
1EF6 E5       01690          PUSH    HL              ;Provide lower/upper
1EF7 CD3120   01700          CALL    GETASC@         ;  case entry
1EFA E1       01710          POP     HL              ;  conversion
1EFB D8       01720          RET     C
1EFC FE20     01730          CP      20H             ;Advance on space
1EFE 2801     01740          JR      Z,TYP5
1F00 77       01750          LD      (HL),A          ;Store new info
1F01 23       01760 TYP5     INC     HL
1F02 22A300   01770          LD      (NXTADR),HL     ;Advance the location
1F05 18E4     01780          JR      TYP2
              01790 ;
              01800 ;        'v'erify aaaa,bbbb,lth - verify block
              01810 ;
1F07 FE46     01820 VERIFY   CP      'v'-'0'
1F09 2057     01830          JR      NZ,WORD
1F0B 2AA600   01840          LD      HL,(DSPADR)     ;1st default start of dsp
1F0E CDE421   01850          CALL    HEXIN@          ;Prompt new start
1F11 22A600   01860          LD      (DSPADR),HL
1F14 200A     01870          JR      NZ,VER1         ;Go if address entered
```

```
1F16 F5        01880          PUSH    AF
1F17 CD1522    01890          CALL    WR2HEX@         ;  else dsply default
1F1A 3E2C      01900          LD      A,','
1F1C CD42Ø6    01910          CALL    @DSP
1F1F F1        01920          POP     AF
1F2Ø 3815      01930 VER1     JR      C,VER2          ;Jump if <ENTER> used prev.
1F22 2AA3ØØ    01940          LD      HL,(NXTADR)     ;2nd default current mod addr
1F25 CDE421    01950          CALL    HEXIN@          ;Prompt new 2nd start
1F28 22A3ØØ    01960          LD      (NXTADR),HL
1F2B 2ØØA      01970          JR      NZ,VER2         ;Go if entered
1F2D F5        01980          PUSH    AF
1F2E CD1522    01990          CALL    WR2HEX@         ;  else dsply default
1F31 3E2C      02000          LD      A,','
1F33 CD42Ø6    02010          CALL    @DSP
1F36 F1        02020          POP     AF
1F37 21ØØØØ    02030 VER2     LD      HL,Ø            ;Default length to verify
1F3A 38ØA      02040          JR      C,VER3          ;Go if <ENTER> sued prev
1F3C CDE421    02050          CALL    HEXIN@          ;Get new length
1F3F 2ØØ5      02060          JR      NZ,VER3         ;Go if new len entered
1F41 E5        02070          PUSH    HL
1F42 CD1522    02080          CALL    WR2HEX@         ;Dsply default len
1F45 E1        02090          POP     HL
1F46 44        02100 VER3     LD      B,H             ;Xfer length to BC
1F47 4D        02110          LD      C,L
1F48 2AA6ØØ    02120          LD      HL,(DSPADR)     ;Set up for compare
1F4B ED5BA3ØØ  02130          LD      DE,(NXTADR)
1F4F 1A        02140 VER4     LD      A,(DE)
1F5Ø BE        02150          CP      (HL)            ;Compare the two locations
1F51 2ØØ7      02160          JR      NZ,VER5         ;Go on non-match
1F53 13        02170          INC     DE              ;  else inc pointers
1F54 23        02180          INC     HL              ;  and loop for length
1F55 ØB        02190          DEC     BC
1F56 78        02200          LD      A,B
1F57 B1        02210          OR      C
1F58 2ØF5      02220          JR      NZ,VER4
1F5A ED53A3ØØ  02230 VER5     LD      (NXTADR),DE     ;Store non-match or end of
1F5E 22A6ØØ    02240          LD      (DSPADR),HL     ;  block
1F61 C9        02250          RET
               02260 ;
               02270 ;        'w'ord aaaa,dddd - search for word dddd
               02280 ;
1F62 FE47      02290 WORD     CP      'w'-'Ø'
1F64 2Ø52      02300          JR      NZ,PRINT
1F66 2AA3ØØ    02310          LD      HL,(NXTADR)     ;Default current address
1F69 23        02320          INC     HL              ;  but bypass next word
1F6A 23        02330          INC     HL
1F6B CDE421    02340          CALL    HEXIN@          ;Get new start
1F6E 22A3ØØ    02350          LD      (NXTADR),HL
1F71 2Ø12      02360          JR      NZ,WOR1         ;Go if value entered
1F73 F5        02370          PUSH    AF              ;  else display default
1F74 CD1522    02380          CALL    WR2HEX@
1F77 3E2C      02390          LD      A,','
1F79 CD42Ø6    02400          CALL    @DSP
1F7C F1        02410          POP     AF
1F7D 3AA5ØØ    02420          LD      A,(NXTBYT)      ;Get next default
1F8Ø 6F        02430          LD      L,A
1F81 3AA2ØØ    02440          LD      A,(SAVTWO+1)
1F84 67        02450          LD      H,A
1F85 38ØF      02460 WOR1     JR      C,WOR2          ;Go if <ENTER>
1F87 CDE421    02470          CALL    HEXIN@          ;Get next value
1F8A 28ØA      02480          JR      Z,WOR2          ;Go if default
```

```
1F8C 7D       Ø249Ø         LD      A,L             ;Store new value
1F8D 32A5ØØ   Ø25ØØ         LD      (NXTBYT),A
1F9Ø 7C       Ø251Ø         LD      A,H
1F91 32A2ØØ   Ø252Ø         LD      (SAVTWO+1),A
1F94 18Ø3     Ø253Ø         JR      WOR3
1F96 CD1522   Ø254Ø WOR2    CALL    WR2HEX@         ;Display value
1F99 2AA3ØØ   Ø255Ø WOR3    LD      HL,(NXTADR)     ;Start looking here
1F9C Ø1ØØØØ   Ø256Ø         LD      BC,Ø            ;Init count to 64K
1F9F 3AA5ØØ   Ø257Ø WOR4    LD      A,(NXTBYT)
1FA2 EDB1     Ø258Ø         CPIR                    ;Find first match
1FA4 CØ       Ø259Ø         RET     NZ              ;Return if none
1FA5 3AA2ØØ   Ø26ØØ         LD      A,(SAVTWO+1)    ;Get 2nd half of word
1FA8 BE       Ø261Ø         CP      (HL)            ;Is a match?
1FA9 2ØF4     Ø262Ø         JR      NZ,WOR4         ;Continue if not
1FAB 2B       Ø263Ø         DEC     HL
1FAC 2B       Ø264Ø         DEC     HL              ;Pt 1 byte before
1FAD 22A3ØØ   Ø265Ø         LD      (NXTADR),HL     ;  and save that address
1FBØ 7D       Ø266Ø         LD      A,L
1FB1 E6CØ     Ø267Ø         AND     ØCØH
1FB3 6F       Ø268Ø         LD      L,A
1FB4 22A6ØØ   Ø269Ø         LD      (DSPADR),HL     ;New display start
1FB7 C9       Ø27ØØ         RET
              Ø271Ø ;
              Ø272Ø ;       'p'rint aaaa,bbbb - print memory
              Ø273Ø ;
1FB8 FE4Ø     Ø274Ø PRINT   CP      'p'-3ØH         ;If command is not 'P',
1FBA CØ       Ø275Ø PRI1    RET     NZ              ;  back to SYS5
1FBB CDE421   Ø276Ø         CALL    HEXIN@          ;Get start
1FBE C8       Ø277Ø         RET     Z               ;Back if no start addr
1FBF E5       Ø278Ø         PUSH    HL
1FCØ CDE421   Ø279Ø         CALL    HEXIN@          ;Get end
1FC3 E3       Ø28ØØ         EX      (SP),HL
1FC4 C1       Ø281Ø         POP     BC              ;Start in HL, end in BC
1FC5 C8       Ø282Ø         RET     Z               ;Back if no end addr
1FC6 7D       Ø283Ø         LD      A,L             ;Round to multiple of 16
1FC7 E6FØ     Ø284Ø         AND     ØFØH
1FC9 6F       Ø285Ø         LD      L,A
1FCA 3EØD     Ø286Ø         LD      A,ØDH           ;Send 2 blank lines to
1FCC CD3DØ6   Ø287Ø         CALL    @PRT            ;  the printer
1FCF CD3DØ6   Ø288Ø         CALL    @PRT
1FD2 E5       Ø289Ø PRI2    PUSH    HL              ;Routine to write HL
1FD3 7C       Ø29ØØ         LD      A,H             ;  as 4 hex digits
1FD4 1F       Ø291Ø         RRA
1FD5 1F       Ø292Ø         RRA
1FD6 1F       Ø293Ø         RRA
1FD7 1F       Ø294Ø         RRA
1FD8 E6ØF     Ø295Ø         AND     ØFH
1FDA C69Ø     Ø296Ø         ADD     A,9ØH
1FDC 27       Ø297Ø         DAA
1FDD CE4Ø     Ø298Ø         ADC     A,4ØH
1FDF 27       Ø299Ø         DAA
1FEØ CD3DØ6   Ø3ØØØ         CALL    @PRT            ;1st one done
1FE3 7C       Ø3Ø1Ø         LD      A,H
1FE4 E6ØF     Ø3Ø2Ø         AND     ØFH
1FE6 C69Ø     Ø3Ø3Ø         ADD     A,9ØH
1FE8 27       Ø3Ø4Ø         DAA
1FE9 CE4Ø     Ø3Ø5Ø         ADC     A,4ØH
1FEB 27       Ø3Ø6Ø         DAA
1FEC CD3DØ6   Ø3Ø7Ø         CALL    @PRT            ;2nd one done
1FEF 7D       Ø3Ø8Ø         LD      A,L
1FFØ 1F       Ø3Ø9Ø         RRA
```

```
1FF1 1F       Ø31ØØ          RRA
1FF2 1F       Ø311Ø          RRA
1FF3 1F       Ø312Ø          RRA
1FF4 E6ØF     Ø313Ø          AND     ØFH
1FF6 C69Ø     Ø314Ø          ADD     A,9ØH
1FF8 27       Ø315Ø          DAA
1FF9 CE4Ø     Ø316Ø          ADC     A,4ØH
1FFB 27       Ø317Ø          DAA
1FFC CD3DØ6   Ø318Ø          CALL    @PRT            ;3rd one done
1FFF 7D       Ø319Ø          LD      A,L
2ØØØ E6ØF     Ø32ØØ          AND     ØFH
2ØØ2 C69Ø     Ø321Ø          ADD     A,9ØH
2ØØ4 27       Ø322Ø          DAA
2ØØ5 CE4Ø     Ø323Ø          ADC     A,4ØH
2ØØ7 27       Ø324Ø          DAA
2ØØ8 CD3DØ6   Ø325Ø          CALL    @PRT            ;4th one done
2ØØB 3E2Ø     Ø326Ø          LD      A,2ØH           ;  & 2 spaces
2ØØD CD3DØ6   Ø327Ø          CALL    @PRT
2Ø1Ø CD3DØ6   Ø328Ø          CALL    @PRT
2Ø13 18Ø2     Ø329Ø          JR      PRI4
2Ø15 18BB     Ø33ØØ PRI3     JR      PRI2
              Ø331Ø ;
              Ø332Ø ;        Write a byte in hex
              Ø333Ø ;
2Ø17 7E       Ø334Ø PRI4     LD      A,(HL)
2Ø18 1F       Ø335Ø          RRA
2Ø19 1F       Ø336Ø          RRA
2Ø1A 1F       Ø337Ø          RRA
2Ø1B 1F       Ø338Ø          RRA
2Ø1C E6ØF     Ø339Ø          AND     ØFH
2Ø1E C69Ø     Ø34ØØ          ADD     A,9ØH
2Ø2Ø 27       Ø341Ø          DAA
2Ø21 CE4Ø     Ø342Ø          ADC     A,4ØH
2Ø23 27       Ø343Ø          DAA
2Ø24 CD3DØ6   Ø344Ø          CALL    @PRT            ;Output it
2Ø27 7E       Ø345Ø          LD      A,(HL)
2Ø28 E6ØF     Ø346Ø          AND     ØFH
2Ø2A C69Ø     Ø347Ø          ADD     A,9ØH
2Ø2C 27       Ø348Ø          DAA
2Ø2D CE4Ø     Ø349Ø          ADC     A,4ØH
2Ø2F 27       Ø35ØØ          DAA
2Ø3Ø CD3DØ6   Ø351Ø          CALL    @PRT            ;Output it
2Ø33 3E2Ø     Ø352Ø          LD      A,2ØH
2Ø35 CD3DØ6   Ø353Ø          CALL    @PRT            ;  & a space
2Ø38 23       Ø354Ø          INC     HL              ;Pt to next byte
2Ø39 7D       Ø355Ø          LD      A,L             ;Test multiple of 16
2Ø3A E6ØF     Ø356Ø          AND     ØFH
2Ø3C 28Ø9     Ø357Ø          JR      Z,PRI5
2Ø3E E6Ø3     Ø358Ø          AND     3               ;Space on multiple of 4
2Ø4Ø 3E2Ø     Ø359Ø          LD      A,2ØH
2Ø42 CC3DØ6   Ø36ØØ          CALL    Z,@PRT
2Ø45 18DØ     Ø361Ø          JR      PRI4
2Ø47 3E2Ø     Ø362Ø PRI5     LD      A,2ØH           ;Space at end of 16
2Ø49 CD3DØ6   Ø363Ø          CALL    @PRT
2Ø4C E1       Ø364Ø          POP     HL
2Ø4D 7E       Ø365Ø PRI6     LD      A,(HL)          ;Print in ASCII if
2Ø4E FE2Ø     Ø366Ø          CP      2ØH             ;  printable; else
2Ø5Ø 38Ø4     Ø367Ø          JR      C,PRI7          ;  convert to '.'
2Ø52 FE8Ø     Ø368Ø          CP      8ØH
2Ø54 38Ø2     Ø369Ø          JR      C,PRI8
2Ø56 3E2E     Ø37ØØ PRI7     LD      A,'.'
```

```
2Ø58 CD3DØ6   Ø371Ø PRI8   CALL   @PRT
2Ø5B 23       Ø372Ø        INC    HL              ;Loop until 16 chars
2Ø5C 7D       Ø373Ø        LD     A,L
2Ø5D E6ØF     Ø374Ø        AND    ØFH
2Ø5F 2ØEC     Ø375Ø        JR     NZ,PRI6
2Ø61 3EØD     Ø376Ø        LD     A,ØDH           ;  then a new line
2Ø63 CD3DØ6   Ø377Ø        CALL   @PRT
2Ø66 E5       Ø378Ø        PUSH   HL
2Ø67 7D       Ø379Ø        LD     A,L             ;Check if HL is ØØØØ
2Ø68 B4       Ø38ØØ        OR     H
2Ø69 2ØØ3     Ø381Ø        JR     NZ,PRI9         ;  is OK > continue
2Ø6B E1       Ø382Ø        POP    HL
2Ø6C 18Ø6     Ø383Ø        JR     PRI1Ø           ;Get OUT now
2Ø6E AF       Ø384Ø PRI9   XOR    A               ;Ck on finished
2Ø6F ED42     Ø385Ø        SBC    HL,BC
2Ø71 E1       Ø386Ø        POP    HL
2Ø72 38A1     Ø387Ø        JR     C,PRI3
2Ø74 3EØD     Ø388Ø PRI1Ø  LD     A,ØDH           ;3 new lines if done
2Ø76 CD3DØ6   Ø389Ø        CALL   @PRT
2Ø79 CD3DØ6   Ø39ØØ        CALL   @PRT
2Ø7C C33DØ6   Ø391Ø        JP     @PRT
2Ø7F          Ø392Ø LAST   EQU    $
              Ø393Ø        IFGT   $,DIRBUF$
              Ø394Ø        ERR    'Module too big'
              Ø395Ø        ENDIF
23FE          Ø396Ø        ORG    MAXCOR$-2
23FE 7FØ2     Ø397Ø        DW     LAST-SYS9       ;Overlay size
              Ø398Ø ;
1EØØ          Ø399Ø        END    SYS9
```

$?1	1E32	$?1Ø	1F1D
$?11	1F2E	$?12	1F38
$?13	1F8F	$?14	1F9B
$?15	1F9F	$?16	1FA4
$?17	1FC5	$?18	1FDF
$?19	2ØØF	$?2	1E37
$?2Ø	2Ø57	$?21	2Ø5C
$?22	2Ø61	$?23	2Ø62
$?24	2Ø66	$?25	2ØA6
$?26	2ØA9	$?27	2ØAA
$?28	2ØB7	$?28A	2ØF1
$?29	2ØF6	$?3	1E49
$?3Ø	2ØF9	$?31	2ØFC
$?32	21Ø2	$?33	21ØB
$?34	2117	$?35	211A
$?36	218Ø	$?37	218E
$?38	219A	$?39	219C
$?4	1EB4	$?4Ø	21BF
$?41	21C3	$?42	21C7
$?43	21CA	$?44	21E1
$?45	21EB	$?46	2223
$?47	222B	$?48	223B
$?5	1EC4	$?6	1EC5
$?8	1EEE	$?9	1F16
$A1	Ø3B7	$A2	Ø3B8
$A3	Ø3B9	$CKEOF	147Ø
@$SYS	Ø8FØ	@@1	ØØØØ
@@2	ØØØØ	@@3	ØØØØ
@@4	ØØØØ	@ABORT	1BØ8
@ADTSK	1CDA	@BANK	Ø877
@BKSP	1486	@BREAK	196F
@BYTEIO	13ØØ	@CHNIO	Ø689
@CKBRKC	Ø553	@CKDRV	1993
@CKEOF	158F	@CKTSK	1CF5
@CLOSE	1999	@CLS	Ø545
@CMNDI	197E	@CMNDR	197B
@CTL	Ø623	@DATE	Ø7A8
@DBGHK	199F	@DCINIT	19CØ
@DCRES	19C4	@DCSTAT	19B5
@DCTBYT	1A2B	@DEBUG	19AØ
@DECHEX	Ø3E1	@DIRCYL	18F7
@DIRRD	18BB	@DIRWR	18Ø3
@DIV16	Ø6E3	@DIV8	1927
@DODIR	19AF	@DOKEY	19A9
@DSP	Ø642	@DSPLY	Ø52D
@ERROR	1BØF	@EXIT	1BØB
@FEXT	1984	@FLAGS	196A
@FNAME	199C	@FRENCH	ØØØØ
@FSPEC	1981	@GATRD	1874
@GATWR	1875	@GERMAN	ØØØØ
@GET	Ø638	@GTDCB	199Ø
@GTDCT	1A1E	@GTMOD	19B2
@HDFMT	19E4	@HEX16	Ø7BD
@HEX8	Ø7C2	@HEXDEC	Ø6F6
@HIGH$	1948	@HITRD	1897
@HITWR	1898	@HZ5Ø	ØØØØ
@ICNFG	ØØ86	@INIT	198D
@INTL	ØØØØ	@IPL	1BF2
@JCL	Ø63Ø	@KBD	Ø635
@KEY	Ø628	@KEYIN	Ø585
@KITSK	ØØ89	@KLTSK	1CDØ
@LOAD	1B38	@LOC	14B3
@LOF	14DE	@LOGER	Ø5Ø3
@LOGOT	Ø5ØØ	@MOD2	ØØØØ
@MOD4	FFFF	@MSG	Ø53Ø
@MUL16	Ø6C9	@MUL8	19ØA
@NMI	ØØ66	@OPEN	198A
@OPREG	ØØ84	@PARAM	1987
@PAUSE	Ø382	@PEOF	14A2
@POSN	1434	@PRINT	Ø528
@PRT	Ø63D	@PUT	Ø645
@RAMDIR	19AC	@RDHDR	19D8
@RDSEC	19F4	@RDSSC	18D8
@RDTRK	19EØ	@READ	1513
@REMOVE	19A6	@RENAME	1996
@REW	149B	@RMTSK	1CD7
@RPTSK	1CEB	@RREAD	1473
@RSLCT	19D4	@RST0Ø	ØØØØ
@RST08	ØØØ8	@RST1Ø	ØØ1Ø
@RST18	ØØ18	@RST2Ø	ØØ2Ø
@RST28	ØØ28	@RST3Ø	ØØ3Ø
@RST38	ØØ38	@RSTNMI	ØFE9
@RSTOR	19C8	@RSTREG	Ø68Ø
@RUN	1B1D	@RWRIT	13AD
@SEEK	19DØ	@SEEKSC	1421
@SKIP	143Ø	@SLCT	19BC
@SOUND	Ø392	@STEPI	19CC
@TIME	Ø78D	@USA	FFFF
@VDCTL	ØB99	@VDCTL3	ØD38
@VER	156Ø	@VRSEC	19DC

@WEOF	14EC	@WHERE	1979	@WRITE	1531
@WRSEC	19E8	@WRSSC	19EC	@WRTRK	19F0
@_VDCTL	0D42	ADDR_2_ROWCOL	0DF1	AFLAG$	006A
AFREG	00A8	AUTO?	1FF1	BAR$	0201
BOOTST$	439D	BREAK?	1C60	BRKVEC$	1C88
BUR$	0200	CASHK$	0A7B	CFCB$	00E0
CFGFCB$	00E0	CFLAG$	006C	CKMOD@	1A7F
CKOPEN@	1568	CMD_AH	1FD6	CMD_C	1E81
CMD_CI	208B	CMD_D	1EAB	CMD_DEC	1EC9
CMD_G	1F82	CMD_INC	1EB1	CMD_O	1ECE
CMD_R	203F	CMD_S	1E9D	CMD_U	1EA1
CMD_X	1E9C	CONFIG$	203F	CORE$	1948
CRTBGN$	F800	CV2HEX@	221A	CYL_GRN	16AE
D@FBYT8	1A26	DATE$	0033	DAYTBL$	04C7
DBGSV$	00A0	DCBKL$	0031	DCT$	0470
DCTBYT8@	1A29	DCTFLD@	1A34	DEBUGE	1E29
DFLAG$	006D	DIRBUF$	2300	DIS_DO_RAM	0846
DODATA$	0B94	DODCB$	0210	DO_CONTROL	0C44
DO_DSPCHAR	0CB8	DO_INVERT_DIS	0C8C	DO_INVERT_ENA	0C89
DO_INVERT_OFF	0C9B	DO_MASK	0000	DO_RET	0BCB
DO_RET1	0BCC	DO_SCROLL	0CCE	DO_TABS	0BEA
DSKTYP$	04C0	DSPADR	00A6	DSPASC@	201B
DTPMT$	04C2	DVREND$	0FF4	DVRHI$	0206
ED_TAB	2150	EFLAG$	006E	ENADIS_DO_RAM	0817
ENT1	1E6F	ENT2	1E74	ENT3	1E85
ENTER	1E59	EXTDBG$	19A4	FDDINT$	000E
FEMSK$	006F	FLGTAB$	006A	GETASC@	2031
GET_@_ROWCOL	0DAE	HERTZ$	0750	HEXIN@	21E4
HIGH$	040E	HKRES$	1A6C	HLREG	00AE
IFLAG$	0072	INBUF$	0420	INPUC@	21D5
INPUT@	21C9	INTIM$	003C	INTMSK$	003D
INTVC$	003E	IXREG	00B8	IYREG	00BA
JCLCB$	0203	JDCB$	0024	JFCB$	00C0
JLDCB$	0230	JRET$	0026	KCK@	07D6
KFLAG$	0074	KIDATA$	08FC	KIDCB$	0208
LAST	207F	LBANK$	0202	LDRV$	0023
LFLAG$	0075	LNKFCB@	1566	LOC1	1EAA
LOC2	1EB7	LOC3	1EBB	LOCATE	1E8C
LOW$	001E	LSVC$	000D	MAXCOR$	2400
MAXDAY$	0401	MINCOR$	3000	MODOUT$	0076
MONTBL$	04DC	NEX1	1E4C	NEXT	1E38
NFLAG$	0077	NXTADR	00A3	NXTBYT	00A5
OPREG$	0078	OPREG_SV_AREA	086E	OPREG_SV_PTR	0835
OP_TAB	211F	ORARET@	14DC	OSRLS$	003B
OSVER$	0085	OVRLY$	0069	PAKNAM$	0410
PAUSE@	0382	PCREG	00BE	PCSAVE$	07AF
PDRV$	001B	PHIGH$	001C	PRDCB$	0218
PRI1	1FBA	PRI10	2074	PRI2	1FD2
PRI3	2015	PRI4	2017	PRI5	2047
PRI6	204D	PRI7	2056	PRI8	2058
PRI9	206E	PRINT	1FB8	PUTA@DE	0DCD
PUT_@	0DCA	PUT_@_ROWCOL	0DC6	REGSAV	00BD
RFLAG$	007B	ROWCOL_2_ADDR	0DD0	RST38@	1BFF
RSTOR$	04C4	RWRIT@	13A2	S1DCB$	0238
SAVONE	00A0	SAVTWO	00A1	SBUFF$	1D00
SET@EXEC	1A79	SET_SCROLL	0CF3	SFCB$	008C
SFLAG$	007C	SIDCB$	0220	SODCB$	0228
SPACE4$	2142	SPREG	00BC	STACK$	0380
START$	0000	SVCRET$	000B	SVCTAB$	0100
SYS9	1E00	SYSERR$	1B13	TCB$	004E
TFLAG$	007D	TIME$	002D	TIMER$	002C

TIMSL$	ØØ2B	TIMTSK$	Ø713	TMPMT$	Ø4C3
TRACE_INT	Ø7B1	TYP1	1EE6	TYP2	1EEB
TYP5	1FØ1	TYPE	1ED3	TYPHK$	ØA8F
TYPTSK$	ØB26	USTOR$	ØØ13	VER1	1F2Ø
VER2	1F37	VER3	1F46	VER4	1F4F
VER5	1F5A	VERIFY	1FØ7	VFLAG$	ØØ7F
WOR1	1F85	WOR2	1F96	WOR3	1F99
WOR4	1F9F	WORD	1F62	WR1HEX@	2211
WR2HEX@	2215	WRINT$	ØØ8Ø	WRSPA@	2231
XY_TAB	2157	ZERO$	Ø4Ø1	ZEROA@	13AØ

1EØØ is the transfer address
ØØØØØ Total errors

NOTES:

SYS1Ø/SYS will kill a file or a device. It contains code for the SVC @REMOV.

```
                00100 ;SYS1Ø/ASM - LS-DOS 6.2
0000            00110         TITLE   <SYS1Ø - LS-DOS 6.2>
                00120 ;
000D            00130 CR      EQU     13
                00140 ;
                00150 *LIST   OFF                     ;Get SYSØ/EQU
                00170 *LIST   ON
0000            00180 *GET    COPYCOM:3               ;Copyright message
                03010 ; COPYCOM - File for Copyright COMment block
                03020 ;
0000            03030         COM     '<*(C) 1982,83,84 by LSI*>'
                03040 ;
                00190 ;
1E00            00200         ORG     1E00H
                00210 ;
1E00 E670       00220 SYS1Ø   AND     70H             ;Strip bit 7
1E02 C8         00230         RET     Z               ;Back on zero entry
1E03 FE10       00240         CP      10H             ;Remove all for now
1E05 C0         00250         RET     NZ              ;Ret if any other entry
1E06 1A         00260         LD      A,(DE)          ;Test device/file
1E07 CB7F       00270         BIT     7,A             ;File open or device?
1E09 2860       00280         JR      Z,CLOSDCB       ;Jump if device
1E0B CD6815     00290         CALL    CKOPEN@         ;Test for open file
1E0E DD7E01     00300         LD      A,(IX+1)        ;  & link the FCB to IX
1E11 E607       00310         AND     7               ;Test for remove access
1E13 FE02       00320         CP      2
1E15 3804       00330         JR      C,REMOV1        ;Jump if access granted
1E17 3E25       00340         LD      A,25H           ;  else init errcod
1E19 B7         00350         OR      A
1E1A C9         00360         RET
1E1B DD4E06     00370 REMOV1  LD      C,(IX+6)        ;P/u drive #
1E1E DD4607     00380         LD      B,(IX+7)        ;P/u DEC
1E21 CD7418     00390         CALL    @GATRD          ;Read GAT => DIRBUF$
1E24 CCBB18     00400 REMOV2  CALL    Z,@DIRRD        ;Read dir for this DEC
1E27 C0         00410         RET     NZ              ;Ret if read error
1E28 3E16       00420         LD      A,22            ;Point to 1st extent
1E2A 85         00430         ADD     A,L
1E2B 6F         00440         LD      L,A
1E2C 5E         00450 REMOV3  LD      E,(HL)          ;P/u relative cylinder
1E2D 2C         00460         INC     L
1E2E 56         00470         LD      D,(HL)          ;P/u granule allocation
1E2F ED53541E   00480         LD      (EXTINFO+1),DE  ;Modify later instruction
1E33 7B         00490         LD      A,E             ;Ck if extent in use
1E34 FEFE       00500         CP      0FEH
1E36 3006       00510         JR      NC,FIXDIR       ;Jump if not used
1E38 2C         00520         INC     L
1E39 CD891E     00530         CALL    RMVEXT          ;Deallocate ext from GAT
1E3C 18EE       00540         JR      REMOV3          ;Loop to next extent
                00550 ;
                00560 ;       Deallocated last extent; clean up directory
                00570 ;
1E3E 7D         00580 FIXDIR  LD      A,L             ;Point to 1st byte
1E3F E6E0       00590         AND     0E0H            ;  of DIR entry
1E41 6F         00600         LD      L,A
1E42 CBA6       00610         RES     4,(HL)          ;Show dir entry spare
1E44 CD0318     00620         CALL    @DIRWR          ;Write the dir record
1E47 CC9718     00630         CALL    Z,@HITRD        ;Grab HIT => SBUFF$
1E4A 261D       00640         LD      H,SBUFF$<-8     ;Point to HIT entry
1E4C 68         00650         LD      L,B             ;  & zero out DEC pos
1E4D 3600       00660         LD      (HL),0
1E4F CC9818     00670         CALL    Z,@HITWR        ;Write HIT back to disk
```

```
1E52 CØ       ØØ68Ø          RET     NZ              ;Ret if read/write error
1E53 11ØØØØ   ØØ69Ø EXTINFO  LD      DE,Ø            ;P/u last extent info
              ØØ7ØØ ;
              ØØ71Ø ;        If extended directory record in use,
              ØØ72Ø ;        D -> DEC of FXDE record
              ØØ73Ø ;        E -> FE if FXDE, FF if extent unused
              ØØ74Ø ;
1E56 42       ØØ75Ø          LD      B,D             ;Ck for FXDE in use
1E57 7B       ØØ76Ø          LD      A,E
1E58 FEFE     ØØ77Ø          CP      ØFEH            ;X'FE' => FXDE in use
1E5A 28C8     ØØ78Ø          JR      Z,REMOV2        ;Jump if FXDE in use
1E5C CD7518   ØØ79Ø          CALL    @GATWR          ;  else write the GAT
1E5F CØ       ØØ8ØØ          RET     NZ              ;Ret if write error
1E6Ø DDE5     ØØ81Ø          PUSH    IX              ;Transfer FCB address
1E62 E1       ØØ82Ø          POP     HL              ;  to HL & zero out FCB
1E63 Ø62Ø     ØØ83Ø          LD      B,32            ;Init for 32 byte field
1E65 AF       ØØ84Ø          XOR     A               ;Zero the accumulator
1E66 77       ØØ85Ø ZERLP1   LD      (HL),A          ;Go for it!
1E67 23       ØØ86Ø          INC     HL
1E68 1ØFC     ØØ87Ø          DJNZ    ZERLP1
1E6A C9       ØØ88Ø          RET
              ØØ89Ø ;
              ØØ9ØØ ;        REMOVE will only close a logical device
              ØØ91Ø ;
1E6B FE1Ø     ØØ92Ø CLOSDCB  CP      1ØH             ;Is this an open DCB?
1E6D 3E26     ØØ93Ø          LD      A,38            ;Init "file not open
1E6F CØ       ØØ94Ø          RET     NZ
1E7Ø CD6615   ØØ95Ø          CALL    LNKFCB@         ;Link to DCB (DE->IX)
1E73 DD4EØ6   ØØ96Ø          LD      C,(IX+6)        ;Get device name
1E76 DD46Ø7   ØØ97Ø          LD      B,(IX+7)
1E79 DD36ØØ2A ØØ98Ø          LD      (IX+Ø),'*'      ;Stuff device indicator
1E7D DD71Ø1   ØØ99Ø          LD      (IX+1),C        ;Stuff 1st char of name
1E8Ø DD7Ø2    Ø1ØØØ          LD      (IX+2),B        ;Stuff 2nd char of name
1E83 DD36Ø3Ø3 Ø1Ø1Ø          LD      (IX+3),3        ;Terminate with ETX
1E87 AF       Ø1Ø2Ø          XOR     A
1E88 C9       Ø1Ø3Ø          RET
              Ø1Ø4Ø ;
              Ø1Ø5Ø ;        Deallocate an extent
              Ø1Ø6Ø ;
1E89 E5       Ø1Ø7Ø RMVEXT   PUSH    HL
1E8A C5       Ø1Ø8Ø          PUSH    BC
1E8B 3EØ8     Ø1Ø9Ø          LD      A,8             ;P/u the # of grans per
1E8D CD2B1A   Ø11ØØ          CALL    @DCTBYT         ;  cylinder into reg A
1E9Ø Ø7       Ø111Ø          RLCA                    ;Shift into bits Ø-2
1E91 Ø7       Ø112Ø          RLCA
1E92 Ø7       Ø113Ø          RLCA
1E93 E6Ø7     Ø114Ø          AND     7               ;Remove all else
1E95 3C       Ø115Ø          INC     A               ;Adjust for zero offset
              Ø116Ø ;
              Ø117Ø ;        Ck for 2-sided operation
              Ø118Ø ;
1E96 6F       Ø119Ø          LD      L,A             ;Save current grans/cyl
1E97 3EØ4     Ø12ØØ          LD      A,4
1E99 CD2B1A   Ø121Ø          CALL    @DCTBYT         ;Get 2-sided flag
1E9C CB6F     Ø122Ø          BIT     5,A             ;Test 2-sided
1E9E 7D       Ø123Ø          LD      A,L             ;Xfer value back
1E9F 28Ø1     Ø124Ø          JR      Z,$+3           ;Bypass if 1-sided
1EA1 87       Ø125Ø          ADD     A,A             ;  else multiply by 2
1EA2 32BA1E   Ø126Ø          LD      (GRNSCYL+1),A   ;Modify later instruction
1EA5 6B       Ø127Ø          LD      L,E             ;Relative cylinder -> L
1EA6 2623     Ø128Ø          LD      H,DIRBUF$<-8    ;Point to GAT byte
```

```
1EA8 7A       Ø129Ø          LD      A,D             ;Rel gran & # of grans
1EA9 E61F     Ø13ØØ          AND     1FH             ;Get # of grans
1EAB 4F       Ø131Ø          LD      C,A             ;  into reg C & adjust
1EAC ØC       Ø132Ø          INC     C               ;  for zero offset
1EAD AA       Ø133Ø          XOR     D               ;Get rel gran & shift
1EAE Ø7       Ø134Ø          RLCA                    ;  into bits Ø-2
1EAF Ø7       Ø135Ø          RLCA
1EBØ Ø7       Ø136Ø          RLCA
1EB1 F5       Ø137Ø RMVEX1   PUSH    AF              ;Save rel starting gran
1EB2 46       Ø138Ø          LD      B,(HL)          ;P/u allocation byte
1EB3 CDC51E   Ø139Ø          CALL    RMVGRN          ;Turn off bit for a gran
1EB6 7Ø       Ø14ØØ          LD      (HL),B          ;Update GAT byte
1EB7 F1       Ø141Ø          POP     AF              ;Recover starting gran
1EB8 3C       Ø142Ø          INC     A               ;Bump up
1EB9 FEØØ     Ø143Ø GRNSCYL  CP      Ø               ;Ck with grans per cyl
1EBB 2ØØ2     Ø144Ø          JR      NZ,DECGRNS      ;Go if still on this cyl
1EBD AF       Ø145Ø          XOR     A               ;  else zero gran counter
1EBE 2C       Ø146Ø          INC     L               ;Bump to next cyl in GAT
1EBF ØD       Ø147Ø DECGRNS  DEC     C               ;Decrement # of grans
1ECØ 2ØEF     Ø148Ø          JR      NZ,RMVEX1       ;Go if more to deallocate
1EC2 C1       Ø149Ø          POP     BC              ;  else recover regs
1EC3 E1       Ø15ØØ          POP     HL              ;  & go home
1EC4 C9       Ø151Ø          RET
              Ø152Ø ;
              Ø153Ø ;        Remove a bit to deallocate & free up a gran
              Ø154Ø ;
1EC5 E6Ø7     Ø155Ø RMVGRN   AND     7               ;Max 8-grans per cyl
1EC7 Ø7       Ø156Ø          RLCA                    ;Shift to create RES
1EC8 Ø7       Ø157Ø          RLCA
1EC9 Ø7       Ø158Ø          RLCA
1ECA F68Ø     Ø159Ø          OR      8ØH             ;Merge rest of RES code
1ECC 32DØ1E   Ø16ØØ          LD      (RMVGRN1+1),A   ;Stuff into the inst
1ECF CB8Ø     Ø161Ø RMVGRN1  RES     Ø,B             ;Reset proper bit
1ED1 C9       Ø162Ø          RET
              Ø163Ø ;
1ED2          Ø164Ø LAST     EQU     $
              Ø165Ø          IFGT    $,DIRBUF$
              Ø166Ø          ERR     'Module too big'
              Ø167Ø          ENDIF
23FE          Ø168Ø          ORG     MAXCOR$-2
23FE D2ØØ     Ø169Ø          DW      LAST-SYS1Ø      ;Overlay size
              Ø17ØØ ;
1EØØ          Ø171Ø          END     SYS1Ø
```

$A1	Ø3B7	$A2	Ø3B8	$A3	Ø3B9		
$CKEOF	147Ø	@$SYS	Ø8FØ	@@1	ØØØØ		
@@2	ØØØØ	@@3	ØØØØ	@@4	ØØØØ		
@ABORT	1BØ8	@ADTSK	1CDA	@BANK	Ø877		
@BKSP	1486	@BREAK	196F	@BYTEIO	13ØØ		
@CHNIO	Ø689	@CKBRKC	Ø553	@CKDRV	1993		
@CKEOF	158F	@CKTSK	1CF5	@CLOSE	1999		
@CLS	Ø545	@CMNDI	197E	@CMNDR	197B		
@CTL	Ø623	@DATE	Ø7A8	@DBGHK	199F		
@DCINIT	19CØ	@DCRES	19C4	@DCSTAT	19B5		
@DCTBYT	1A2B	@DEBUG	19AØ	@DECHEX	Ø3E1		
@DIRCYL	18F7	@DIRRD	18BB	@DIRWR	18Ø3		
@DIV16	Ø6E3	@DIV8	1927	@DODIR	19AF		
@DOKEY	19A9	@DSP	Ø642	@DSPLY	Ø52D		
@ERROR	1BØF	@EXIT	1BØB	@FEXT	1984		
@FLAGS	196A	@FNAME	199C	@FRENCH	ØØØØ		
@FSPEC	1981	@GATRD	1874	@GATWR	1875		
@GERMAN	ØØØØ	@GET	Ø638	@GTDCB	199Ø		
@GTDCT	1A1E	@GTMOD	19B2	@HDFMT	19E4		
@HEX16	Ø7BD	@HEX8	Ø7C2	@HEXDEC	Ø6F6		
@HIGH$	1948	@HITRD	1897	@HITWR	1898		
@HZ5Ø	ØØØØ	@ICNFG	ØØ86	@INIT	198D		
@INTL	ØØØØ	@IPL	1BF2	@JCL	Ø63Ø		
@KBD	Ø635	@KEY	Ø628	@KEYIN	Ø585		
@KITSK	ØØ89	@KLTSK	1CDØ	@LOAD	1B38		
@LOC	14B3	@LOF	14DE	@LOGER	Ø5Ø3		
@LOGOT	Ø5ØØ	@MOD2	ØØØØ	@MOD4	FFFF		
@MSG	Ø53Ø	@MUL16	Ø6C9	@MUL8	19ØA		
@NMI	ØØ66	@OPEN	198A	@OPREG	ØØ84		
@PARAM	1987	@PAUSE	Ø382	@PEOF	14A2		
@POSN	1434	@PRINT	Ø528	@PRT	Ø63D		
@PUT	Ø645	@RAMDIR	19AC	@RDHDR	19D8		
@RDSEC	19F4	@RDSSC	18D8	@RDTRK	19EØ		
@READ	1513	@REMOVE	19A6	@RENAME	1996		
@REW	149B	@RMTSK	1CD7	@RPTSK	1CEB		
@RREAD	1473	@RSLCT	19D4	@RSTØØ	ØØØØ		
@RSTØ8	ØØØ8	@RST1Ø	ØØ1Ø	@RST18	ØØ18		
@RST2Ø	ØØ2Ø	@RST28	ØØ28	@RST3Ø	ØØ3Ø		
@RST38	ØØ38	@RSTNMI	ØFE9	@RSTOR	19C8		
@RSTREG	Ø68Ø	@RUN	1B1D	@RWRIT	13AD		
@SEEK	19DØ	@SEEKSC	1421	@SKIP	143Ø		
@SLCT	19BC	@SOUND	Ø392	@STEPI	19CC		
@TIME	Ø78D	@USA	FFFF	@VDCTL	ØB99		
@VDCTL3	ØD38	@VER	156Ø	@VRSEC	19DC		
@WEOF	14EC	@WHERE	1979	@WRITE	1531		
@WRSEC	19E8	@WRSSC	19EC	@WRTRK	19FØ		
@_VDCTL	ØD42	ADDR_2_ROWCOL	ØDF1	AFLAG$	ØØ6A		
AUTO?	1FF1	BAR$	Ø2Ø1	BOOTST$	439D		
BREAK?	1C6Ø	BRKVEC$	1C88	BUR$	Ø2ØØ		
CASHK$	ØA7B	CFCB$	ØØEØ	CFGFCB$	ØØEØ		
CFLAG$	ØØ6C	CKMOD@	1A7F	CKOPEN@	1568		
CLOSDCB	1E6B	CONFIG$	2Ø3F	CORE$	Ø3ØØ		
CR	ØØØD	CRTBGN$	F8ØØ	CYL_GRN	16AE		
D@FBYT8	1A26	DATE$	ØØ33	DAYTBL$	Ø4C7		
DBGSV$	ØØAØ	DCBKL$	ØØ31	DCT$	Ø47Ø		
DCTBYT8@	1A29	DCTFLD@	1A34	DECGRNS	1EBF		
DFLAG$	ØØ6D	DIRBUF$	23ØØ	DIS_DO_RAM	Ø846		
DODATA$	ØB94	DODCB$	Ø21Ø	DO_CONTROL	ØC44		
DO_DSPCHAR	ØCB8	DO_INVERT_DIS	ØC8C	DO_INVERT_ENA	ØC89		
DO_INVERT_OFF	ØC9B	DO_MASK	ØØØØ	DO_RET	ØBCB		

```
DO_RET1          ØBCC DO_SCROLL       ØCCE DO_TABS          ØBEA
DSKTYP$          Ø4CØ DTPMT$          Ø4C2 DVREND$          ØFF4
DVRHI$           Ø2Ø6 EFLAG$          ØØ6E ENADIS_DO_RAM    Ø817
EXTDBG$          19A4 EXTINFO         1E53 FDDINT$          ØØØE
FEMSK$           ØØ6F FIXDIR          1E3E FLGTAB$          ØØ6A
GET_@_ROWCOL     ØDAE GRNSCYL         1EB9 HERTZ$           Ø75Ø
HIGH$            Ø4ØE HKRES$          1A6C IFLAG$           ØØ72
INBUF$           Ø42Ø INTIM$          ØØ3C INTMSK$          ØØ3D
INTVC$           ØØ3E JCLCB$          Ø2Ø3 JDCB$            ØØ24
JFCB$            ØØCØ JLDCB$          Ø23Ø JRET$            ØØ26
KCK@             Ø7D6 KFLAG$          ØØ74 KIDATA$          Ø8FC
KIDCB$           Ø2Ø8 LAST            1ED2 LBANK$           Ø2Ø2
LDRV$            ØØ23 LFLAG$          ØØ75 LNKFCB@          1566
LOW$             ØØ1E LSVC$           ØØØD MAXCOR$          24ØØ
MAXDAY$          Ø4Ø1 MINCOR$         3ØØØ MODOUT$          ØØ76
MONTBL$          Ø4DC NFLAG$          ØØ77 OPREG$           ØØ78
OPREG_SV_AREA    Ø86E OPREG_SV_PTR    Ø835 ORARET@          14DC
OSRLS$           ØØ3B OSVER$          ØØ85 OVRLY$           ØØ69
PAKNAM$          Ø41Ø PAUSE@          Ø382 PCSAVE$          Ø7AF
PDRV$            ØØ1B PHIGH$          ØØ1C PRDCB$           Ø218
PUTA@DE          ØDCD PUT_@           ØDCA PUT_@_ROWCOL     ØDC6
REMOV1           1E1B REMOV2          1E24 REMOV3           1E2C
RFLAG$           ØØ7B RMVEX1          1EB1 RMVEXT           1E89
RMVGRN           1EC5 RMVGRN1         1ECF ROWCOL_2_ADDR    ØDDØ
RST38@           1BFF RSTOR$          Ø4C4 RWRIT@           13A2
S1DCB$           Ø238 SBUFF$          1DØØ SET@EXEC         1A79
SET_SCROLL       ØCF3 SFCB$           ØØ8C SFLAG$           ØØ7C
SIDCB$           Ø22Ø SODCB$          Ø228 SPACE4$          2142
STACK$           Ø38Ø START$          ØØØØ SVCRET$          ØØØB
SVCTAB$          Ø1ØØ SYS1Ø           1EØØ SYSERR$          1B13
TCB$             ØØ4E TFLAG$          ØØ7D TIME$            ØØ2D
TIMER$           ØØ2C TIMSL$          ØØ2B TIMTSK$          Ø713
TMPMT$           Ø4C3 TRACE_INT       Ø7B1 TYPHK$           ØA8F
TYPTSK$          ØB26 USTOR$          ØØ13 VFLAG$           ØØ7F
WRINT$           ØØ8Ø ZERLP1          1E66 ZERO$            Ø4Ø1
ZEROA@           13AØ
1EØØ is the transfer address
ØØØØØ Total errors
```

NOTES:

NOTES:

SYS11 handles the execution of a JCL file. It automatically links and unlinks itself into the system. It is used to supply input from the JCL file in response to keyboard line requests.

```
                00100 ;SYS11/ASM - LS-DOS 6.2
                00110 *MOD
0000            00120         TITLE   <SYS11 - LS-DOS 6.2>
                00130 ;
000A            00140 LF      EQU     10
000D            00150 CR      EQU     13
                00160 *LIST   OFF                     ;Get SYS0/EQU
                00180 *LIST   ON
0000            00190 *GET    COPYCOM:3               ;Copyright message
                03010 ; COPYCOM - File for Copyright COMment block
                03020 ;
0000            03030         COM     '<*(C) 1982,83,84 by LSI*>'
                03040 ;
                00200 ;
1E00            00210         ORG     1E00H
                00220 ;
1E00 E670       00230 SYS11   AND     70H
1E02 C8         00240         RET     Z               ;Back on zero entry
1E03 E5         00250         PUSH    HL
1E04 217400     00260         LD      HL,KFLAG$       ;Reset the <ENTER>
1E07 CB96       00270         RES     2,(HL)          ;  bit every time
1E09 E1         00280         POP     HL
1E0A FE20       00290         CP      20H             ;New @EXIT?
1E0C 2847       00300         JR      Z,NEWEXIT
1E0E FE40       00310         CP      40H             ;New keyboard request
1E10 CAD61E     00320         JP      Z,KEYREQ        ;  after input of a line?
1E13 FE50       00330         CP      50H             ;//INPUT followup
1E15 CAB420     00340         JP      Z,GETKEY
1E18 FE10       00350         CP      10H             ;Initial entry to DO?
1E1A C0         00360         RET     NZ
                00370 ;
                00380 ;       <DO> initialization of sysres hooks
                00390 ;
1E1B F3         00400         DI                      ;Clock off for now
1E1C 217400     00410         LD      HL,KFLAG$       ;Reset break bit only on
1E1F CB86       00420         RES     0,(HL)          ;  initial entry
1E21 217C00     00430         LD      HL,SFLAG$
1E24 CB6E       00440         BIT     5,(HL)          ;If DO already in effect
1E26 CBEE       00450         SET     5,(HL)          ;  don't rehook
1E28 2005       00460         JR      NZ,IPLDO1
1E2A 3EAD       00470         LD      A,0ADH          ;Change @EXIT,@ABORT to use
1E2C 320C1B     00480         LD      (@EXIT+1),A     ;  SYS11 rather than SYS1
1E2F 318003     00490 IPLDO1  LD      SP,STACK$
1E32 FB         00500         EI                      ;Clock back on
1E33 11C000     00510         LD      DE,JFCB$        ;At end of SYSTEM/JCL?
1E36 CD8F15     00520         CALL    @CKEOF
1E39 C20F1B     00530         JP      NZ,@ERROR
1E3C 114F1E     00540         LD      DE,IPLDO2       ;Init JCLCB$
1E3F ED530402   00550         LD      (JCLCB$+1),DE
1E43 CD771E     00560         CALL    GETLINE         ;Get a line from the file
1E46 11A919     00570         LD      DE,@DOKEY       ;Change vector to SYS11,
1E49 ED530402   00580         LD      (JCLCB$+1),DE   ;  entry 4
1E4D 1825       00590         JR      $?1             ;Go interpret it
1E4F 11C000     00600 IPLDO2  LD      DE,JFCB$        ;JCLCB$ input routine
1E52 C33806     00610         JP      @GET
                00620 ;
                00630 ;       New @EXIT processing
                00640 ;
1E55 318003     00650 NEWEXIT LD      SP,STACK$       ;Reset the stack
1E58 FB         00660         EI
1E59 7C         00670         LD      A,H             ;Ck for error return
```

```
1E5A B5       00680          OR      L
1E5B 2023     00690          JR      NZ,ABORT
1E5D 217C00   00700          LD      HL,SFLAG$
1E60 CB66     00710          BIT     4,(HL)          ;BREAK key disabled?
1E62 2005     00720          JR      NZ,NEWEX1
1E64 CD5305   00730          CALL    @CKBRKC         ;Check on <BREAK>
1E67 2017     00740          JR      NZ,ABORT
1E69 11C000   00750 NEWEX1   LD      DE,JFCB$        ;Exit if end of JCL
1E6C CD8F15   00760          CALL    @CKEOF
1E6F 2041     00770          JR      NZ,EXIT
1E71 CD771E   00780          CALL    GETLINE         ;Grab a JCL line
1E74 C37E19   00790 $?1      JP      @CMNDI
1E77 212004   00800 GETLINE  LD      HL,INBUF$       ;Pt to line buffer
1E7A 01004F   00810          LD      BC,79<8         ;Max 79 chars
1E7D C38505   00820          JP      @KEYIN
              00830 ;
              00840 ;        New ABORT processor
              00850 ;
1E80 216221   00860 ABORT    LD      HL,ABORT$       ;"Job abort...
1E83 11081B   00870          LD      DE,@ABORT
1E86 1830     00880          JR      EXIT1
              00890 ;
              00900 ;        Scan for ENTER or BREAK
              00910 ;
1E88 3A7C00   00920 KSCN     LD      A,(SFLAG$)      ;Only test BREAK if
1E8B CB67     00930          BIT     4,A             ;  BREAK key enabled
1E8D 3A7400   00940          LD      A,(KFLAG$)
1E90 2004     00950          JR      NZ,KSCN1
1E92 CB47     00960          BIT     0,A             ;BREAK detected?
1E94 20EA     00970          JR      NZ,ABORT
1E96 CB57     00980 KSCN1    BIT     2,A             ;Test <ENTER>
1E98 C8       00990          RET     Z               ;Back if not
1E99 CD3506   01000 KSCN2    CALL    @KBD            ;Clear the type ahead
1E9C 28FB     01010          JR      Z,KSCN2
1E9E 217400   01020          LD      HL,KFLAG$       ;Reset the ENTER bit
1EA1 CB96     01030          RES     2,(HL)
1EA3 C5       01040          PUSH    BC
1EA4 060B     01050          LD      B,3000<-8
1EA6 CD8203   01060          CALL    @PAUSE
1EA9 C1       01070          POP     BC
1EAA 7E       01080          LD      A,(HL)          ;Don't return until clear
1EAB E604     01090          AND     4
1EAD EE04     01100          XOR     4
1EAF 28E8     01110          JR      Z,KSCN2
1EB1 C9       01120          RET
              01130 ;
              01140 ;        Continuation of EXIT processing
              01150 ;
1EB2 216E21   01160 EXIT     LD      HL,JOBDUN$      ;"Job done...
1EB5 110B1B   01170          LD      DE,@EXIT
1EB8 D5       01180 EXIT1    PUSH    DE
1EB9 CD0005   01190          CALL    @LOGOT          ;Log & fall thru
              01200 ;
              01210 ;        Turn off the DO processor
              01220 ;
1EBC          01230 DOOFF    EQU     $
1EBC F3       01240          DI
1EBD 217C00   01250          LD      HL,SFLAG$       ;Reset <DO> flag
1EC0 CBAE     01260          RES     5,(HL)
1EC2 AF       01270          XOR     A
1EC3 32C000   01280          LD      (JFCB$),A       ;Show fcb is closed
```

```
1EC6 67       Ø129Ø          LD      H,A             ;Set = Ø for @EXIT
1EC7 6F       Ø13ØØ          LD      L,A
1EC8 11Ø8Ø2   Ø131Ø          LD      DE,KIDCB$       ;Clear any type ahead
1ECB 3EØ3     Ø132Ø          LD      A,3
1ECD CD23Ø6   Ø133Ø          CALL    @CTL            ;  buffer (no streaming)
1EDØ 3E93     Ø134Ø          LD      A,93H           ;Restore @EXIT SVC
1ED2 32ØC1B   Ø135Ø          LD      (@EXIT+1),A     ;  back to SYS1
1ED5 C9       Ø136Ø          RET
              Ø137Ø ;
              Ø138Ø ;        Keyboard request processor
              Ø139Ø ;
1ED6 21ØAØØ   Ø14ØØ KEYREQ   LD      HL,1Ø           ;Back stack up 5 words
1ED9 39       Ø141Ø          ADD     HL,SP           ;SYSØ,RET,DE,HL,IX,BC
1EDA 4E       Ø142Ø          LD      C,(HL)          ;Get contents of BC
1EDB 23       Ø143Ø          INC     HL              ;  prior to keyboard
1EDC 46       Ø144Ø          LD      B,(HL)          ;  request & DRIVER save
              Ø145Ø ;
              Ø146Ø ;        @KEYIN is requesting an entire line
              Ø147Ø ;
1EDD 11CØØØ   Ø148Ø KEYLINE  LD      DE,JFCB$        ;Ck on end of JCL file
1EEØ C5       Ø149Ø          PUSH    BC
1EE1 CD8F15   Ø15ØØ          CALL    @CKEOF
1EE4 C1       Ø151Ø          POP     BC
1EE5 2ØCB     Ø152Ø          JR      NZ,EXIT
1EE7 78       Ø153Ø          LD      A,B             ;Do we need to re-read
1EE8 B9       Ø154Ø          CP      C               ;  the JCL sector?
1EE9 C238Ø6   Ø155Ø          JP      NZ,@GET
1EEC CD7314   Ø156Ø          CALL    @RREAD          ;Get the sector back
1EEF C2ØF1B   Ø157Ø          JP      NZ,@ERROR
1EF2 CD38Ø6   Ø158Ø          CALL    @GET
1EF5 B7       Ø159Ø          OR      A
1EF6 28BA     Ø16ØØ          JR      Z,EXIT
1EF8 FE2F     Ø161Ø          CP      '/'             ;Is this line execution
1EFA 28Ø2     Ø162Ø          JR      Z,GOTSLSH       ;  JCL code to parse?
1EFC BF       Ø163Ø          CP      A               ;Set Z-flag
1EFD C9       Ø164Ø          RET
              Ø165Ø ;
              Ø166Ø ;        Found an execution code line
              Ø167Ø ;
1EFE C5       Ø168Ø GOTSLSH  PUSH    BC
1EFF D5       Ø169Ø          PUSH    DE
1FØØ Ø64F     Ø17ØØ          LD      B,79            ;Only 79 char line
1FØ2 212ØØ4   Ø171Ø          LD      HL,INBUF$       ;Get rest of line
1FØ5 E5       Ø172Ø          PUSH    HL              ;  into JCL buffer
1FØ6 77       Ø173Ø GOTSL1   LD      (HL),A          ;Compare for CR as end
1FØ7 23       Ø174Ø          INC     HL              ;  of line
1FØ8 FEØD     Ø175Ø          CP      CR
1FØA 28Ø7     Ø176Ø          JR      Z,GOTSL2
1FØC CD38Ø6   Ø177Ø          CALL    @GET            ;Get a character
1FØF 1ØF5     Ø178Ø          DJNZ    GOTSL1          ;  up to 79 max
1F11 183F     Ø179Ø          JR      BADJCL          ;Line too long
1F13 E1       Ø18ØØ GOTSL2   POP     HL              ;Rcvr pointer to buf
1F14 E5       Ø181Ø          PUSH    HL
1F15 23       Ø182Ø          INC     HL              ;Pt to 2nd char
1F16 7E       Ø183Ø          LD      A,(HL)
1F17 FE2F     Ø184Ø          CP      '/'             ;Found a //?
1F19 2Ø32     Ø185Ø          JR      NZ,REKEY2
1F1B 23       Ø186Ø          INC     HL              ;Ck on ///
1F1C 96       Ø187Ø          SUB     (HL)
1F1D CAC11F   Ø188Ø          JP      Z,KEYIN6        ;Jump if ///
1F2Ø D6F6     Ø189Ø          SUB     ØF6H
```

```
1F22 D2BD1F   01900          JP       NC,KEYIN5       ;Jump if 3rd char is 0-9
1F25 E3       01910          EX       (SP),HL         ;P/u start of command
1F26 CD0305   01920          CALL     @LOGER          ;  line & log it
1F29 E3       01930          EX       (SP),HL
1F2A 7E       01940 GOTSL3   LD       A,(HL)          ;Was char ENTER?
1F2B FE0D     01950          CP       CR
1F2D 281E     01960          JR       Z,REKEY2
1F2F FE20     01970          CP       ' '             ;Ignore leading spaces
1F31 23       01980          INC      HL
1F32 28F6     01990          JR       Z,GOTSL3
1F34 2B       02000          DEC      HL
1F35 115321   02010          LD       DE,LILBUF       ;Put possible parm -> buf
1F38 0605     02020          LD       B,5             ;Max length of parm
1F3A CDC820   02030          CALL     PARSER          ;Parse parm
1F3D 200E     02040          JR       NZ,REKEY2
1F3F 115321   02050          LD       DE,LILBUF
1F42 017721   02060          LD       BC,PARMTBL      ;Is the parm a macro?
1F45 CD1121   02070          CALL     FNDPARM
1F48 2003     02080          JR       NZ,REKEY2       ;Bypass if not in tbl
1F4A D5       02090          PUSH     DE              ;Stack routine's entry
1F4B C9       02100          RET                      ;  & go to it
1F4C C1       02110 REKEY1   POP      BC
1F4D E1       02120 REKEY2   POP      HL
1F4E D1       02130          POP      DE
1F4F C1       02140          POP      BC
1F50 188B     02150          JR       KEYLINE
1F52 215921   02160 BADJCL   LD       HL,BADJCL$      ;"invalid JCL...
1F55 C3831E   02170          JP       ABORT+3
              02180 ;
              02190 ;        Process  //STOP
              02200 ;
1F58 CDBC1E   02210 STOP     CALL     DOOFF           ;Turn off DO proc
1F5B E1       02220          POP      HL
1F5C D1       02230          POP      DE
1F5D C1       02240          POP      BC
1F5E FB       02250          EI
1F5F C32806   02260          JP       @KEY            ;Go back to keyboard
              02270 ;
              02280 ;        Process  //DELAY
              02290 ;
1F62 E3       02300 DELAY    EX       (SP),HL         ;Pt to //delay  line
1F63 CD2D05   02310          CALL     @DSPLY          ;  and display it
1F66 E3       02320          EX       (SP),HL
1F67 CDE103   02330          CALL     @DECHEX         ;Cvrt entry to binary
1F6A 41       02340          LD       B,C             ;Set count
1F6B CD1C20   02350 DELAY1   CALL     SILEN1          ;Delay a bit
1F6E 10FB     02360          DJNZ     DELAY1
1F70 18DB     02370          JR       REKEY2
              02380 ;
              02390 ;        Process  //PAUSE
              02400 ;
1F72 E1       02410 PAUSE    POP      HL              ;Display "pause..
1F73 E5       02420          PUSH     HL
1F74 CD2D05   02430          CALL     @DSPLY
1F77 CD881E   02440 PAUSE1   CALL     KSCN            ;Loop for BREAK or ENTER
1F7A 28FB     02450          JR       Z,PAUSE1
1F7C 18CF     02460          JR       REKEY2
              02470 ;
              02480 ;        Process  //KEYIN
              02490 ;
1F7E E1       02500 KEYIN    POP      HL              ;Rcvr pointer to "KEYIN
```

```
1F7F E5       Ø251Ø          PUSH    HL
1F8Ø 7E       Ø252Ø KEYIN1   LD      A,(HL)          ;Display JCL command line
1F81 23       Ø253Ø          INC     HL
1F82 FEØD     Ø254Ø          CP      CR
1F84 28Ø5     Ø255Ø          JR      Z,KEYIN2
1F86 CD42Ø6   Ø256Ø          CALL    @DSP
1F89 18F5     Ø257Ø          JR      KEYIN1
1F8B CD28Ø6   Ø258Ø KEYIN2   CALL    @KEY            ;Get & display the char
1F8E CD42Ø6   Ø259Ø          CALL    @DSP
1F91 32BE1F   Ø26ØØ          LD      (KEYIN5+1),A    ;Stuff for compare
1F94 3EØD     Ø261Ø          LD      A,CR
1F96 CD42Ø6   Ø262Ø          CALL    @DSP            ;Write new line
1F99 E1       Ø263Ø KEYIN3   POP     HL
1F9A E5       Ø264Ø          PUSH    HL
1F9B 11CØØØ   Ø265Ø          LD      DE,JFCB$        ;Ck for end of JCL
1F9E CD8F15   Ø266Ø          CALL    @CKEOF
1FA1 C2B21E   Ø267Ø          JP      NZ,EXIT
1FA4 CD38Ø6   Ø268Ø KEYIN4   CALL    @GET            ;Xfer a line of JCL
1FA7 77       Ø269Ø          LD      (HL),A          ;  to buffer
1FA8 23       Ø27ØØ          INC     HL
1FA9 FEØD     Ø271Ø          CP      CR
1FAB 2ØF7     Ø272Ø          JR      NZ,KEYIN4
1FAD E1       Ø273Ø          POP     HL
1FAE E5       Ø274Ø          PUSH    HL
1FAF 7E       Ø275Ø          LD      A,(HL)          ;Look for // to find
1FBØ FE2F     Ø276Ø          CP      '/'             ;Start of procedure block
1FB2 2ØE5     Ø277Ø          JR      NZ,KEYIN3
1FB4 23       Ø278Ø          INC     HL
1FB5 BE       Ø279Ø          CP      (HL)            ;//?
1FB6 2ØE1     Ø28ØØ          JR      NZ,KEYIN3
1FB8 23       Ø281Ø          INC     HL              ;Point to proc label
1FB9 96       Ø282Ø          SUB     (HL)            ;Is label a '/' noting
1FBA 28Ø5     Ø283Ø          JR      Z,KEYIN6        ;  exec phase cond's end?
1FBC 7E       Ø284Ø          LD      A,(HL)          ;Nope, get proc label
1FBD FEØØ     Ø285Ø KEYIN5   CP      Ø               ;Same as key entry?
1FBF 2ØD8     Ø286Ø          JR      NZ,KEYIN3       ;No match? check next one
1FC1 32BE1F   Ø287Ø KEYIN6   LD      (KEYIN5+1),A    ;Stuff Ø if ///
1FC4 E1       Ø288Ø          POP     HL
1FC5 E5       Ø289Ø          PUSH    HL
1FC6 CDØ3Ø5   Ø29ØØ          CALL    @LOGER          ;Log the command
1FC9 1882     Ø291Ø          JR      REKEY2
              Ø292Ø ;
              Ø293Ø ;        Process //ALERT
              Ø294Ø ;
1FCB AF       Ø295Ø ALERT    XOR     A
1FCC 32FB1F   Ø296Ø          LD      (ALERT4+1),A    ;Start with clean flag
1FCF 7E       Ø297Ø ALERT1   LD      A,(HL)          ;Ignore spaces
1FDØ 23       Ø298Ø          INC     HL
1FD1 FE2Ø     Ø299Ø          CP      ' '
1FD3 28FA     Ø3ØØØ          JR      Z,ALERT1
1FD5 FE2C     Ø3Ø1Ø          CP      ','             ;Comma separator?
1FD7 28F6     Ø3Ø2Ø          JR      Z,ALERT1
1FD9 FEØD     Ø3Ø3Ø          CP      CR              ;End of line?
1FDB CA4D1F   Ø3Ø4Ø          JP      Z,REKEY2
1FDE FE29     Ø3Ø5Ø          CP      ')'             ;Closing paren?
1FEØ 28Ø9     Ø3Ø6Ø          JR      Z,ALERT2
1FE2 FE28     Ø3Ø7Ø          CP      '('             ;Start of parms?
1FE4 2ØØF     Ø3Ø8Ø          JR      NZ,ALERT3       ;If none of the above...
1FE6 22EC1F   Ø3Ø9Ø          LD      (ALERT2+1),HL   ;Save ptr to parm start
1FE9 18E4     Ø31ØØ          JR      ALERT1
              Ø311Ø ;
```

```
                    Ø312Ø ;       Check here when closing parm received
                    Ø313Ø ;
1FEB 21ØØØØ         Ø314Ø ALERT2  LD      HL,Ø            ;P/u ptr to '(' if there
1FEE 7C             Ø315Ø         LD      A,H             ;If the //ALERT1 started
1FEF B5             Ø316Ø         OR      L               ;  with a '(', then
1FFØ 2ØDD           Ø317Ø         JR      NZ,ALERT1       ;  repeat the parm
1FF2 C3521F         Ø318Ø         JP      BADJCL          ;  parsing else exit
                    Ø319Ø ;
                    Ø32ØØ ;       Assumed integer parm found
                    Ø321Ø ;
1FF5 2B             Ø322Ø ALERT3  DEC     HL              ;Backup pointer
1FF6 CDE1Ø3         Ø323Ø         CALL    @DECHEX         ;Cvrt value to binary
1FF9 41             Ø324Ø         LD      B,C             ;Keep value as counter
1FFA 3EØØ           Ø325Ø ALERT4  LD      A,Ø             ;Flip flag: entries 1, 3,
1FFC EEFF           Ø326Ø         XOR     ØFFH            ;  5, ... are noise, 2,
1FFE 32FB1F         Ø327Ø         LD      (ALERT4+1),A    ;  4, 6, ... are silence
2ØØ1 4F             Ø328Ø         LD      C,A
2ØØ2 CB41           Ø329Ø         BIT     Ø,C             ;Test noise or silence
2ØØ4 C492Ø3         Ø33ØØ         CALL    NZ,@SOUND       ;Call for sound out
2ØØ7 CB41           Ø331Ø         BIT     Ø,C             ;  then test again
2ØØ9 CC142Ø         Ø332Ø         CALL    Z,SILENCE       ;Silence is golden
2ØØC CD881E         Ø333Ø         CALL    KSCN            ;Ck BREAK or ENTER
2ØØF C24D1F         Ø334Ø         JP      NZ,REKEY2       ;Go on enter
2Ø12 18BB           Ø335Ø         JR      ALERT1          ;Loop if not
                    Ø336Ø ;
                    Ø337Ø ;       Silence routine
                    Ø338Ø ;
2Ø14 BØ             Ø339Ø SILENCE OR      B               ;A was zero
2Ø15 C8             Ø34ØØ         RET     Z
2Ø16 CD1C2Ø         Ø341Ø         CALL    SILEN1          ;Delay a bit
2Ø19 1ØF9           Ø342Ø         DJNZ    SILENCE         ;  for duration
2Ø1B C9             Ø343Ø         RET
2Ø1C C5             Ø344Ø SILEN1  PUSH    BC              ;Delay for Ø.1 sec
2Ø1D Ø19B19         Ø345Ø         LD      BC,6555
2Ø2Ø CD82Ø3         Ø346Ø         CALL    @PAUSE
2Ø23 C1             Ø347Ø         POP     BC
2Ø24 C9             Ø348Ø         RET
                    Ø349Ø ;
                    Ø35ØØ ;       Process //FLASH
                    Ø351Ø ;
2Ø25 CDE1Ø3         Ø352Ø FLASH   CALL    @DECHEX
2Ø28 41             Ø353Ø         LD      B,C             ;P/u the flash count
2Ø29 E1             Ø354Ø         POP     HL
2Ø2A E5             Ø355Ø         PUSH    HL
2Ø2B C5             Ø356Ø FLASH1  PUSH    BC
2Ø2C CD2DØ5         Ø357Ø         CALL    @DSPLY          ;Display the prompt
2Ø2F Ø1ØØ4Ø         Ø358Ø         LD      BC,4ØØØH        ;Countdown to flash msg
2Ø32 CD881E         Ø359Ø FLASH2  CALL    KSCN            ;Keep testing <ENTER>
2Ø35 C24C1F         Ø36ØØ         JP      NZ,REKEY1       ;  key during countdown
2Ø38 ØB             Ø361Ø         DEC     BC              ;BREAK would abort
2Ø39 78             Ø362Ø         LD      A,B
2Ø3A B1             Ø363Ø         OR      C
2Ø3B 2ØF5           Ø364Ø         JR      NZ,FLASH2       ;Loop until count=Ø
2Ø3D 3E1B           Ø365Ø         LD      A,27            ;Erase the message line
2Ø3F CD42Ø6         Ø366Ø         CALL    @DSP
2Ø42 3E1E           Ø367Ø         LD      A,3Ø
2Ø44 CD42Ø6         Ø368Ø         CALL    @DSP
2Ø47 CD1C2Ø         Ø369Ø         CALL    SILEN1          ;Delay while blanked
2Ø4A C1             Ø37ØØ         POP     BC
2Ø4B 1ØDE           Ø371Ø         DJNZ    FLASH1
2Ø4D C34D1F         Ø372Ø FLASH3  JP      REKEY2
```

```
                    03730 ;
                    03740 ;       Process //SLEEP and //WAIT
                    03750 ;
2050 3E             03760 SLEEP   DB      3EH             ;Make it LD A,0AFH
2051 AF             03770 WAIT    XOR     A
2052 32712D         03780         LD      (SLPWT+1),A     ;Save entry state
2055 E3             03790         EX      (SP),HL         ;Display the JCL line
2056 CD2D05         03800         CALL    @DSPLY
2059 E3             03810         EX      (SP),HL
205A 115321         03820         LD      DE,TIMFLD       ;Pt to time field
205D 0603           03830         LD      B,3             ;Set up loop counter
205F 1805           03840         JR      PAKTIM1
2061 FE0A           03850 PAKTIM  CP      ':'-30H         ;Test valid separator
2063 C2521F         03860         JP      NZ,BADJCL
2066 C5             03870 PAKTIM1 PUSH    BC
2067 CDE103         03880         CALL    @DECHEX         ;Cvrt the hours
206A 71             03890         LD      (HL),C          ;Store time parm
206B EDA0           03900         LDI                     ;Shift & bump HL & DE
206D C1             03910         POP     BC              ;Rcvr the loop counter
206E 10F1           03920         DJNZ    PAKTIM          ;Loop for 3 values
2070 3E00           03930 SLPWT   LD      A,0             ;P/u sleep/wait flag
2072 B7             03940         OR      A
2073 281F           03950         JR      Z,TSTIME        ;Go if //WAIT
2075 215521         03960         LD      HL,TIMFLD+2     ;Point to seconds
2078 112D00         03970         LD      DE,TIME$
207B 0602           03980         LD      B,2
207D 1A             03990 SLP1    LD      A,(DE)          ;Add secs/mins
207E 86             04000         ADD     A,(HL)
207F 77             04010         LD      (HL),A          ;Store
2080 D63C           04020         SUB     60              ;Ck overflow to mins/hrs
2082 3804           04030         JR      C,SLP2          ;Go if none
2084 77             04040         LD      (HL),A          ;Update value mod 60
2085 2B             04050         DEC     HL              ;  & bump next field
2086 34             04060         INC     (HL)
2087 23             04070         INC     HL              ;Adj for dec
2088 13             04080 SLP2    INC     DE              ;Bump time$
2089 2B             04090         DEC     HL              ;Bump user field
208A 10F1           04100         DJNZ    SLP1
208C 1A             04110         LD      A,(DE)          ;Add hours
208D 86             04120         ADD     A,(HL)
208E 77             04130         LD      (HL),A
208F D618           04140         SUB     24              ;Wrap past midnight?
2091 3801           04150         JR      C,TSTIME        ;Go if not else
2093 77             04160         LD      (HL),A          ;  adjust mod 24
                    04170 ;
                    04180 ;       Wait until the system clock advances to request
                    04190 ;
2094 CD881E         04200 TSTIME  CALL    KSCN            ;Scan for BREAK
2097 215321         04210         LD      HL,TIMFLD
209A 112F00         04220         LD      DE,TIME$+2
209D 0603           04230         LD      B,3             ;Set loop counter
209F 1A             04240 CKTIME  LD      A,(DE)          ;P/u a time value
20A0 BE             04250         CP      (HL)            ;Match user input?
20A1 20F1           04260         JR      NZ,TSTIME       ;Go if no match
20A3 23             04270         INC     HL              ;Inc the user req ptr
20A4 1B             04280         DEC     DE              ;Dec the time string ptr
20A5 10F8           04290         DJNZ    CKTIME          ;Loop for 3 values
20A7 18A4           04300         JR      FLASH3          ;All match, exit!
                    04310 ;
                    04320 ;       Process //INPUT request
                    04330 ;
```

```
20A9 E1       04340 INPUT   POP     HL              ;Recover JCL line &
20AA CD2D05   04350         CALL    @DSPLY          ;  pump it to screen
20AD 3EDD     04360         LD      A,0DDH          ;Change sysres hook
20AF 32AA19   04370         LD      (@DOKEY+1),A
20B2 D1       04380         POP     DE              ;Stack integrity
20B3 C1       04390         POP     BC              ;Get @KEYIN values
              04400 ;
              04410 ;       This next routine will satisfy the request
              04420 ;
20B4 CD2806   04430 GETKEY  CALL    @KEY            ;Fetch from keyboard
20B7 F5       04440         PUSH    AF              ;Don't disturb flag
20B8 3D       04450         DEC     A
20B9 2806     04460         JR      Z,UNHOOK        ;Change back on BREAK
20BB FE0C     04470         CP      CR-1            ;  or ENTER
20BD 2802     04480         JR      Z,UNHOOK
20BF F1       04490         POP     AF
20C0 C9       04500         RET
20C1 3ECD     04510 UNHOOK  LD      A,0CDH          ;Restore sysres hook
20C3 32AA19   04520         LD      (@DOKEY+1),A
20C6 F1       04530         POP     AF              ;Get saved character
20C7 C9       04540         RET
              04550 ;
              04560 ;       Parameter list & scanners
              04570 ;
              04580 ;       Parse a field
              04590 ;       (HL) => command line
              04600 ;       (DE) => FCB area
              04610 ;       Z    <= found valid field
              04620 ;       NZ   <= found invalid field
              04630 ;
20C8 0608     04640 PARSER  LD      B,8             ;Set length
20CA 78       04650 PAR1    LD      A,B
20CB 32FF20   04660         LD      (PAR6+1),A
20CE 04       04670         INC     B
20CF 7E       04680 PAR2    LD      A,(HL)
20D0 FE03     04690         CP      3               ;ETX?
20D2 2826     04700         JR      Z,PAR5
20D4 FE0D     04710         CP      CR              ;<ENTER>?
20D6 2822     04720         JR      Z,PAR5
20D8 FE28     04730         CP      '('             ;Begin of parm?
20DA 281E     04740         JR      Z,PAR5
20DC 23       04750         INC     HL              ;Bump pointer to next
20DD CD0321   04760         CALL    TST09AZ         ;Test if 0-9,A-Z
20E0 300A     04770         JR      NC,PAR3         ;Go if one of the above
20E2 FE61     04780         CP      'a'             ;Check on lower case
20E4 3814     04790         JR      C,PAR5          ;Jump on non-alpha
20E6 FE7B     04800         CP      'z'+1           ;Is it a-z?
20E8 3010     04810         JR      NC,PAR5         ;Jump on non-alpha
20EA CBAF     04820         RES     5,A             ;Convert lower to upper
20EC 05       04830 PAR3    DEC     B               ;Count down
20ED 2808     04840         JR      Z,PAR4
20EF 12       04850         LD      (DE),A          ;Xfer the char
20F0 AF       04860         XOR     A               ;Show at least 1 valid
20F1 32FF20   04870         LD      (PAR6+1),A      ;  char was detected
20F4 13       04880         INC     DE              ;Bump FCB pointer
20F5 18D8     04890         JR      PAR2            ;Loop
              04900 ;
20F7 04       04910 PAR4    INC     B               ;Here on max chars ck'd
20F8 18D5     04920         JR      PAR2
20FA 4F       04930 PAR5    LD      C,A             ;Save separator
20FB 3E03     04940         LD      A,3             ;Stuff ETX
```

```
20FD 12       04950          LD      (DE),A
20FE 3E00     04960 PAR6     LD      A,0             ;Set Z-flag if at least
2100 B7       04970          OR      A               ;  1 valid char detected
2101 79       04980          LD      A,C             ;Recover separator char
2102 C9       04990          RET
2103 FE30     05000 TST09AZ  CP      '0'             ;Special character?
2105 D8       05010          RET     C               ;Go if not in range
2106 FE3A     05020          CP      '9'+1           ;Jump on digit 0-9
2108 3805     05030          JR      C,EXITC         ;Go if 0-9 & make NC
210A FE41     05040          CP      'A'             ;Jump on spec char
210C D8       05050          RET     C               ;Go with C-flag if 3B-40
210D FE5B     05060          CP      'Z'+1           ;Jump on A-Z
210F 3F       05070 EXITC    CCF                     ;Switch flag of result
2110 C9       05080          RET
              05090 ;
              05100 ;        Find parameter in table
              05110 ;        (HL) => pointer to line
              05120 ;        (DE) => pointer to buffer area
              05130 ;        (BC) => pointer to parameter table
              05140 ;          C  <= entry # of parm in table
              05150 ;        (DE) <= parm vector address
              05160 ;          Z <= set if found
              05170 ;         NZ <= if not found in table
              05180 ;        Routine similar as FIND.PARM in SYS1 - dif width
              05190 ;
2111 E5       05200 FNDPARM  PUSH    HL
2112 60       05210          LD      H,B             ;Xfer the table address
2113 69       05220          LD      L,C
2114 1A       05230 FND1     LD      A,(DE)          ;P/u input byte
2115 BE       05240          CP      (HL)            ;Match 1st char of table?
2116 280D     05250          JR      Z,FND3          ;Jump if 1st matches
2118 C5       05260 FND2     PUSH    BC              ;  else bypass that entry
2119 010700   05270          LD      BC,7            ;Width of table
211C 09       05280          ADD     HL,BC
211D C1       05290          POP     BC
211E 7E       05300          LD      A,(HL)          ;Test for table end
211F B7       05310          OR      A
2120 20F2     05320          JR      NZ,FND1         ;Loop if not at end
2122 E1       05330          POP     HL
2123 3C       05340          INC     A               ;  else set NZ return
2124 C9       05350          RET
              05360 ;
              05370 ;        1st matches, does the rest?
              05380 ;
2125 0604     05390 FND3     LD      B,4             ;# chars remaining
2127 E5       05400          PUSH    HL
2128 D5       05410          PUSH    DE
2129 13       05420 FND4     INC     DE
212A 23       05430          INC     HL
212B 1A       05440          LD      A,(DE)          ;P/u input char
212C FE03     05450          CP      3               ;ETX?
212E 281A     05460          JR      Z,FND7
2130 FE0D     05470          CP      CR              ;End of line?
2132 2816     05480          JR      Z,FND7
2134 BE       05490          CP      (HL)            ;Match with table?
2135 200E     05500          JR      NZ,FND6         ;Exit & test the char
2137 10F0     05510          DJNZ    FND4            ;Loop for limit
2139 D1       05520 FND5     POP     DE              ;Must be a match
213A C1       05530          POP     BC
213B 210500   05540          LD      HL,5            ;Point to vector
213E 09       05550          ADD     HL,BC
```

```
213F 5E        Ø5560          LD      E,(HL)          ;Xfer vector to DE
2140 23        Ø5570          INC     HL
2141 56        Ø5580          LD      D,(HL)
2142 E1        Ø5590          POP     HL
2143 AF        Ø5600          XOR     A               ;  & show it found
2144 C9        Ø5610          RET
               Ø5620 ;
               Ø5630 ;        No match if alphanumeric unless a space
               Ø5640 ;
2145 CDØ321    Ø5650 FND6     CALL    TSTØ9AZ         ;Ck for Ø-9, A-Z
2148 3ØØ5      Ø5660          JR      NC,FND8         ;Go if one of the above
214A 7E        Ø5670 FND7     LD      A,(HL)          ;Loop if table has
214B FE2Ø      Ø5680          CP      ' '             ;  trailing spaces
214D 28EA      Ø5690          JR      Z,FND5
214F D1        Ø5700 FND8     POP     DE
2150 E1        Ø5710          POP     HL
2151 18C5      Ø5720          JR      FND2
               Ø5730 ;
ØØØ6           Ø5740 LILBUF   DS      6
2153           Ø5750 TIMFLD   EQU     LILBUF
2159 42        Ø5760 BADJCL$ DB       'Bad JCL, '
     61 64 2Ø 4A 43 4C 2C 2Ø
2162 4A        Ø5770 ABORT$   DB       'Job aborted',CR
     6F 62 2Ø 61 62 6F 72 74
     65 64 ØD
216E 4A        Ø5780 JOBDUN$ DB       'Job done',CR
     6F 62 2Ø 64 6F 6E 65 ØD
2177 41        Ø5790 PARMTBL DB       'ABORT'
     42 4F 52 54
217C 8Ø1E      Ø5800          DW      ABORT
217E 41        Ø5810          DB      'ALERT'
     4C 45 52 54
2183 CB1F      Ø5820          DW      ALERT
2185 44        Ø5830          DB      'DELAY'
     45 4C 41 59
218A 621F      Ø5840          DW      DELAY
218C 45        Ø5850          DB      'EXIT '
     58 49 54 2Ø
2191 B21E      Ø5860          DW      EXIT
2193 46        Ø5870          DB      'FLASH'
     4C 41 53 48
2198 252Ø      Ø5880          DW      FLASH
219A 4B        Ø5890          DB      'KEYIN'
     45 59 49 4E
219F 7E1F      Ø5900          DW      KEYIN
21A1 5Ø        Ø5910          DB      'PAUSE'
     41 55 53 45
21A6 721F      Ø5920          DW      PAUSE
21A8 53        Ø5930          DB      'SLEEP'
     4C 45 45 5Ø
21AD 5Ø2Ø      Ø5940          DW      SLEEP
21AF 53        Ø5950          DB      'STOP '
     54 4F 5Ø 2Ø
21B4 581F      Ø5960          DW      STOP
21B6 57        Ø5970          DB      'WAIT '
     41 49 54 2Ø
21BB 512Ø      Ø5980          DW      WAIT
21BD 49        Ø5990          DB      'INPUT'
     4E 5Ø 55 54
21C2 A92Ø      Ø6000          DW      INPUT
21C4 ØØ        Ø6010          NOP
```

```
21C5          Ø62Ø LAST    EQU     $
              Ø63Ø         IFGT    $,DIRBUF$
              Ø64Ø         ERR     'Module too big'
              Ø65Ø         ENDIF
23FE          Ø66Ø         ORG     MAXCOR$-2
23FE C5Ø3     Ø67Ø         DW      LAST-SYS11      ;Overlay size
              Ø68Ø ;
1EØØ          Ø69Ø         END     SYS11
```

$?1	1E74	$A1	Ø3B7
$A3	Ø3B9	$CKEOF	147Ø
@@1	ØØØØ	@@2	ØØØØ
@@4	ØØØØ	@ABORT	1BØ8
@BANK	Ø877	@BKSP	1486
@BYTEIO	13ØØ	@CHNIO	Ø689
@CKDRV	1993	@CKEOF	158F
@CLOSE	1999	@CLS	Ø545
@CMNDR	197B	@CTL	Ø623
@DBGHK	199F	@DCINIT	19CØ
@DCSTAT	19B5	@DCTBYT	1A2B
@DECHEX	Ø3E1	@DIRCYL	18F7
@DIRWR	18Ø3	@DIV16	Ø6E3
@DODIR	19AF	@DOKEY	19A9
@DSPLY	Ø52D	@ERROR	1BØF
@FEXT	1984	@FLAGS	196A
@FRENCH	ØØØØ	@FSPEC	1981
@GATWR	1875	@GERMAN	ØØØØ
@GTDCB	199Ø	@GTDCT	1A1E
@HDFMT	19E4	@HEX16	Ø7BD
@HEXDEC	Ø6F6	@HIGH$	1948
@HITWR	1898	@HZ5Ø	ØØØØ
@INIT	198D	@INTL	ØØØØ
@JCL	Ø63Ø	@KBD	Ø635
@KEYIN	Ø585	@KITSK	ØØ89
@LOAD	1B38	@LOC	14B3
@LOGER	Ø5Ø3	@LOGOT	Ø5ØØ
@MOD4	FFFF	@MSG	Ø53Ø
@MUL8	19ØA	@NMI	ØØ66
@OPREG	ØØ84	@PARAM	1987
@PEOF	14A2	@POSN	1434
@PRT	Ø63D	@PUT	Ø645
@RDHDR	19D8	@RDSEC	19F4
@RDTRK	19EØ	@READ	1513
@RENAME	1996	@REW	149B
@RPTSK	1CEB	@RREAD	1473
@RSTØØ	ØØØØ	@RSTØ8	ØØØ8
@RST18	ØØ18	@RST2Ø	ØØ2Ø
@RST3Ø	ØØ3Ø	@RST38	ØØ38
@RSTOR	19C8	@RSTREG	Ø68Ø
@RWRIT	13AD	@SEEK	19DØ
@SKIP	143Ø	@SLCT	19BC
@STEPI	19CC	@TIME	Ø78D
@VDCTL	ØB99	@VDCTL3	ØD38
@VRSEC	19DC	@WEOF	14EC
@WRITE	1531	@WRSEC	19E8
@WRTRK	19FØ	@_VDCTL	ØD42
ABORT$	2162	ADDR_2_ROWCOL	ØDF1
ALERT	1FCB	ALERT1	1FCF
ALERT3	1FF5	ALERT4	1FFA
BADJCL	1F52	BADJCL$	2159
BOOTST$	439D	BREAKA	1C6Ø
BUR$	Ø2ØØ	CASHK$	ØA7B
CFGFCB$	ØØEØ	CFLAG$	ØØ6C
CKOPEN@	1568	CKTIME	2Ø9F
CORE$	Ø3ØØ	CR	ØØØD
CYL_GRN	16AE	D@FBYT8	1A26
DAYTBL$	Ø4C7	DBGSV$	ØØAØ
DCT$	Ø47Ø	DCTBYT8@	1A29
DELAY	1F62	DELAY1	1F6B

$A2	Ø3B8	$CKEOF	147Ø
@$SYS	Ø8FØ	@@1	ØØØØ
@@3	ØØØØ	@@4	ØØØØ
@ABORT	1BØ8	@ADTSK	1CDA
@BANK	Ø877	@BKSP	1486
@BREAK	196F	@BYTEIO	13ØØ
@CHNIO	Ø689	@CKBRKC	Ø553
@CKDRV	1993	@CKEOF	158F
@CKTSK	1CF5	@CLOSE	1999
@CLS	Ø545	@CMNDI	197E
@CMNDR	197B	@CTL	Ø623
@DATE	Ø7A8	@DBGHK	199F
@DCINIT	19CØ	@DCRES	19C4
@DCSTAT	19B5	@DCTBYT	1A2B
@DEBUG	19AØ	@DECHEX	Ø3E1
@DIRCYL	18F7	@DIRRD	18BB
@DIRWR	18Ø3	@DIV16	1927
@DIV8	Ø6E3	@DODIR	19AF
@DOKEY	19A9	@DSP	Ø642
@DSPLY	Ø52D	@ERROR	1BØF
@EXIT	1BØB	@FEXT	1984
@FLAGS	196A	@FNAME	199C
@FRENCH	ØØØØ	@FSPEC	1981
@GATRD	1874	@GATWR	1875
@GERMAN	ØØØØ	@GET	Ø638
@GTDCB	199Ø	@GTDCT	1A1E
@GTMOD	19B2	@HDFMT	19E4
@HEX16	Ø7BD	@HEX8	Ø7C2
@HEXDEC	Ø6F6	@HIGH$	1948
@HITRD	1897	@HITWR	1898
@HZ5Ø	ØØØØ	@ICNFG	ØØ86
@INIT	198D	@INTL	ØØØØ
@IPL	1BF2	@JCL	Ø63Ø
@KBD	Ø635	@KEY	Ø628
@KEYIN	Ø585	@KITSK	ØØ89
@KLTSK	1CDØ	@LOAD	1B38
@LOC	14B3	@LOF	14DE
@LOGER	Ø5Ø3	@LOGOT	Ø5ØØ
@MOD2	ØØØØ	@MOD4	FFFF
@MSG	Ø53Ø	@MUL16	Ø6C9
@MUL8	19ØA	@NMI	ØØ66
@OPEN	198A	@OPREG	ØØ84
@PARAM	1987	@PAUSE	Ø382
@PEOF	14A2	@POSN	1434
@PRINT	Ø528	@PRT	Ø63D
@PUT	Ø645	@RAMDIR	19AC
@RDHDR	19D8	@RDSEC	19F4
@RDSSC	18D8	@RDTRK	19EØ
@READ	1513	@REMOVE	19A6
@RENAME	1996	@REW	149B
@RMTSK	1CD7	@RPTSK	1CEB
@RREAD	1473	@RSLCT	19D4
@RSTØØ	ØØØØ	@RSTØ8	ØØØ8
@RST1Ø	ØØ1Ø	@RST18	ØØ18
@RST2Ø	ØØ2Ø	@RST28	ØØ28
@RST3Ø	ØØ3Ø	@RST38	ØØ38
@RSTNMI	ØFE9	@RSTOR	19C8
@RSTREG	Ø68Ø	@RUN	1B1D
@RWRIT	13AD	@SEEK	19DØ
@SEEKSC	1421	@SKIP	143Ø
@SLCT	19BC	@SOUND	Ø392
@STEPI	19CC	@TIME	Ø78D
@USA	FFFF	@VDCTL	ØB99
@VDCTL3	ØD38	@VER	156Ø
@VRSEC	19DC	@WEOF	14EC
@WHERE	1979	@WRITE	1531
@WRSEC	19E8	@WRSSC	19EC
@WRTRK	19FØ	@_VDCTL	ØD42
ABORT	1E8Ø	ABORT$	2162
ADDR_2_ROWCOL	ØDF1	AFLAG$	ØØ6A
ALERT	1FCB	ALERT1	1FCF
ALERT2	1FEB	ALERT3	1FF5
ALERT4	1FFA	AUTOA	1FF1
BADJCL	1F52	BADJCL$	2159
BAR$	Ø2Ø1	BOOTST$	439D
BREAKA	1C6Ø	BRKVEC$	1C88
BUR$	Ø2ØØ	CASHK$	ØA7B
CFCB$	ØØEØ	CFGFCB$	ØØEØ
CFLAG$	ØØ6C	CKMOD@	1A7F
CKOPEN@	1568	CKTIME	2Ø9F
CONFIG$	2Ø3F	CORE$	Ø3ØØ
CR	ØØØD	CRTBGN$	F8ØØ
CYL_GRN	16AE	D@FBYT8	1A26
DATE$	ØØ33	DAYTBL$	Ø4C7
DBGSV$	ØØAØ	DCBKL$	ØØ31
DCT$	Ø47Ø	DCTBYT8@	1A29
DCTFLD@	1A34	DELAY	1F62
DELAY1	1F6B	DFLAG$	ØØ6D

```
DIRBUF$          23ØØ DIS_DO_RAM      Ø846 DODATA$         ØB94
DODCB$           Ø21Ø DOOFF           1EBC DO_CONTROL      ØC44
DO_DSPCHAR       ØCB8 DO_INVERT_DIS   ØC8C DO_INVERT_ENA   ØC89
DO_INVERT_OFF    ØC9B DO_MASK         ØØØØ DO_RET          ØBCB
DO_RET1          ØBCC DO_SCROLL       ØCCE DO_TABS         ØBEA
DSKTYP$          Ø4CØ DTPMT$          Ø4C2 DVREND$         ØFF4
DVRHI$           Ø2Ø6 EFLAG$          ØØ6E ENADIS_DO_RAM   Ø817
EXIT             1EB2 EXIT1           1EB8 EXITC           21ØF
EXTDBG$          19A4 FDDINT$         ØØØE FEMSK$          ØØ6F
FLASH            2Ø25 FLASH1          2Ø2B FLASH2          2Ø32
FLASH3           2Ø4D FLGTAB$         ØØ6A FND1            2114
FND2             2118 FND3            2125 FND4            2129
FND5             2139 FND6            2145 FND7            214A
FND8             214F FNDPARM         2111 GETKEY          2ØB4
GETLINE          1E77 GET_@_ROWCOL    ØDAE GOTSL1          1FØ6
GOTSL2           1F13 GOTSL3          1F2A GOTSLSH         1EFE
HERTZ$           Ø75Ø HIGH$           Ø4ØE HKRES$          1A6C
IFLAG$           ØØ72 INBUF$          Ø42Ø INPUT           2ØA9
INTIM$           ØØ3C INTMSK$         ØØ3D INTVC$          ØØ3E
IPLDO1           1E2F IPLDO2          1E4F JCLCB$          Ø2Ø3
JDCB$            ØØ24 JFCB$           ØØCØ JLDCB$          Ø23Ø
JOBDUN$          216E JRET$           ØØ26 KCK@            Ø7D6
KEYIN            1F7E KEYIN1          1F8Ø KEYIN2          1F8B
KEYIN3           1F99 KEYIN4          1FA4 KEYIN5          1FBD
KEYIN6           1FC1 KEYLINE         1EDD KEYREQ          1ED6
KFLAG$           ØØ74 KIDATA$         Ø8FC KIDCB$          Ø2Ø8
KSCN             1E88 KSCN1           1E96 KSCN2           1E99
LAST             21C5 LBANK$          Ø2Ø2 LDRV$           ØØ23
LF               ØØØA LFLAG$          ØØ75 LILBUF          2153
LNKFCB@          1566 LOW$            ØØ1E LSVC$           ØØØD
MAXCOR$          24ØØ MAXDAY$         Ø4Ø1 MINCOR$         3ØØØ
MODOUT$          ØØ76 MONTBL$         Ø4DC NEWEX1          1E69
NEWEXIT          1E55 NFLAG$          ØØ77 OPREG$          ØØ78
OPREG_SV_AREA    Ø86E OPREG_SV_PTR    Ø835 ORARET@         14DC
OSRLS$           ØØ3B OSVER$          ØØ85 OVRLY$          ØØ69
PAKNAM$          Ø41Ø PAKTIM          2Ø61 PAKTIM1         2Ø66
PAR1             2ØCA PAR2            2ØCF PAR3            2ØEC
PAR4             2ØF7 PAR5            2ØFA PAR6            2ØFE
PARMTBL          2177 PARSER          2ØC8 PAUSE           1F72
PAUSE1           1F77 PAUSE@          Ø382 PCSAVE$         Ø7AF
PDRV$            ØØ1B PHIGH$          ØØ1C PRDCB$          Ø218
PUTA@DE          ØDCD PUT_@           ØDCA PUT_@_ROWCOL    ØDC6
REKEY1           1F4C REKEY2          1F4D RFLAG$          ØØ7B
ROWCOL_2_ADDR    ØDDØ RST38@          1BFF RSTOR$          Ø4C4
RWRIT@           13A2 S1DCB$          Ø238 SBUFF$          1DØØ
SET@EXEC         1A79 SET_SCROLL      ØCF3 SFCB$           ØØ8C
SFLAG$           ØØ7C SIDCB$          Ø22Ø SILEN1          2Ø1C
SILENCE          2Ø14 SLEEP           2Ø5Ø SLP1            2Ø7D
SLP2             2Ø88 SLPWT           2Ø7Ø SODCB$          Ø228
SPACE4$          2142 STACK$          Ø38Ø START$          ØØØØ
STOP             1F58 SVCRET$         ØØØB SVCTAB$         Ø1ØØ
SYS11            1EØØ SYSERR$         1B13 TCB$            ØØ4E
TFLAG$           ØØ7D TIME$           ØØ2D TIMER$          ØØ2C
TIMFLD           2153 TIMSL$          ØØ2B TIMTSK$         Ø713
TMPMT$           Ø4C3 TRACE_INT       Ø7B1 TSTØ9AZ         21Ø3
TSTIME           2Ø94 TYPHK$          ØA8F TYPTSK$         ØB26
UNHOOK           2ØC1 USTOR$          ØØ13 VFLAG$          ØØ7F
WAIT             2Ø51 WRINT$          ØØ8Ø ZERO$           Ø4Ø1
ZEROA@           13AØ
1EØØ is the transfer address
ØØØØØ Total errors
```

NOTES:

NOTES:

SYS12/SYS

SYS12 handles the two mini directory and free space SVCs, as well as locates or checks for a memory module header. It contains the code for the SVCs @DODIR, @RAMDIR and @GTMOD.

```
                00100 ;SYS12/ASM - LS-DOS 6.2
0000            00110         TITLE   <SYS12 - LS-DOS 6.2>
                00120 ;
000D            00130 CR      EQU     13
                00140 *LIST   OFF                     ;Get SYS0/EQU
                00160 *LIST   ON
0000            00170 *GET    COPYCOM:3               ;Copyright message
                03010 ; COPYCOM - File for Copyright COMment block
                03020 ;
0000            03030         COM     '<*(C) 1982,83,84 by LSI*>'
                03040 ;
                00180 ;
1E00            00190         ORG     1E00H
                00200 ;
1E00 E670       00210 SYS12   AND     70H             ;Strip bit 7
1E02 C8         00220         RET     Z               ;Back on zero entry
1E03 FE30       00230         CP      30H             ;Locate module address?
1E05 CAAD20     00240         JP      Z,GTMOD
1E08 FE20       00250         CP      20H             ;Mini dir?
1E0A CAF01E     00260         JP      Z,MDIR
1E0D FE10       00270         CP      10H             ;RAMDIR?
1E0F C0         00280         RET     NZ              ;Ret if any other entry
                00290 ;
                00300 ;       RAMDIR interfacing
                00310 ;       HL = user buffer area
                00320 ;        B = drive #
                00330 ;        C = 0 for entire directory
                00340 ;        C = 1-254 for selected DEC-1 (02-FF)
                00350 ;        C = 255 for disk space; in use/free
                00360 ;
1E10 3E07       00370 RAMDIR  LD      A,7             ;Ck on valid drive #
1E12 B8         00380         CP      B
1E13 3E20       00390         LD      A,32            ;Init "illegal drive
1E15 D8         00400         RET     C
1E16 CD6615     00410         CALL    LNKFCB@         ;Save regs
1E19 78         00420         LD      A,B             ;Get drive where needed
1E1A 41         00430         LD      B,C             ;Xfer DEC to B
1E1B 4F         00440         LD      C,A             ;  & drive to C
1E1C F630       00450         OR      '0'             ;Make it ASCII
1E1E 32162D     00460         LD      (DSTDRV+1),A    ;Stuff for STUFBUF
1E21 CD0B21     00470         CALL    CKDRV           ;Be sure disk is there
1E24 C0         00480         RET     NZ
1E25 04         00490         INC     B               ;Test 0, 1-254, 255
1E26 2019       00500         JR      NZ,DIRINFO      ;Go if directory req
                00510 ;
                00520 ;       Get FREE SPACE info
                00530 ;
1E28 E5         00540         PUSH    HL              ;Save buffer pointer
1E29 CD362D     00550         CALL    SPACE           ;Get our info
1E2C 46         00560         LD      B,(HL)          ;P/u free space in K
1E2D 2B         00570         DEC     HL              ;  into BC
1E2E 4E         00580         LD      C,(HL)
1E2F 2B         00590         DEC     HL
1E30 7E         00600         LD      A,(HL)          ;Get total space in K
1E31 2B         00610         DEC     HL              ;  into HL
1E32 6E         00620         LD      L,(HL)
1E33 67         00630         LD      H,A
1E34 ED52       00640         SBC     HL,DE           ;Calc "in use" (CF=0)
1E36 EB         00650         EX      DE,HL           ;Xfer to DE
1E37 E1         00660         POP     HL              ;Rcvr user buf ptr
1E38 73         00670         LD      (HL),E          ;Stuff "in use"
```

```
1E39 23       ØØ68Ø          INC     HL
1E3A 72       ØØ69Ø          LD      (HL),D
1E3B 23       ØØ7ØØ          INC     HL
1E3C 71       ØØ71Ø          LD      (HL),C          ;Stuff "free to use"
1E3D 23       ØØ72Ø          INC     HL
1E3E 7Ø       ØØ73Ø          LD      (HL),B
1E3F AF       ØØ74Ø          XOR     A               ;Show no error
1E4Ø C9       ØØ75Ø          RET
              ØØ76Ø ;
              ØØ77Ø ;        Do RAMDIR directory info
              ØØ78Ø ;
1E41 Ø5       ØØ79Ø DIRINFO  DEC     B               ;If DEC=Ø, do it all
1E42 286C     ØØ8ØØ          JR      Z,DOALL         ;Go if all of it
1E44 Ø4       ØØ81Ø          INC     B               ;1=>2, 2=>3, ..., FE=>FF
              ØØ82Ø ;
              ØØ83Ø ;        Calculate the number of directory sectors
              ØØ84Ø ;        = (#sectors x #heads) - 2 for GAT & HIT
              ØØ85Ø ;
1E45 3EØ7     ØØ86Ø          LD      A,7             ;Get highest # sector
1E47 CD2B1A   ØØ87Ø          CALL    @DCTBYT
1E4A 57       ØØ88Ø          LD      D,A             ;Store heads & sectors
1E4B E61F     ØØ89Ø          AND     1FH             ;Rake off # sectors
1E4D 5F       ØØ9ØØ          LD      E,A             ;  & stuff into E
1E4E 1C       ØØ91Ø          INC     E               ;Bump for Ø offset
1E4F AA       ØØ92Ø          XOR     D               ;Recover # heads
1E5Ø Ø7       ØØ93Ø          RLCA                    ;  into bits Ø-2
1E51 Ø7       ØØ94Ø          RLCA
1E52 Ø7       ØØ95Ø          RLCA
1E53 3C       ØØ96Ø          INC     A               ;Bump for Ø offset
1E54 CDØA19   ØØ97Ø          CALL    @MUL8           ;Multiply sectors x heads
1E57 5F       ØØ98Ø          LD      E,A             ;Now check double bit
1E58 3EØ4     ØØ99Ø          LD      A,4
1E5A CD2B1A   Ø1ØØØ          CALL    @DCTBYT
1E5D CB6F     Ø1Ø1Ø          BIT     5,A             ;Set if 2-sided
1E5F 7B       Ø1Ø2Ø          LD      A,E
1E6Ø 28Ø1     Ø1Ø3Ø          JR      Z,ONESID        ;Go if not set else
1E62 87       Ø1Ø4Ø          ADD     A,A             ;  double value
1E63 D6Ø2     Ø1Ø5Ø ONESID   SUB     2               ;Reduce for GAT & HIT
1E65 57       Ø1Ø6Ø          LD      D,A             ;D => # dir sectors
1E66 78       Ø1Ø7Ø          LD      A,B             ;Get requested DEC
1E67 E61F     Ø1Ø8Ø          AND     1FH
1E69 BA       Ø1Ø9Ø          CP      D               ;See if in range
1E6A 38Ø4     Ø11ØØ          JR      C,DIRINF1       ;Go if so
1E6C 3E1Ø     Ø111Ø          LD      A,16            ;"Illegal logical file #
1E6E B7       Ø112Ø          OR      A               ;Return out of range error
1E6F C9       Ø113Ø          RET
              Ø114Ø ;
1E7Ø E5       Ø115Ø DIRINF1  PUSH    HL              ;Save buffer ptr
1E71 CDBB18   Ø116Ø          CALL    @DIRRD          ;Get its directory record
1E74 D1       Ø117Ø          POP     DE              ;Rcvr buf ptr
1E75 CØ       Ø118Ø          RET     NZ              ;Back on an error
1E76 7E       Ø119Ø          LD      A,(HL)          ;Get attributes
1E77 E6D8     Ø12ØØ          AND     ØD8H            ;Only if in use & VIS
1E79 EE1Ø     Ø121Ø          XOR     1ØH             ;Flip state so NZ=no
1E7B 3E19     Ø122Ø          LD      A,25            ;Init file access denied
1E7D CØ       Ø123Ø          RET     NZ              ;Back on no file, SYS, INV
1E7E E5       Ø124Ø GETSTUF  PUSH    HL              ;Save DIR ptr
1E7F CDE11F   Ø125Ø          CALL    STUFBUF         ;Stuff the filespec
1E82 E1       Ø126Ø          POP     HL
1E83 7E       Ø127Ø          LD      A,(HL)
1E84 E6Ø7     Ø128Ø          AND     7               ;Keep the access level
```

```
1E86 12       01290          LD      (DE),A
1E87 13       01300          INC     DE
1E88 2C       01310          INC     L               ;Go up to EOF offset
1E89 2C       01320          INC     L
1E8A 2C       01330          INC     L
1E8B EDA0     01340          LDI                     ;Move in the offset & LRL
1E8D EDA0     01350          LDI
1E8F 7D       01360          LD      A,L             ;Bump to ERN
1E90 C60F     01370          ADD     A,15
1E92 6F       01380          LD      L,A
1E93 7E       01390          LD      A,(HL)          ;P/u ERN
1E94 12       01400          LD      (DE),A          ;  and xfer it
1E95 2C       01410          INC     L
1E96 13       01420          INC     DE
1E97 66       01430          LD      H,(HL)
1E98 6F       01440          LD      L,A             ;# sectors to HL
1E99 EB       01450          EX      DE,HL           ;  hence to DE
1E9A 72       01460          LD      (HL),D          ;Stuff ERN High-order
1E9B 23       01470          INC     HL              ;Bump buf ptr
1E9C 13       01480          INC     DE              ;Adjust for rounding
1E9D 13       01490          INC     DE
1E9E 13       01500          INC     DE
1E9F CB3A     01510          SRL     D               ;Divide by 4 to calc K
1EA1 CB1B     01520          RR      E
1EA3 CB3A     01530          SRL     D
1EA5 CB1B     01540          RR      E
1EA7 73       01550          LD      (HL),E          ;Xfer result into buffer
1EA8 23       01560          INC     HL
1EA9 72       01570          LD      (HL),D
1EAA 23       01580          INC     HL
1EAB 362B     01590          LD      (HL),'+'        ;Stuff buffer terminator
1EAD EB       01600          EX      DE,HL
1EAE AF       01610          XOR     A
1EAF C9       01620          RET
              01630 ;
              01640 ;        RAMDIR - Do all of the directory
              01650 ;
1EB0 EB       01660 DOALL    EX      DE,HL           ;Buffer pointer to DE
1EB1 CD9B20   01670          CALL    HITRD1          ;Read in the HIT
1EB4 C0       01680          RET     NZ              ;Exit if read error
1EB5 1811     01690          JR      DOALL3
              01700 ;
1EB7 C1       01710 DOALL1   POP     BC              ;Recover HIT pointer lo
1EB8 2623     01720          LD      H,DIRBUF$<-8
1EBA 68       01730          LD      L,B             ;Advance to next dir
1EBB 7D       01740 DOALL2   LD      A,L             ;  record of this sector
1EBC C620     01750          ADD     A,32
1EBE 6F       01760          LD      L,A
1EBF 3007     01770          JR      NC,DOALL3       ;Bypass if still same
1EC1 2C       01780          INC     L               ;  else point to next one
1EC2 CB6D     01790          BIT     5,L             ;Finished with
1EC4 2802     01800          JR      Z,DOALL3        ;  this drive?
1EC6 AF       01810          XOR     A
1EC7 C9       01820          RET
              01830 ;
1EC8 7E       01840 DOALL3   LD      A,(HL)          ;P/u HIT entry
1EC9 B7       01850          OR      A
1ECA 28EF     01860          JR      Z,DOALL2        ;Jump if spare
1ECC 45       01870          LD      B,L             ;Save DEC in regB
1ECD C5       01880          PUSH    BC              ;  & to stack
1ECE 7D       01890          LD      A,L             ;Pt to dir record for
```

```
1ECF E6E0      01900          AND     0E0H            ;  this DEC
1ED1 6F        01910          LD      L,A             ;Get the dir sector for
1ED2 A8        01920          XOR     B               ;  this DEC
1ED3 FEFF      01930 DOALL4   CP      0FFH            ;Same as one in core?
1ED5 2809      01940          JR      Z,DOALL5        ;Jump if so else
1ED7 32D41E    01950          LD      (DOALL4+1),A    ;  update one we have and
1EDA CDBB18    01960          CALL    @DIRRD          ;  read it into buffer
1EDD C2C11F    01970          JP      NZ,MDIR12       ;Jump on read error
1EE0 261D      01980 DOALL5   LD      H,SBUFF$<-8     ;Sysbuf hi order
1EE2 7E        01990          LD      A,(HL)          ;P/u attributes
1EE3 E6D8      02000          AND     0D8H            ;Test FXDE & in-use
1EE5 EE10      02010          XOR     10H             ;If not used or FXDE
1EE7 20CE      02020          JR      NZ,DOALL1       ;  then back to DOALL1
1EE9 E5        02030          PUSH    HL
1EEA CD7E1E    02040          CALL    GETSTUF         ;Get the dir info
1EED E1        02050          POP     HL
1EEE 18C7      02060          JR      DOALL1
               02070 ;
               02080 ;        Routine to display a mini directory
               02090 ;         C => drive number in binary (0-7)
               02100 ;         B => option, 0 = display, 1 = buffer stuff
               02110 ;         2 = display /EXT, 3 = buffer /EXT
               02120 ;         4 = space into buffer
               02130 ;        HL => address of buffer to stuff dir info & EXT
               02140 ;         Z <= set on valid conclusion
               02150 ;        NZ <= set on any error
               02160 ;
1EF0 3E07      02170 MDIR     LD      A,7             ;Test for bad drive #
1EF2 B9        02180          CP      C
1EF3 3E20      02190          LD      A,32            ;Init "illegal drive...
1EF5 D8        02200          RET     C
1EF6 CD0B21    02210          CALL    CKDRV           ;Be sure disk is there
1EF9 C0        02220          RET     NZ
1EFA CD6615    02230          CALL    LNKFCB@         ;Save the regs
1EFD 78        02240          LD      A,B             ;Stuff the option
1EFE 326B1F    02250          LD      (TSTOPT+1),A
1F01 FE04      02260          CP      4               ;If option 4, go get
1F03 CA2420    02270          JP      Z,SPACE0        ;  space info
1F06 3E2B      02280          LD      A,43            ;Init "SVC parm error
1F08 D0        02290          RET     NC              ;Back if option > 4
1F09 E5        02300          PUSH    HL              ;Save possible buffer
1F0A C5        02310          PUSH    BC
1F0B 11B121    02320          LD      DE,LILBUF       ;Save possible /EXT
1F0E 010300    02330          LD      BC,3
1F11 EDB0      02340          LDIR
1F13 C1        02350          POP     BC
1F14 79        02360          LD      A,C             ;Get drive # and
1F15 F630      02370          OR      '0'             ;  make it ASCII
1F17 32162D    02380          LD      (DSTDRV+1),A
1F1A 3E05      02390          LD      A,5             ;Init to 5 files/line
1F1C 329F1F    02400          LD      (MDIR11+1),A
1F1F 3E17      02410          LD      A,23            ;  & 23 lines/page
1F21 32B01F    02420          LD      (CKPAGE+1),A
1F24 CD9B20    02430          CALL    HITRD1          ;Read in the HIT
1F27 D1        02440          POP     DE              ;Rcvr possible buffer
1F28 C0        02450          RET     NZ              ;Exit if read error
1F29 1822      02460          JR      MDIR3
1F2B C1        02470 MDIR1    POP     BC              ;Recover HIT pointer lo
1F2C 2623      02480          LD      H,DIRBUF$<-8
1F2E 68        02490          LD      L,B             ;Advance to next dir
1F2F 7D        02500 MDIR2    LD      A,L             ;  record of this sector
```

```
1F3Ø C62Ø     Ø251Ø          ADD     A,32
1F32 6F       Ø252Ø          LD      L,A
1F33 3Ø18     Ø253Ø          JR      NC,MDIR3        ;Bypass if still same
1F35 2C       Ø254Ø          INC     L               ;  else point to next one
1F36 CB6D     Ø255Ø          BIT     5,L             ;Finished with
1F38 2813     Ø256Ø          JR      Z,MDIR3         ;  this drive?
1F3A 3A6B1F   Ø257Ø          LD      A,(TSTOPT+1)    ;If option 1 or 3,
1F3D E6Ø1     Ø258Ø          AND     1               ;  must stuff buffer end
1F3F 2ØØ7     Ø259Ø          JR      NZ,CLSBUF
1F41 3EØD     Ø26ØØ          LD      A,CR            ;  else do a blank line
1F43 CD42Ø6   Ø261Ø          CALL    @DSP
1F46 AF       Ø262Ø          XOR     A
1F47 C9       Ø263Ø          RET
              Ø264Ø ;
1F48 3EFF     Ø265Ø CLSBUF   LD      A,ØFFH          ;Put in buffer terminator
1F4A 12       Ø266Ø          LD      (DE),A
1F4B AF       Ø267Ø          XOR     A
1F4C C9       Ø268Ø          RET
              Ø269Ø ;
1F4D 7E       Ø27ØØ MDIR3    LD      A,(HL)          ;P/u HIT entry
1F4E B7       Ø271Ø          OR      A
1F4F 28DE     Ø272Ø          JR      Z,MDIR2         ;Jump if spare
1F51 45       Ø273Ø          LD      B,L             ;Save DEC in reg B
1F52 C5       Ø274Ø          PUSH    BC              ;  & to stack
1F53 7D       Ø275Ø          LD      A,L             ;Pt to dir record for
1F54 E6EØ     Ø276Ø          AND     ØEØH            ;  this DEC
1F56 6F       Ø277Ø          LD      L,A             ;Get the dir sector for
1F57 A8       Ø278Ø          XOR     B               ;  this DEC
1F58 FEFF     Ø279Ø MDIR4    CP      ØFFH            ;Same as one in core?
1F5A 28Ø8     Ø28ØØ          JR      Z,MDIR5         ;Jump if so
1F5C 32591F   Ø281Ø          LD      (MDIR4+1),A     ;Else update one we have
1F5F CDBB18   Ø282Ø          CALL    @DIRRD          ;  and read it into buf
1F62 2Ø5D     Ø283Ø          JR      NZ,MDIR12       ;Jump on read error
1F64 261D     Ø284Ø MDIR5    LD      H,SBUFF$<-8     ;Sysbuf hi order
1F66 Ø12B1F   Ø285Ø          LD      BC,MDIR1        ;Set up the return addr
1F69 C5       Ø286Ø          PUSH    BC
1F6A 3EØØ     Ø287Ø TSTOPT   LD      A,Ø             ;P/u option number
1F6C E5       Ø288Ø          PUSH    HL
1F6D D5       Ø289Ø          PUSH    DE
1F6E CDC31F   Ø29ØØ          CALL    TSTSAM          ;Check for extension match
1F71 D1       Ø291Ø          POP     DE
1F72 E1       Ø292Ø          POP     HL
1F73 CØ       Ø293Ø          RET     NZ              ;Back to MDIR1
1F74 3A6B1F   Ø294Ø          LD      A,(TSTOPT+1)
1F77 ØF       Ø295Ø          RRCA                    ;Test option 1 or 3
1F78 7E       Ø296Ø          LD      A,(HL)
1F79 3Ø13     Ø297Ø          JR      NC,DSPLYIT      ;Go if Ø or 2
1F7B E69Ø     Ø298Ø          AND     9ØH             ;Test FXDE & in-use
1F7D EE1Ø     Ø299Ø          XOR     1ØH             ;If not used, FXDE
1F7F CØ       Ø3ØØØ          RET     NZ              ;Back to MDIR1
1F8Ø Ø11ØØØ   Ø3Ø1Ø          LD      BC,16
1F83 EDBØ     Ø3Ø2Ø          LDIR                    ;User's buffer
1F85 2C       Ø3Ø3Ø          INC     L               ;Bypass stored passwords
1F86 2C       Ø3Ø4Ø          INC     L
1F87 2C       Ø3Ø5Ø          INC     L
1F88 2C       Ø3Ø6Ø          INC     L
1F89 ØE02     Ø3Ø7Ø          LD      C,2             ;  and xfer ERN
1F8B EDBØ     Ø3Ø8Ø          LDIR
1F8D C9       Ø3Ø9Ø          RET                     ;Back to MDIR1
              Ø31ØØ ;
1F8E E6D8     Ø311Ø DSPLYIT  AND     ØD8H            ;Test if we want this
```

```
1F9Ø EE1Ø     Ø312Ø          XOR     1ØH             ;Only if in-use & VIS
1F92 CØ       Ø313Ø          RET     NZ              ;Back to MDIR1
1F93 11B421   Ø314Ø          LD      DE,LILBUF+3
1F96 D5       Ø315Ø          PUSH    DE
1F97 CDE11F   Ø316Ø          CALL    STUFBUF         ;Move filespec to buffer
1F9A E1       Ø317Ø          POP     HL              ;Rcvr LILBUF ptr
1F9B CD2DØ5   Ø318Ø          CALL    @DSPLY          ;Display the file
1F9E 3EØØ     Ø319Ø MDIR11   LD      A,Ø             ;Count down 5-across
1FAØ 3D       Ø32ØØ          DEC     A
1FA1 329F1F   Ø321Ø          LD      (MDIR11+1),A    ;Update count
1FA4 CØ       Ø322Ø          RET     NZ              ;Loop if more to go
1FA5 3EØ5     Ø323Ø          LD      A,5             ;  else re-init
1FA7 329F1F   Ø324Ø          LD      (MDIR11+1),A
1FAA 3EØD     Ø325Ø          LD      A,CR
1FAC CD42Ø6   Ø326Ø          CALL    @DSP            ;New line
1FAF 3EØØ     Ø327Ø CKPAGE   LD      A,Ø             ;P/u display count
1FB1 3D       Ø328Ø          DEC     A
1FB2 32BØ1F   Ø329Ø          LD      (CKPAGE+1),A
1FB5 CØ       Ø33ØØ          RET     NZ
1FB6 3E17     Ø331Ø          LD      A,23
1FB8 32BØ1F   Ø332Ø          LD      (CKPAGE+1),A    ;Reset for max
1FBB CD28Ø6   Ø333Ø          CALL    @KEY            ;Wait for keyboard input
1FBE C345Ø5   Ø334Ø          JP      @CLS            ;Clear screen and ret
              Ø335Ø ;
1FC1 C1       Ø336Ø MDIR12   POP     BC
1FC2 C9       Ø337Ø          RET
              Ø338Ø ;
1FC3 CB4F     Ø339Ø TSTSAM   BIT     1,A             ;Ck if /EXT option
1FC5 C8       Ø34ØØ          RET     Z               ;Ret with Z if
1FC6 Ø1ØDØØ   Ø341Ø          LD      BC,13           ;  option <> /EXT
1FC9 Ø9       Ø342Ø          ADD     HL,BC           ;Else point to /EXT
1FCA Ø6Ø3     Ø343Ø          LD      B,3             ;  field of dir record
1FCC 11B121   Ø344Ø          LD      DE,LILBUF       ;  & check for match
1FCF 1A       Ø345Ø TSTS1    LD      A,(DE)
1FDØ FE24     Ø346Ø          CP      '$'             ;'$' matches with all
1FD2 28Ø8     Ø347Ø          JR      Z,TSTS2
1FD4 FE41     Ø348Ø          CP      'A'             ;If numeric, don't cvrt
1FD6 38Ø2     Ø349Ø          JR      C,$+4           ;  to upper case
1FD8 CBAF     Ø35ØØ          RES     5,A             ;Cvrt to UC if lc
1FDA BE       Ø351Ø          CP      (HL)
1FDB CØ       Ø352Ø          RET     NZ              ;Ret on no match
1FDC 23       Ø353Ø TSTS2    INC     HL
1FDD 13       Ø354Ø          INC     DE
1FDE 1ØEF     Ø355Ø          DJNZ    TSTS1           ;Loop for 3 chars
1FEØ C9       Ø356Ø          RET
              Ø357Ø ;
              Ø358Ø ;        Routine to construct the filespec field
              Ø359Ø ;
1FE1 7D       Ø36ØØ STUFBUF  LD      A,L
1FE2 C6Ø5     Ø361Ø          ADD     A,5             ;Pt to start of filename
1FE4 6F       Ø362Ø          LD      L,A
1FE5 ØEØD     Ø363Ø          LD      C,13            ;Init for 15 (-2) chars
1FE7 Ø6Ø8     Ø364Ø          LD      B,8             ;Filename
1FE9 7E       Ø365Ø STUFB1   LD      A,(HL)
1FEA 23       Ø366Ø          INC     HL
1FEB FE2Ø     Ø367Ø          CP      ' '             ;Exit on 1st space
1FED 28Ø7     Ø368Ø          JR      Z,STUFB2
1FEF 12       Ø369Ø          LD      (DE),A          ;Stuff the char
1FFØ 13       Ø37ØØ          INC     DE
1FF1 ØD       Ø371Ø          DEC     C               ;String count down
1FF2 1ØF5     Ø372Ø          DJNZ    STUFB1          ;Field loop
```

```
1FF4 18Ø4     Ø373Ø          JR      STUFB3          ;Bypass ext calculation
1FF6 7D       Ø374Ø STUFB2   LD      A,L             ;Calculate start of
1FF7 8Ø       Ø375Ø          ADD     A,B             ;EXT field in dir record
1FF8 3D       Ø376Ø          DEC     A
1FF9 6F       Ø377Ø          LD      L,A
1FFA 7E       Ø378Ø STUFB3   LD      A,(HL)          ;Display EXT if present
1FFB FE2Ø     Ø379Ø          CP      ' '
1FFD 2812     Ø38ØØ          JR      Z,STUFB5        ;Exit if no EXT
1FFF 3E2F     Ø381Ø          LD      A,'/'           ;Display slash
2ØØ1 12       Ø382Ø          LD      (DE),A          ;Stuff the char
2ØØ2 13       Ø383Ø          INC     DE
2ØØ3 ØD       Ø384Ø          DEC     C               ;Dsplay char countdown
2ØØ4 Ø6Ø3     Ø385Ø          LD      B,3             ;3 chars max for EXT
2ØØ6 7E       Ø386Ø STUFB4   LD      A,(HL)
2ØØ7 23       Ø387Ø          INC     HL
2ØØ8 FE2Ø     Ø388Ø          CP      ' '
2ØØA 28Ø5     Ø389Ø          JR      Z,STUFB5        ;Exit on 1st blank
2ØØC 12       Ø39ØØ          LD      (DE),A          ;  else stuff the char
2ØØD 13       Ø391Ø          INC     DE
2ØØE ØD       Ø392Ø          DEC     C
2ØØF 1ØF5     Ø393Ø          DJNZ    STUFB4          ;Loop 3 chars
2Ø11 3E3A     Ø394Ø STUFB5   LD      A,':'           ;Stuff a drive sep
2Ø13 12       Ø395Ø          LD      (DE),A          ;Reg C already accounted
2Ø14 13       Ø396Ø          INC     DE              ;  for in the init
2Ø15 3EØØ     Ø397Ø DSTDRV   LD      A,Ø             ;P/u drive #
2Ø17 12       Ø398Ø          LD      (DE),A
2Ø18 13       Ø399Ø          INC     DE
2Ø19 3E2Ø     Ø4ØØØ STUFB6   LD      A,' '           ;Stuff a space
2Ø1B 12       Ø4Ø1Ø          LD      (DE),A
2Ø1C 13       Ø4Ø2Ø          INC     DE
2Ø1D ØD       Ø4Ø3Ø          DEC     C               ;Count down
2Ø1E 2ØF9     Ø4Ø4Ø          JR      NZ,STUFB6       ;Display trailing spaces
2Ø2Ø 3EØ3     Ø4Ø5Ø          LD      A,3             ;Stuff the ETX
2Ø22 12       Ø4Ø6Ø          LD      (DE),A
2Ø23 C9       Ø4Ø7Ø          RET
              Ø4Ø8Ø ;
              Ø4Ø9Ø ;        Routine to get the free space info
              Ø41ØØ ;
2Ø24 E5       Ø411Ø SPACEØ   PUSH    HL              ;Save buf start
2Ø25 111ØØØ   Ø412Ø          LD      DE,16           ;Index for space
2Ø28 D5       Ø413Ø          PUSH    DE
2Ø29 19       Ø414Ø          ADD     HL,DE
2Ø2A CD362Ø   Ø415Ø          CALL    SPACE           ;Get the space data
2Ø2D C1       Ø416Ø          POP     BC              ;  name & date
2Ø2E D1       Ø417Ø          POP     DE              ;Now shift in the
2Ø2F 21DØ23   Ø418Ø          LD      HL,DIRBUF$+ØDØH ;  disk name and date
2Ø32 EDBØ     Ø419Ø          LDIR
2Ø34 AF       Ø42ØØ          XOR     A
2Ø35 C9       Ø421Ø          RET
              Ø422Ø ;
2Ø36 CD7418   Ø423Ø SPACE    CALL    @GATRD          ;Read GAT
2Ø39 CØ       Ø424Ø          RET     NZ              ;Ret on GAT read error
2Ø3A FDE5     Ø425Ø          PUSH    IY
2Ø3C CD1E1A   Ø426Ø          CALL    @GTDCT          ;Get DCT vector
2Ø3F EB       Ø427Ø          EX      DE,HL           ;User buf ptr to DE
2Ø4Ø 26ØØ     Ø428Ø          LD      H,Ø             ;P/u highest # cylinder
2Ø42 FD6EØ6   Ø429Ø          LD      L,(IY+6)        ;  & adjust for Ø offset
2Ø45 23       Ø43ØØ          INC     HL
2Ø46 FD7EØ8   Ø431Ø          LD      A,(IY+8)        ;P/u # of sectors/granule
2Ø49 E61F     Ø432Ø          AND     1FH
2Ø4B 3C       Ø433Ø          INC     A               ;Adjust for zero offset
```

```
2Ø4C F5          Ø434Ø           PUSH    AF              ;Save # of sectors/gran
2Ø4D D5          Ø435Ø           PUSH    DE              ;Save user buf ptr
2Ø4E 5F          Ø436Ø           LD      E,A
2Ø4F FD7EØ8      Ø437Ø           LD      A,(IY+8)        ;P/u # of granules/cyl
2Ø52 E6EØ        Ø438Ø           AND     ØEØH
2Ø54 Ø7          Ø439Ø           RLCA                    ;  & shift to bits Ø-2
2Ø55 Ø7          Ø44ØØ           RLCA
2Ø56 Ø7          Ø441Ø           RLCA
2Ø57 3C          Ø442Ø           INC     A               ;Adjust for zero offset
2Ø58 CDØA19      Ø443Ø           CALL    @MUL8           ;Calc # of sectors/cyl
2Ø5B FDCBØ46E    Ø444Ø           BIT     5,(IY+4)        ;Double sided?
2Ø5F 28Ø1        Ø445Ø           JR      Z,$+3           ;Bypass  if one-sided
2Ø61 87          Ø446Ø           ADD     A,A             ;  else double the count
2Ø62 C1          Ø447Ø           POP     BC              ;Rcvr user buf ptr
2Ø63 CD862Ø      Ø448Ø           CALL    DOMUL16         ;Calculate total sectors
2Ø66 23          Ø449Ø           INC     HL              ;Bump to next buf pos
2Ø67 E5          Ø45ØØ           PUSH    HL              ;  & save pointer
2Ø68 21ØØ23      Ø451Ø           LD      HL,DIRBUF$      ;Pt to start of GAT
2Ø6B 11ØØØØ      Ø452Ø           LD      DE,Ø            ;Init gran counter
2Ø6E 3ACC23      Ø453Ø           LD      A,(DIRBUF$+ØCCH)        ;P/u cyl excess
2Ø71 C623        Ø454Ø           ADD     A,35            ;Add base
2Ø73 47          Ø455Ø           LD      B,A             ;Set loop counter
2Ø74 7E          Ø456Ø PUGAT     LD      A,(HL)          ;P/u GAT byte
2Ø75 37          Ø457Ø KEEP7     SCF                     ;Keep bit 7 set
2Ø76 1F          Ø458Ø           RRA                     ;Slide gran bit to carry
2Ø77 38Ø1        Ø459Ø           JR      C,BYTEND?       ;Ignore if in use
2Ø79 13          Ø46ØØ           INC     DE              ;Free, bump gran counter
2Ø7A FEFF        Ø461Ø BYTEND?   CP      ØFFH            ;End of byte?
2Ø7C 2ØF7        Ø462Ø           JR      NZ,KEEP7        ;Loop if not
2Ø7E 2C          Ø463Ø           INC     L               ;Bump GAT byte pointer
2Ø7F 1ØF3        Ø464Ø           DJNZ    PUGAT           ;Loop for # cyls
2Ø81 EB          Ø465Ø           EX      DE,HL           ;# free grans -> HL
2Ø82 C1          Ø466Ø           POP     BC              ;Pop user buf ptr
2Ø83 F1          Ø467Ø           POP     AF              ;Rcvr # of sectors/gran
2Ø84 FDE1        Ø468Ø           POP     IY
2Ø86 CDC9Ø6      Ø469Ø DOMUL16   CALL    @MUL16          ;Calc # of free sectors
2Ø89 6Ø          Ø47ØØ           LD      H,B             ;Cvrt # of free sectors
2Ø8A 55          Ø471Ø           LD      D,L
2Ø8B 69          Ø472Ø           LD      L,C             ;To free space in K by
2Ø8C 5F          Ø473Ø           LD      E,A
2Ø8D 13          Ø474Ø           INC     DE              ;  dividing the # by 4
2Ø8E 13          Ø475Ø           INC     DE              ;Round up adjustment
2Ø8F CB3A        Ø476Ø           SRL     D               ;Divide 16-bit reg by 2
2Ø91 CB1B        Ø477Ø           RR      E
2Ø93 CB3A        Ø478Ø           SRL     D               ;  & divide again
2Ø95 CB1B        Ø479Ø           RR      E
2Ø97 73          Ø48ØØ           LD      (HL),E          ;Stuff the value
2Ø98 23          Ø481Ø           INC     HL
2Ø99 72          Ø482Ø           LD      (HL),D
2Ø9A C9          Ø483Ø           RET
                 Ø484Ø ;
                 Ø485Ø ;         Read the hash index table
                 Ø486Ø ;
2Ø9B 21ØØ23      Ø487Ø HITRD1    LD      HL,DIRBUF$      ;Pt to buffer
2Ø9E C5          Ø488Ø           PUSH    BC
2Ø9F D5          Ø489Ø           PUSH    DE
2ØAØ CDF718      Ø49ØØ           CALL    @DIRCYL         ;Dir cyl to reg D
2ØA3 1EØ1        Ø491Ø           LD      E,1             ;Sector one
2ØA5 CDD818      Ø492Ø           CALL    @RDSSC
2ØA8 D1          Ø493Ø           POP     DE
2ØA9 C1          Ø494Ø           POP     BC
```

```
20AA 3E16      04950          LD      A,22            ;"HIT read error"
20AC C9        04960          RET
               04970 ;
               04980 ;        Routine to locate the address of a module
               04990 ;        DE => pointer to module name
               05000 ;        HL <= address of module start if found
               05010 ;        DE <= address of end-of-module name + 1 if found
               05020 ;        Z  <= if found, else NZ & A = 8
               05030 ;
20AD C5        05040 GTMOD    PUSH    BC              ;Save this reg pair
20AE 0EFF      05050          LD      C,0FFH          ;Init length counter
20B0 D5        05060          PUSH    DE              ;Save name start
20B1 0C        05070 GTM1     INC     C               ;Bump counter
20B2 1A        05080          LD      A,(DE)          ;Search for end-of-name
20B3 13        05090          INC     DE
20B4 FE21      05100          CP      ' '+1
20B6 30F9      05110          JR      NC,GTM1
20B8 D1        05120          POP     DE
               05130 ;
               05140 ;        Start search at system core
               05150 ;
20B9 21F008    05160          LD      HL,@$SYS        ;Pointer to driver start
               05170 ;
               05180 ;        Loop through core searching names
               05190 ;
20BC 7C        05200 GTM2     LD      A,H             ;Are we currently
20BD FE13      05210          CP      @BYTEIO<-8      ;  the driver zone ?
20BF 300D      05220          JR      NC,GTM2A        ;No - check himem
               05230 ;
               05240 ;        In the Driver zone - is it allocated ?
               05250 ;
20C1 C5        05260          PUSH    BC              ;Save BC
20C2 ED4B0602  05270          LD      BC,(DVRHI$)     ;P/u next available
20C6 B7        05280          OR      A               ;  addr in Driver zone.
20C7 E5        05290          PUSH    HL              ;Is this module
20C8 ED42      05300          SBC     HL,BC           ;  accounted for in
20CA E1        05310          POP     HL              ;  the driver zone ?
20CB C1        05320          POP     BC              ;
20CC 3038      05330          JR      NC,GTM8         ;No - get out of d/z
               05340 ;
               05350 ;        Does this module have a legal header ?
               05360 ;
20CE 7E        05370 GTM2A    LD      A,(HL)          ;Ck for "JR xx"
20CF FE18      05380          CP      18H
20D1 202E      05390          JR      NZ,GTM7         ;Exit on non-JR
20D3 E5        05400          PUSH    HL              ;Save pointer to start
20D4 23        05410          INC     HL              ;Advance to length/name
20D5 23        05420          INC     HL
20D6 23        05430          INC     HL
20D7 23        05440          INC     HL
20D8 7E        05450          LD      A,(HL)          ;P/u length field
20D9 E60F      05460          AND     0FH             ;Strip flags
20DB B9        05470          CP      C               ;Lengths match?
20DC 2012      05480          JR      NZ,GTM5
20DE 23        05490          INC     HL              ;Point to start of name
20DF 47        05500          LD      B,A             ;Set loop counter
20E0 D5        05510          PUSH    DE              ;Save user's name pointer
20E1 1A        05520 GTM3     LD      A,(DE)          ;Compare the name strings
20E2 BE        05530          CP      (HL)
20E3 200A      05540          JR      NZ,GTM4         ;Go if no match
20E5 23        05550          INC     HL
```

```
20E6 13       Ø5560          INC     DE
20E7 10F8     Ø5570          DJNZ    GTM3
20E9 EB       Ø5580          EX      DE,HL           ;Name+1 to DE
              Ø5590 ;
              Ø5600 ;        Found a match - exit with info
              Ø5610 ;
20EA E1       Ø5620          POP     HL              ;Keep DE to name end+1
20EB E1       Ø5630          POP     HL              ;Module start address
20EC C1       Ø5640          POP     BC              ;Reg restoral
20ED AF       Ø5650          XOR     A               ;Set Z-flag for return
20EE C9       Ø5660          RET
              Ø5670 ;
              Ø5680 ;        No match - loop to next module
              Ø5690 ;
20EF D1       Ø5700 GTM4     POP     DE
20F0 E1       Ø5710 GTM5     POP     HL
20F1 23       Ø5720          INC     HL              ;Point to last byte used
20F2 23       Ø5730          INC     HL
20F3 7E       Ø5740          LD      A,(HL)          ;P/u low-order
20F4 23       Ø5750          INC     HL
20F5 66       Ø5760          LD      H,(HL)          ;P/u high-order
20F6 6F       Ø5770          LD      L,A
20F7 23       Ø5780 GTM5A    INC     HL              ;Bump to next address
20F8 7C       Ø5790          LD      A,H             ;Ck for wrap to zero
20F9 B5       Ø5800          OR      L
20FA 20CØ     Ø5810          JR      NZ,GTM2         ;Loop if not through
20FC C1       Ø5820 GTM6     POP     BC              ;Restore reg
20FD 3EØ8     Ø5830          LD      A,8             ;Set "device not avail..
20FF B7       Ø5840          OR      A
2100 C9       Ø5850          RET
              Ø5860 ;
              Ø5870 ;        Found non-JR - Advance to high memory?
              Ø5880 ;
2101 7C       Ø5890 GTM7     LD      A,H             ;Past driver core?
2102 FE13     Ø5900          CP      @BYTEIO<-8
2104 30F6     Ø5910          JR      NC,GTM6         ;Exit with "not found"
2106 2AØE04   Ø5920 GTM8     LD      HL,(HIGH$)      ;  else p/u himem pointer
2109 18EC     Ø5930          JR      GTM5A           ;  & hop to it if in use
              Ø5940 ;
              Ø5950 ;        Check a drive for availability
              Ø5960 ;
210B FDE5     Ø5970 CKDRV    PUSH    IY              ;We use IY in disk I/O
210D CD1E1A   Ø5980          CALL    @GTDCT          ;Get driver routine addr
2110 FD7EØØ   Ø5990          LD      A,(IY+Ø)        ;P/u drive vector
2113 FEC3     Ø6000          CP      ØC3H            ;Ck for enabled
2115 C29D21   Ø6010          JP      NZ,CKDR5        ;Bypass if disabled
2118 E5       Ø6020          PUSH    HL
2119 D5       Ø6030          PUSH    DE
211A FD7EØ6   Ø6040          LD      A,(IY+6)        ;Make sure that the current
211D FDBEØ5   Ø6050          CP      (IY+5)          ;  cylinder count is in range
2120 D22921   Ø6060          JP      NC,CKDRV1       ;Go if in range
2123 CDC819   Ø6070          CALL    @RSTOR          ;Restore drive
2126 C2AC21   Ø6080          JP      NZ,CKDR7A       ;Go if error
              Ø6090 ;
2129 FD56Ø5   Ø6100 CKDRV1   LD      D,(IY+5)        ;P/u current track
212C 1EØØ     Ø6110          LD      E,Ø             ;Set for sector Ø
212E CDDØ19   Ø6120          CALL    @SEEK           ;Set track info to FDC
2131 2Ø79     Ø6130          JR      NZ,CKDR7A       ;Go if error
2133 CDD419   Ø6140          CALL    @RSLCT          ;Wait until not busy
2136 2Ø74     Ø6150          JR      NZ,CKDR7A       ;Not there - ret NZ
2138 FDCBØ35E Ø6160          BIT     3,(IY+3)        ;If hard drive, bypass
```

```
213C 2Ø55      Ø617Ø          JR      NZ,CKDR3A       ;  GAT data update
213E FDCBØ466  Ø618Ø          BIT     4,(IY+4)        ;If "ALIEN" by pass
2142 2Ø2C      Ø619Ø          JR      NZ,CKDR2B       ;  test of index pulses
               Ø62ØØ          IF      @MOD4
2144 3AØEØØ    Ø621Ø          LD      A,(FDDINT$)     ;Check 'SMOOTH' state
2147 B7        Ø622Ø          OR      A
2148 3EØ9      Ø623Ø          LD      A,Ø9            ;Set MSB of count down
214A 28Ø3      Ø624Ø          JR      Z,INTRON        ;Go if not SMOOTH
214C CB3F      Ø625Ø          SRL     A               ;Divide the count by two
214E F3        Ø626Ø          DI
               Ø627Ø          ENDIF
               Ø628Ø          IF      @MOD2
               Ø629Ø          LD      A,2Ø
               Ø63ØØ          ENDIF
214F 326121    Ø631Ø INTRON   LD      (CDCNT+1),A     ;Store in 'LD H' instruction
2152 212ØØØ    Ø632Ø          LD      HL,ØØ2ØH        ;Set up count (short)
               Ø633Ø ;
               Ø634Ø ;        Test for diskette in drive & rotating
               Ø635Ø ;
2155 CDAØ21    Ø636Ø CKDR1    CALL    INDEX           ;Test index pulse
2158 2ØFB      Ø637Ø          JR      NZ,CKDR1        ;Jump on index
215A FDCBØ47E  Ø638Ø          BIT     7,(IY+4)        ;Check CKDRV inhibit bit
215E 2Ø1Ø      Ø639Ø          JR      NZ,CKDR2B       ;  if on skip index test
216Ø 26ØØ      Ø64ØØ CDCNT    LD      H,ØØH           ;CKDRV counter (long)
               Ø641Ø          ;Count set from above
2162 CDAØ21    Ø642Ø CKDR2    CALL    INDEX           ;Test index pulse
2165 28FB      Ø643Ø          JR      Z,CKDR2         ;Jump on no index
               Ø644Ø          IF      @MOD4
2167 FB        Ø645Ø          EI                      ;OK for INTs now
               Ø646Ø          ENDIF
2168 212ØØØ    Ø647Ø          LD      HL,ØØ2ØH        ;Index off wait (short)
216B CDAØ21    Ø648Ø CKDR2A   CALL    INDEX
216E 2ØFB      Ø649Ø          JR      NZ,CKDR2A       ;Jump on index
               Ø65ØØ ;
               Ø651Ø ;        Diskette is rotating
               Ø652Ø ;
217Ø F5        Ø653Ø CKDR2B   PUSH    AF              ;Save FDC status
2171 CDF718    Ø654Ø          CALL    @DIRCYL         ;Get directory track in D
2174 21ØØ1D    Ø655Ø          LD      HL,SBUFF$       ;Point to HIT buffer
2177 5D        Ø656Ø          LD      E,L             ;Sector Ø for GAT
2178 CDD818    Ø657Ø          CALL    @RDSSC          ;Read the GAT
217B 2Ø2E      Ø658Ø          JR      NZ,CKDR7        ;Jump on error
217D 2ACC1D    Ø659Ø          LD      HL,(SBUFF$+ØCCH)        ;P/u excess tracks
218Ø 3E22      Ø66ØØ          LD      A,22H           ;Add offset
2182 85        Ø661Ø          ADD     A,L
2183 FD77Ø6    Ø662Ø          LD      (IY+6),A        ;Max track # to DCT
2186 FDCBØ4AE  Ø663Ø          RES     5,(IY+4)        ;Set to side Ø
218A CB6C      Ø664Ø          BIT     5,H             ;Test double sided
218C 28Ø4      Ø665Ø          JR      Z,CKDR3         ;Jump if only single
218E FDCBØ4EE  Ø666Ø          SET     5,(IY+4)        ;Set for side 2
2192 F1        Ø667Ø CKDR3    POP     AF              ;Recover FDC status
2193 Ø7        Ø668Ø CKDR3A   RLCA                    ;Shift write prot to 7
2194 FDB6Ø3    Ø669Ø          OR      (IY+3)          ;Merge Soft WP bit
2197 E68Ø      Ø67ØØ          AND     8ØH             ;Strip all but 7
2199 87        Ø671Ø          ADD     A,A             ;Write prot to carry flg
               Ø672Ø ;
219A           Ø673Ø CKDR4    EQU     $
219A FB        Ø674Ø          EI
219B D1        Ø675Ø          POP     DE
219C E1        Ø676Ø          POP     HL
219D FDE1      Ø677Ø CKDR5    POP     IY
```

```
219F C9        Ø678Ø          RET
               Ø679Ø ;
21AØ 7C        Ø68ØØ INDEX    LD      A,H             ;Check countdown timer
21A1 B5        Ø681Ø          OR      L
21A2 28Ø7      Ø682Ø          JR      Z,CKDR7         ;Err exit if Ø
21A4 2B        Ø683Ø          DEC     HL
21A5 CDD419    Ø684Ø          CALL    @RSLCT          ;Check for index pulse
21A8 CB4F      Ø685Ø          BIT     1,A             ;Test index
21AA C9        Ø686Ø          RET
               Ø687Ø ;
21AB F1        Ø688Ø CKDR7    POP     AF
21AC 3EØ8      Ø689Ø CKDR7A   LD      A,8             ;Set Device not avail
21AE B7        Ø69ØØ          OR      A               ;Set NZ ret
21AF 18E9      Ø691Ø          JR      CKDR4           ;Exit
               Ø692Ø ;
ØØ12           Ø693Ø LILBUF   DS      18
21C3           Ø694Ø LAST     EQU     $
               Ø695Ø          IFGT    $,DIRBUF$
               Ø696Ø          ERR     'Module too big'
               Ø697Ø          ENDIF
23FE           Ø698Ø          ORG     MAXCOR$-2
23FE C3Ø3      Ø699Ø          DW      LAST-SYS12      ;Overlay size
               Ø7ØØØ ;
1EØØ           Ø7Ø1Ø          END     SYS12
```

$A1	Ø3B7	$A2	Ø3B8	$A3	Ø3B9		
$CKEOF	147Ø	@$SYS	Ø8FØ	@@1	ØØØØ		
@@2	ØØØØ	@@3	ØØØØ	@@4	ØØØØ		
@ABORT	1BØ8	@ADTSK	1CDA	@BANK	Ø877		
@BKSP	1486	@BREAK	196F	@BYTEIO	13ØØ		
@CHNIO	Ø689	@CKBRKC	Ø553	@CKDRV	1993		
@CKEOF	158F	@CKTSK	1CF5	@CLOSE	1999		
@CLS	Ø545	@CMNDI	197E	@CMNDR	197B		
@CTL	Ø623	@DATE	Ø7A8	@DBGHK	199F		
@DCINIT	19CØ	@DCRES	19C4	@DCSTAT	19B5		
@DCTBYT	1A2B	@DEBUG	19AØ	@DECHEX	Ø3E1		
@DIRCYL	18F7	@DIRRD	18BB	@DIRWR	18Ø3		
@DIV16	Ø6E3	@DIV8	1927	@DODIR	19AF		
@DOKEY	19A9	@DSP	Ø642	@DSPLY	Ø52D		
@ERROR	1BØF	@EXIT	1BØB	@FEXT	1984		
@FLAGS	196A	@FNAME	199C	@FRENCH	ØØØØ		
@FSPEC	1981	@GATRD	1874	@GATWR	1875		
@GERMAN	ØØØØ	@GET	Ø638	@GTDCB	199Ø		
@GTDCT	1A1E	@GTMOD	19B2	@HDFMT	19E4		
@HEX16	Ø7BD	@HEX8	Ø7C2	@HEXDEC	Ø6F6		
@HIGH$	1948	@HITRD	1897	@HITWR	1898		
@HZ5Ø	ØØØØ	@ICNFG	ØØ86	@INIT	198D		
@INTL	ØØØØ	@IPL	1BF2	@JCL	Ø63Ø		
@KBD	Ø635	@KEY	Ø628	@KEYIN	Ø585		
@KITSK	ØØ89	@KLTSK	1CDØ	@LOAD	1B38		
@LOC	14B3	@LOF	14DE	@LOGER	Ø5Ø3		
@LOGOT	Ø5ØØ	@MOD2	ØØØØ	@MOD4	FFFF		
@MSG	Ø53Ø	@MUL16	Ø6C9	@MUL8	19ØA		
@NMI	ØØ66	@OPEN	198A	@OPREG	ØØ84		
@PARAM	1987	@PAUSE	Ø382	@PEOF	14A2		
@POSN	1434	@PRINT	Ø528	@PRT	Ø63D		
@PUT	Ø645	@RAMDIR	19AC	@RDHDR	19D8		
@RDSEC	19F4	@RDSSC	18D8	@RDTRK	19EØ		
@READ	1513	@REMOVE	19A6	@RENAME	1996		
@REW	149B	@RMTSK	1CD7	@RPTSK	1CEB		
@RREAD	1473	@RSLCT	19D4	@RSTØØ	ØØØØ		
@RST08	ØØØ8	@RST1Ø	ØØ1Ø	@RST18	ØØ18		
@RST2Ø	ØØ2Ø	@RST28	ØØ28	@RST3Ø	ØØ3Ø		
@RST38	ØØ38	@RSTNMI	ØFE9	@RSTOR	19C8		
@RSTREG	Ø68Ø	@RUN	1B1D	@RWRIT	13AD		
@SEEK	19DØ	@SEEKSC	1421	@SKIP	143Ø		
@SLCT	19BC	@SOUND	Ø392	@STEPI	19CC		
@TIME	Ø78D	@USA	FFFF	@VDCTL	ØB99		
@VDCTL3	ØD38	@VER	156Ø	@VRSEC	19DC		
@WEOF	14EC	@WHERE	1979	@WRITE	1531		
@WRSEC	19E8	@WRSSC	19EC	@WRTRK	19FØ		
@_VDCTL	ØD42	ADDR_2_ROWCOL	ØDF1	AFLAG$	ØØ6A		
AUTO?	1FF1	BAR$	Ø2Ø1	BOOTST$	439D		
BREAK?	1C6Ø	BRKVEC$	1C88	BUR$	Ø2ØØ		
BYTEND?	2Ø7A	CASHK$	ØA7B	CDCNT	216Ø		
CFCB$	ØØEØ	CFGFCB$	ØØEØ	CFLAG$	ØØ6C		
CKDR1	2155	CKDR2	2162	CKDR2A	216B		
CKDR2B	217Ø	CKDR3	2192	CKDR3A	2193		
CKDR4	219A	CKDR5	219D	CKDR7	21AB		
CKDR7A	21AC	CKDRV	21ØB	CKDRV1	2129		
CKMOD@	1A7F	CKOPEN@	1568	CKPAGE	1FAF		
CLSBUF	1F48	CONFIG$	2Ø3F	CORE$	Ø3ØØ		
CR	ØØØD	CRTBGN$	F8ØØ	CYL_GRN	16AE		
D@FBYT8	1A26	DATE$	ØØ33	DAYTBL$	Ø4C7		
DBGSV$	ØØAØ	DCBKL$	ØØ31	DCT$	Ø47Ø		

```
DCTBYT8@        1A29 DCTFLD@         1A34 DFLAG$          ØØ6D
DIRBUF$         23ØØ DIRINF1         1E7Ø DIRINFO         1E41
DIS_DO_RAM      Ø846 DOALL           1EBØ DOALL1          1EB7
DOALL2          1EBB DOALL3          1EC8 DOALL4          1ED3
DOALL5          1EEØ DODATA$         ØB94 DODCB$          Ø21Ø
DOMUL16         2Ø86 DO_CONTROL      ØC44 DO_DSPCHAR      ØCB8
DO_INVERT_DIS   ØC8C DO_INVERT_ENA   ØC89 DO_INVERT_OFF   ØC9B
DO_MASK         ØØØØ DO_RET          ØBCB DO_RET1         ØBCC
DO_SCROLL       ØCCE DO_TABS         ØBEA DSKTYP$         Ø4CØ
DSPLYIT         1F8E DSTDRV          2Ø15 DTPMT$          Ø4C2
DVREND$         ØFF4 DVRHI$          Ø2Ø6 EFLAG$          ØØ6E
ENADIS_DO_RAM   Ø817 EXTDBG$         19A4 FDDINT$         ØØØE
FEMSK$          ØØ6F FLGTAB$         ØØ6A GETSTUF         1E7E
GET_@_ROWCOL    ØDAE GTM1            2ØB1 GTM2            2ØBC
GTM2A           2ØCE GTM3            2ØE1 GTM4            2ØEF
GTM5            2ØFØ GTM5A           2ØF7 GTM6            2ØFC
GTM7            21Ø1 GTM8            21Ø6 GTMOD           2ØAD
HERTZ$          Ø75Ø HIGH$           Ø4ØE HITRD1          2Ø9B
HKRES$          1A6C IFLAG$          ØØ72 INBUF$          Ø42Ø
INDEX           21AØ INTIM$          ØØ3C INTMSK$         ØØ3D
INTRON          214F INTVC$          ØØ3E JCLCB$          Ø2Ø3
JDCB$           ØØ24 JFCB$           ØØCØ JLDCB$          Ø23Ø
JRET$           ØØ26 KCK@            Ø7D6 KEEP7           2Ø75
KFLAG$          ØØ74 KIDATA$         Ø8FC KIDCB$          Ø2Ø8
LAST            21C3 LBANK$          Ø2Ø2 LDRV$           ØØ23
LFLAG$          ØØ75 LILBUF          21B1 LNKFCB@         1566
LOW$            ØØ1E LSVC$           ØØØD MAXCOR$         24ØØ
MAXDAY$         Ø4Ø1 MDIR            1EFØ MDIR1           1F2B
MDIR11          1F9E MDIR12          1FC1 MDIR2           1F2F
MDIR3           1F4D MDIR4           1F58 MDIR5           1F64
MINCOR$         3ØØØ MODOUT$         ØØ76 MONTBL$         Ø4DC
NFLAG$          ØØ77 ONESID          1E63 OPREG$          ØØ78
OPREG_SV_AREA   Ø86E OPREG_SV_PTR    Ø835 ORARET@         14DC
OSRLS$          ØØ3B OSVER$          ØØ85 OVRLY$          ØØ69
PAKNAM$         Ø41Ø PAUSE@          Ø382 PCSAVE$         Ø7AF
PDRV$           ØØ1B PHIGH$          ØØ1C PRDCB$          Ø218
PUGAT           2Ø74 PUTA@DE         ØDCD PUT_@           ØDCA
PUT_@_ROWCOL    ØDC6 RAMDIR          1E1Ø RFLAG$          ØØ7B
ROWCOL_2_ADDR   ØDDØ RST38@          1BFF RSTOR$          Ø4C4
RWRIT@          13A2 S1DCB$          Ø238 SBUFF$          1DØØ
SET@EXEC        1A79 SET_SCROLL      ØCF3 SFCB$           ØØ8C
SFLAG$          ØØ7C SIDCB$          Ø22Ø SODCB$          Ø228
SPACE           2Ø36 SPACEØ          2Ø24 SPACE4$         2142
STACK$          Ø38Ø START$          ØØØØ STUFB1          1FE9
STUFB2          1FF6 STUFB3          1FFA STUFB4          2ØØ6
STUFB5          2Ø11 STUFB6          2Ø19 STUFBUF         1FE1
SVCRET$         ØØØB SVCTAB$         Ø1ØØ SYS12           1EØØ
SYSERR$         1B13 TCB$            ØØ4E TFLAG$          ØØ7D
TIME$           ØØ2D TIMER$          ØØ2C TIMSL$          ØØ2B
TIMTSK$         Ø713 TMPMT$          Ø4C3 TRACE_INT       Ø7B1
TSTOPT          1F6A TSTS1           1FCF TSTS2           1FDC
TSTSAM          1FC3 TYPHK$          ØA8F TYPTSK$         ØB26
USTOR$          ØØ13 VFLAG$          ØØ7F WRINT$          ØØ8Ø
ZERO$           Ø4Ø1 ZEROA@          13AØ
ØØØØØ Total errors
```

NOTES:

SYS13 holds the place of an extended command interpreter. It performs no function, but is on the disk to set the proper attributes to any file copied over it to act as an extended interpreter. It also prevents a system hangup if the extended interpreter flag (EFLAG$) is set without the presence of a user file in the SYS13 directory slot.

```
                00100 ;SYS13/ASM - LS-DOS 6.2
0000            00110         TITLE   <SYS13 - LS-DOS 6.2>
                00120 ;
000D            00130 CR      EQU     13
000A            00140 LF      EQU     10
0000            00150 *GET    COPYCOM:3               ;Copyright message
                00010 ; COPYCOM - File for Copyright COMment block
                00020 ;
0000            00030         COM     '<*(C) 1982,83,84 by LSI*>'
                00040 ;
                00160 ;
1E00            00170         ORG     1E00H
                00180 ;
1E00 1820       00190 SYS13   JR      START
1E02 00         00200         DC      32,0              ;Slack
     00 00 00 00 00 00 00 00
     00 00 00 00 00 00 00 00
     00 00 00 00 00 00 00 00
     00 00 00 00 00 00 00 00
                00210 ;
1E22 E670       00220 START   AND     70H               ;Strip bit 7
1E24 FE70       00230         CP      70H               ;Go if 0111,0000
1E26 CA381E     00240         JP      Z,NOCMD           ;  to no * command
1E29 3E65       00250 NOSYS13 LD      A,101             ;Get flags
1E2B EF         00260         RST     40
1E2C FD360400 00270         LD      (IY+'E'-'A'),0    ;Reset ECI flag
1E30 21401E     00280         LD      HL,NXCI$          ;"No ECI present...
1E33 3E0C       00290         LD      A,12              ;Display and log it
1E35 EF         00300         RST     40
1E36 AF         00310         XOR     A
1E37 C9         00320         RET
                00330 ;
1E38 21741E     00340 NOCMD   LD      HL,NOCMD$         ;"No sys13...
1E3B 3E0C       00350         LD      A,12              ;Display and log it
1E3D EF         00360         RST     40
1E3E AF         00370         XOR     A
1E3F C9         00380         RET
                00390 ;
1E40 4E         00400 NXCI$   DB      'No Extended Command Interpreter Present, as SYS13 ',LF,CR
     6F 20 45 78 74 65 6E 64
     65 64 20 43 6F 6D 6D 61
     6E 64 20 49 6E 74 65 72
     70 72 65 74 65 72 20 50
     72 65 73 65 6E 74 2C 20
     61 73 20 53 59 53 31 33
     20 0A 0D
1E74 4E         00410 NOCMD$  DB      'No command <*> present, as SYS13 ',LF,CR
     6F 20 63 6F 6D 6D 61 6E
     64 20 3C 2A 3E 20 70 72
     65 73 65 6E 74 2C 20 61
     73 20 53 59 53 31 33 20
     0A 0D
                00420 ;
                00430 *LIST OFF                         ;500 spare bytes
                00450 *LIST ON
                00460 ;
1E00            00470         END     SYS13
```

```
@@1             ØØØØ @@2             ØØØØ @@3             ØØØØ
@@4             ØØØØ CR              ØØØD LF              ØØØA
NOCMD           1E38 NOCMD$          1E74 NOSYS13         1E29
NXCI$           1E4Ø START           1E22 SYS13           1EØØ

1EØØ is the transfer address
ØØØØØ Total errors
```

NOTES:

NOTES:

NOTES:

NOTES:

NOTES:

NOTES:

```
                ØØ1ØØ ;LDOS6Ø/EQU - Equates from cross reference of Lowcore
ØØØØ            ØØ11Ø          TITLE   <LDOS6Ø/EQU>
                ØØ12Ø ;
Ø8FØ            ØØ13Ø @$SYS    EQU     Ø8FØH
ØØØØ            ØØ14Ø @@1      DEFL    ØØØØH
ØØØØ            ØØ15Ø @@2      DEFL    ØØØØH
ØØØØ            ØØ16Ø @@3      DEFL    ØØØØH
ØØØØ            ØØ17Ø @@4      DEFL    ØØØØH
Ø877            ØØ18Ø @BANK    EQU     Ø877H
13ØØ            ØØ19Ø @BYTEIO  EQU     13ØØH
Ø689            ØØ2ØØ @CHNIO   EQU     Ø689H
Ø553            ØØ21Ø @CKBRKC  EQU     Ø553H
Ø545            ØØ22Ø @CLS     EQU     Ø545H
Ø623            ØØ23Ø @CTL     EQU     Ø623H
Ø7A8            ØØ24Ø @DATE    EQU     Ø7A8H
Ø6E3            ØØ25Ø @DIV16   EQU     Ø6E3H
Ø642            ØØ26Ø @DSP     EQU     Ø642H
Ø52D            ØØ27Ø @DSPLY   EQU     Ø52DH
ØØØØ            ØØ28Ø @FRENCH  EQU     ØØØØH
ØØØØ            ØØ29Ø @GERMAN  EQU     ØØØØH
Ø638            ØØ3ØØ @GET     EQU     Ø638H
Ø7BD            ØØ31Ø @HEX16   EQU     Ø7BDH
Ø7C2            ØØ32Ø @HEX8    EQU     Ø7C2H
Ø6F6            ØØ33Ø @HEXDEC  EQU     Ø6F6H
ØØØØ            ØØ34Ø @HZ5Ø    EQU     ØØØØH
ØØØØ            ØØ35Ø @INTL    EQU     ØØØØH
Ø63Ø            ØØ36Ø @JCL     EQU     Ø63ØH
Ø635            ØØ37Ø @KBD     EQU     Ø635H
Ø628            ØØ38Ø @KEY     EQU     Ø628H
Ø585            ØØ39Ø @KEYIN   EQU     Ø585H
ØØ89            ØØ4ØØ @KITSK   EQU     ØØ89H
Ø5Ø3            ØØ41Ø @LOGER   EQU     Ø5Ø3H
Ø5ØØ            ØØ42Ø @LOGOT   EQU     Ø5ØØH
ØØØØ            ØØ43Ø @MOD2    EQU     ØØØØH
FFFF            ØØ44Ø @MOD4    EQU     ØFFFFH
Ø53Ø            ØØ45Ø @MSG     EQU     Ø53ØH
Ø6C9            ØØ46Ø @MUL16   EQU     Ø6C9H
ØØ84            ØØ47Ø @OPREG   EQU     ØØ84H
Ø528            ØØ48Ø @PRINT   EQU     Ø528H
Ø63D            ØØ49Ø @PRT     EQU     Ø63DH
Ø645            ØØ5ØØ @PUT     EQU     Ø645H
ØFE9            ØØ51Ø @RSTNMI  EQU     ØFE9H
Ø68Ø            ØØ52Ø @RSTREG  EQU     Ø68ØH
Ø78D            ØØ53Ø @TIME    EQU     Ø78DH
FFFF            ØØ54Ø @USA     EQU     ØFFFFH
ØB99            ØØ55Ø @VDCTL   EQU     ØB99H
ØD38            ØØ56Ø @VDCTL3  EQU     ØD38H
ØD42            ØØ57Ø @_VDCTL  EQU     ØD42H
ØDF1            ØØ58Ø ADDR_2_ROWCOL    EQU     ØDF1H
Ø2Ø1            ØØ59Ø BAR$     EQU     Ø2Ø1H
439D            ØØ6ØØ BOOTST$  EQU     439DH
Ø2ØØ            ØØ61Ø BUR$     EQU     Ø2ØØH
ØA7B            ØØ62Ø CASHK$   EQU     ØA7BH
ØØ6C            ØØ63Ø CFLAG$   EQU     ØØ6CH
Ø3ØØ            ØØ64Ø CORE$    DEFL    Ø3ØØH
F8ØØ            ØØ65Ø CRTBGN$  EQU     ØF8ØØH
ØØ33            ØØ66Ø DATE$    EQU     ØØ33H
Ø4C7            ØØ67Ø DAYTBL$  EQU     Ø4C7H
ØØ31            ØØ68Ø DCBKL$   EQU     ØØ31H
Ø47Ø            ØØ69Ø DCT$     EQU     Ø47ØH
ØØ6D            ØØ7ØØ DFLAG$   EQU     ØØ6DH
```

```
0846          00710 DIS_DO_RAM      EQU     0846H
0B94          00720 DODATA$ EQU     0B94H
0210          00730 DODCB$  EQU     0210H
0C44          00740 DO_CONTROL      EQU     0C44H
0CB8          00750 DO_DSPCHAR      EQU     0CB8H
0C8C          00760 DO_INVERT_DIS   EQU     0C8CH
0C89          00770 DO_INVERT_ENA   EQU     0C89H
0C9B          00780 DO_INVERT_OFF   EQU     0C9BH
0000          00790 DO_MASK EQU     0000H
0BCB          00800 DO_RET  EQU     0BCBH
0BCC          00810 DO_RET1 EQU     0BCCH
0CCE          00820 DO_SCROLL       EQU     0CCEH
0BEA          00830 DO_TABS EQU     0BEAH
04C0          00840 DSKTYP$ EQU     04C0H
04C2          00850 DTPMT$  EQU     04C2H
0FF4          00860 DVREND$ EQU     0FF4H
0206          00870 DVRHI$  EQU     0206H
0817          00880 ENADIS_DO_RAM   EQU     0817H
000E          00890 FDDINT$ EQU     000EH
006A          00900 FLGTAB$ EQU     006AH
0DAE          00910 GET_@_ROWCOL    EQU     0DAEH
0750          00920 HERTZ$  EQU     0750H
040E          00930 HIGH$   EQU     040EH
0072          00940 IFLAG$  EQU     0072H
0420          00950 INBUF$  EQU     0420H
003E          00960 INTVC$  EQU     003EH
0203          00970 JCLCB$  EQU     0203H
0230          00980 JLDCB$  EQU     0230H
07D6          00990 KCK@    EQU     07D6H
0074          01000 KFLAG$  EQU     0074H
08FC          01010 KIDATA$ EQU     08FCH
0208          01020 KIDCB$  EQU     0208H
0202          01030 LBANK$  EQU     0202H
0401          01040 MAXDAY$ EQU     0401H
0076          01050 MODOUT$ EQU     0076H
04DC          01060 MONTBL$ EQU     04DCH
0077          01070 NFLAG$  EQU     0077H
0078          01080 OPREG$  EQU     0078H
086E          01090 OPREG_SV_AREA   EQU     086EH
0835          01100 OPREG_SV_PTR    EQU     0835H
0410          01110 PAKNAM$ EQU     0410H
0382          01120 PAUSE@  EQU     0382H
07AF          01130 PCSAVE$ EQU     07AFH
001B          01140 PDRV$   EQU     001BH
0218          01150 PRDCB$  EQU     0218H
0DCD          01160 PUTA@DE EQU     0DCDH
0DCA          01170 PUT_@   EQU     0DCAH
0DC6          01180 PUT_@_ROWCOL    EQU     0DC6H
007B          01190 RFLAG$  EQU     007BH
0DD0          01200 ROWCOL_2_ADDR   EQU     0DD0H
04C4          01210 RSTOR$  EQU     04C4H
0238          01220 S1DCB$  EQU     0238H
0CF3          01230 SET_SCROLL      EQU     0CF3H
007C          01240 SFLAG$  EQU     007CH
0220          01250 SIDCB$  EQU     0220H
0228          01260 SODCB$  EQU     0228H
0380          01270 STACK$  EQU     0380H
0000          01280 START$  EQU     0000H
002D          01290 TIME$   EQU     002DH
002C          01300 TIMER$  EQU     002CH
002B          01310 TIMSL$  EQU     002BH
```

```
Ø713          Ø132Ø TIMTSK$ EQU     Ø713H
Ø4C3          Ø133Ø TMPMT$  EQU     Ø4C3H
Ø7B1          Ø134Ø TRACE_INT       EQU     Ø7B1H
ØA8F          Ø135Ø TYPHK$  EQU     ØA8FH
ØB26          Ø136Ø TYPTSK$ EQU     ØB26H
ØØ7F          Ø137Ø VFLAG$  EQU     ØØ7FH
Ø4Ø1          Ø138Ø ZERO$   EQU     Ø4Ø1H
              Ø139Ø ;
No end statement
ØØØØØ Total errors
```

```
                ØØ1ØØ ;SYSØ/EQU - Equates from cross reference of Sysres
ØØØØ            ØØ11Ø          TITLE   <SYSØ/EQU>
                ØØ12Ø ;
Ø3B7            ØØ13Ø $A1      EQU     Ø3B7H
Ø3B8            ØØ14Ø $A2      EQU     Ø3B8H
Ø3B9            ØØ15Ø $A3      EQU     Ø3B9H
147Ø            ØØ16Ø $CKEOF   EQU     147ØH
Ø8FØ            ØØ17Ø @$SYS    EQU     Ø8FØH
ØØØØ            ØØ18Ø @@1      DEFL    ØØØØH
ØØØØ            ØØ19Ø @@1      DEFL    ØØØØH
ØØØØ            ØØ2ØØ @@2      DEFL    ØØØØH
ØØØØ            ØØ21Ø @@2      DEFL    ØØØØH
ØØØØ            ØØ22Ø @@3      DEFL    ØØØØH
ØØØØ            ØØ23Ø @@3      DEFL    ØØØØH
ØØØØ            ØØ24Ø @@4      DEFL    ØØØØH
ØØØØ            ØØ25Ø @@4      DEFL    ØØØØH
1BØ8            ØØ26Ø @ABORT   EQU     1BØ8H
1CDA            ØØ27Ø @ADTSK   EQU     1CDAH
Ø877            ØØ28Ø @BANK    EQU     Ø877H
1486            ØØ29Ø @BKSP    EQU     1486H
196F            ØØ3ØØ @BREAK   EQU     196FH
13ØØ            ØØ31Ø @BYTEIO  EQU     13ØØH
Ø689            ØØ32Ø @CHNIO   EQU     Ø689H
Ø553            ØØ33Ø @CKBRKC  EQU     Ø553H
1993            ØØ34Ø @CKDRV   EQU     1993H
158F            ØØ35Ø @CKEOF   EQU     158FH
1CF5            ØØ36Ø @CKTSK   EQU     1CF5H
1999            ØØ37Ø @CLOSE   EQU     1999H
Ø545            ØØ38Ø @CLS     EQU     Ø545H
197E            ØØ39Ø @CMNDI   EQU     197EH
197B            ØØ4ØØ @CMNDR   EQU     197BH
Ø623            ØØ41Ø @CTL     EQU     Ø623H
Ø7A8            ØØ42Ø @DATE    EQU     Ø7A8H
199F            ØØ43Ø @DBGHK   EQU     199FH
19CØ            ØØ44Ø @DCINIT  EQU     19CØH
19C4            ØØ45Ø @DCRES   EQU     19C4H
19B5            ØØ46Ø @DCSTAT  EQU     19B5H
1A2B            ØØ47Ø @DCTBYT  EQU     1A2BH
19AØ            ØØ48Ø @DEBUG   EQU     19AØH
Ø3E1            ØØ49Ø @DECHEX  EQU     Ø3E1H
18F7            ØØ5ØØ @DIRCYL  EQU     18F7H
18BB            ØØ51Ø @DIRRD   EQU     18BBH
18Ø3            ØØ52Ø @DIRWR   EQU     18Ø3H
Ø6E3            ØØ53Ø @DIV16   EQU     Ø6E3H
1927            ØØ54Ø @DIV8    EQU     1927H
19AF            ØØ55Ø @DODIR   EQU     19AFH
19A9            ØØ56Ø @DOKEY   EQU     19A9H
Ø642            ØØ57Ø @DSP     EQU     Ø642H
Ø52D            ØØ58Ø @DSPLY   EQU     Ø52DH
1BØF            ØØ59Ø @ERROR   EQU     1BØFH
1BØB            ØØ6ØØ @EXIT    EQU     1BØBH
1984            ØØ61Ø @FEXT    EQU     1984H
196A            ØØ62Ø @FLAGS   EQU     196AH
199C            ØØ63Ø @FNAME   EQU     199CH
ØØØØ            ØØ64Ø @FRENCH  EQU     ØØØØH
1981            ØØ65Ø @FSPEC   EQU     1981H
1874            ØØ66Ø @GATRD   EQU     1874H
1875            ØØ67Ø @GATWR   EQU     1875H
ØØØØ            ØØ68Ø @GERMAN  EQU     ØØØØH
Ø638            ØØ69Ø @GET     EQU     Ø638H
199Ø            ØØ7ØØ @GTDCB   EQU     199ØH
```

```
1A1E  ØØ71Ø @GTDCT  EQU  1A1EH
19B2  ØØ72Ø @GTMOD  EQU  19B2H
19E4  ØØ73Ø @HDFMT  EQU  19E4H
Ø7BD  ØØ74Ø @HEX16  EQU  Ø7BDH
Ø7C2  ØØ75Ø @HEX8   EQU  Ø7C2H
Ø6F6  ØØ76Ø @HEXDEC EQU  Ø6F6H
1948  ØØ77Ø @HIGH$  EQU  1948H
1897  ØØ78Ø @HITRD  EQU  1897H
1898  ØØ79Ø @HITWR  EQU  1898H
ØØØØ  ØØ8ØØ @HZ5Ø   EQU  ØØØØH
ØØ86  ØØ81Ø @ICNFG  EQU  ØØ86H
198D  ØØ82Ø @INIT   EQU  198DH
ØØØØ  ØØ83Ø @INTL   EQU  ØØØØH
1BF2  ØØ84Ø @IPL    EQU  1BF2H
Ø63Ø  ØØ85Ø @JCL    EQU  Ø63ØH
Ø635  ØØ86Ø @KBD    EQU  Ø635H
Ø628  ØØ87Ø @KEY    EQU  Ø628H
Ø585  ØØ88Ø @KEYIN  EQU  Ø585H
ØØ89  ØØ89Ø @KITSK  EQU  ØØ89H
ØØ89  ØØ9ØØ @KITSK  EQU  ØØ89H
1CDØ  ØØ91Ø @KLTSK  EQU  1CDØH
1B38  ØØ92Ø @LOAD   EQU  1B38H
14B3  ØØ93Ø @LOC    EQU  14B3H
14DE  ØØ94Ø @LOF    EQU  14DEH
Ø5Ø3  ØØ95Ø @LOGER  EQU  Ø5Ø3H
Ø5ØØ  ØØ96Ø @LOGOT  EQU  Ø5ØØH
ØØØØ  ØØ97Ø @MOD2   EQU  ØØØØH
FFFF  ØØ98Ø @MOD4   EQU  ØFFFFH
Ø53Ø  ØØ99Ø @MSG    EQU  Ø53ØH
Ø6C9  Ø1ØØØ @MUL16  EQU  Ø6C9H
19ØA  Ø1Ø1Ø @MUL8   EQU  19ØAH
ØØ66  Ø1Ø2Ø @NMI    EQU  ØØ66H
198A  Ø1Ø3Ø @OPEN   EQU  198AH
ØØ84  Ø1Ø4Ø @OPREG  EQU  ØØ84H
1987  Ø1Ø5Ø @PARAM  EQU  1987H
Ø382  Ø1Ø6Ø @PAUSE  EQU  Ø382H
14A2  Ø1Ø7Ø @PEOF   EQU  14A2H
1434  Ø1Ø8Ø @POSN   EQU  1434H
Ø528  Ø1Ø9Ø @PRINT  EQU  Ø528H
Ø63D  Ø11ØØ @PRT    EQU  Ø63DH
Ø645  Ø111Ø @PUT    EQU  Ø645H
19AC  Ø112Ø @RAMDIR EQU  19ACH
19D8  Ø113Ø @RDHDR  EQU  19D8H
19F4  Ø114Ø @RDSEC  EQU  19F4H
18D8  Ø115Ø @RDSSC  EQU  18D8H
19EØ  Ø116Ø @RDTRK  EQU  19EØH
1513  Ø117Ø @READ   EQU  1513H
19A6  Ø118Ø @REMOVE EQU  19A6H
1996  Ø119Ø @RENAME EQU  1996H
149B  Ø12ØØ @REW    EQU  149BH
1CD7  Ø121Ø @RMTSK  EQU  1CD7H
1CEB  Ø122Ø @RPTSK  EQU  1CEBH
1473  Ø123Ø @RREAD  EQU  1473H
19D4  Ø124Ø @RSLCT  EQU  19D4H
ØØØØ  Ø125Ø @RSTØØ  EQU  ØØØØH
ØØØ8  Ø126Ø @RSTØ8  EQU  ØØØ8H
ØØ1Ø  Ø127Ø @RST1Ø  EQU  ØØ1ØH
ØØ18  Ø128Ø @RST18  EQU  ØØ18H
ØØ2Ø  Ø129Ø @RST2Ø  EQU  ØØ2ØH
ØØ28  Ø13ØØ @RST28  EQU  ØØ28H
ØØ3Ø  Ø131Ø @RST3Ø  EQU  ØØ3ØH
```

```
ØØ38          Ø132Ø @RST38  EQU     ØØ38H
ØFE9          Ø133Ø @RSTNMI EQU     ØFE9H
19C8          Ø134Ø @RSTOR  EQU     19C8H
Ø68Ø          Ø135Ø @RSTREG EQU     Ø68ØH
1B1D          Ø136Ø @RUN    EQU     1B1DH
13AD          Ø137Ø @RWRIT  EQU     13ADH
19DØ          Ø138Ø @SEEK   EQU     19DØH
1421          Ø139Ø @SEEKSC EQU     1421H
143Ø          Ø14ØØ @SKIP   EQU     143ØH
19BC          Ø141Ø @SLCT   EQU     19BCH
Ø392          Ø142Ø @SOUND  EQU     Ø392H
19CC          Ø143Ø @STEPI  EQU     19CCH
Ø78D          Ø144Ø @TIME   EQU     Ø78DH
FFFF          Ø145Ø @USA    EQU     ØFFFFH
ØB99          Ø146Ø @VDCTL  EQU     ØB99H
ØD38          Ø147Ø @VDCTL3 EQU     ØD38H
156Ø          Ø148Ø @VER    EQU     156ØH
19DC          Ø149Ø @VRSEC  EQU     19DCH
14EC          Ø15ØØ @WEOF   EQU     14ECH
1979          Ø151Ø @WHERE  EQU     1979H
1531          Ø152Ø @WRITE  EQU     1531H
19E8          Ø153Ø @WRSEC  EQU     19E8H
19EC          Ø154Ø @WRSSC  EQU     19ECH
19FØ          Ø155Ø @WRTRK  EQU     19FØH
ØD42          Ø156Ø @_VDCTL EQU     ØD42H
ØDF1          Ø157Ø ADDR_2_ROWCOL   EQU     ØDF1H
ØØ6A          Ø158Ø AFLAG$  EQU     ØØ6AH
1FF1          Ø159Ø AUTO?   EQU     1FF1H
Ø2Ø1          Ø16ØØ BAR$    EQU     Ø2Ø1H
439D          Ø161Ø BOOTST$ EQU     439DH
1C6Ø          Ø162Ø BREAK?  EQU     1C6ØH
1C88          Ø163Ø BRKVEC$ EQU     1C88H
Ø2ØØ          Ø164Ø BUR$    EQU     Ø2ØØH
ØA7B          Ø165Ø CASHK$  EQU     ØA7BH
ØØEØ          Ø166Ø CFCB$   EQU     ØØEØH
ØØEØ          Ø167Ø CFGFCB$ EQU     ØØEØH
ØØ6C          Ø168Ø CFLAG$  EQU     ØØ6CH
ØØ6C          Ø169Ø CFLAG$  EQU     ØØ6CH
1A7F          Ø17ØØ CKMOD@  EQU     1A7FH
1568          Ø171Ø CKOPEN@ EQU     1568H
2Ø3F          Ø172Ø CONFIG$ EQU     2Ø3FH
1CFF          Ø173Ø CORE$   DEFL    1CFFH
1BFF          Ø174Ø CORE$   DEFL    1BFFH
1948          Ø175Ø CORE$   DEFL    1948H
1948          Ø176Ø CORE$   DEFL    1948H
Ø3ØØ          Ø177Ø CORE$   DEFL    Ø3ØØH
F8ØØ          Ø178Ø CRTBGN$ EQU     ØF8ØØH
16AE          Ø179Ø CYL_GRN EQU     16AEH
1A26          Ø18ØØ D@FBYT8 EQU     1A26H
ØØ33          Ø181Ø DATE$   EQU     ØØ33H
ØØ33          Ø182Ø DATE$   EQU     ØØ33H
Ø4C7          Ø183Ø DAYTBL$ EQU     Ø4C7H
ØØAØ          Ø184Ø DBGSV$  EQU     ØØAØH
ØØ31          Ø185Ø DCBKL$  EQU     ØØ31H
Ø47Ø          Ø186Ø DCT$    EQU     Ø47ØH
1A29          Ø187Ø DCTBYT8@        EQU     1A29H
1A34          Ø188Ø DCTFLD@ EQU     1A34H
ØØ6D          Ø189Ø DFLAG$  EQU     ØØ6DH
ØØ6D          Ø19ØØ DFLAG$  EQU     ØØ6DH
23ØØ          Ø191Ø DIRBUF$ EQU     23ØØH
Ø846          Ø192Ø DIS_DO_RAM      EQU     Ø846H
```

```
ØB94          Ø193Ø DODATA$ EQU     ØB94H
Ø21Ø          Ø194Ø DODCB$  EQU     Ø21ØH
ØC44          Ø195Ø DO_CONTROL      EQU     ØC44H
ØCB8          Ø196Ø DO_DSPCHAR      EQU     ØCB8H
ØC8C          Ø197Ø DO_INVERT_DIS   EQU     ØC8CH
ØC89          Ø198Ø DO_INVERT_ENA   EQU     ØC89H
ØC9B          Ø199Ø DO_INVERT_OFF   EQU     ØC9BH
ØØØØ          Ø2ØØØ DO_MASK EQU     ØØØØH
ØBCB          Ø2Ø1Ø DO_RET  EQU     ØBCBH
ØBCC          Ø2Ø2Ø DO_RET1 EQU     ØBCCH
ØCCE          Ø2Ø3Ø DO_SCROLL       EQU     ØCCEH
ØBEA          Ø2Ø4Ø DO_TABS EQU     ØBEAH
Ø4CØ          Ø2Ø5Ø DSKTYP$ EQU     Ø4CØH
Ø4C2          Ø2Ø6Ø DTPMT$  EQU     Ø4C2H
ØFF4          Ø2Ø7Ø DVREND$ EQU     ØFF4H
Ø2Ø6          Ø2Ø8Ø DVRHI$  EQU     Ø2Ø6H
ØØ6E          Ø2Ø9Ø EFLAG$  EQU     ØØ6EH
Ø817          Ø21ØØ ENADIS_DO_RAM   EQU     Ø817H
19A4          Ø211Ø EXTDBG$ EQU     19A4H
ØØØE          Ø212Ø FDDINT$ EQU     ØØØEH
ØØØE          Ø213Ø FDDINT$ EQU     ØØØEH
ØØ6F          Ø214Ø FEMSK$  EQU     ØØ6FH
ØØ6A          Ø215Ø FLGTAB$ EQU     ØØ6AH
ØØ6A          Ø216Ø FLGTAB$ EQU     ØØ6AH
ØDAE          Ø217Ø GET_@_ROWCOL    EQU     ØDAEH
Ø75Ø          Ø218Ø HERTZ$  EQU     Ø75ØH
Ø4ØE          Ø219Ø HIGH$   EQU     Ø4ØEH
1A6C          Ø22ØØ HKRES$  EQU     1A6CH
ØØ72          Ø221Ø IFLAG$  EQU     ØØ72H
ØØ72          Ø222Ø IFLAG$  EQU     ØØ72H
Ø42Ø          Ø223Ø INBUF$  EQU     Ø42ØH
ØØ3C          Ø224Ø INTIM$  EQU     ØØ3CH
ØØ3D          Ø225Ø INTMSK$ EQU     ØØ3DH
ØØ3E          Ø226Ø INTVC$  EQU     ØØ3EH
ØØ3E          Ø227Ø INTVC$  EQU     ØØ3EH
Ø2Ø3          Ø228Ø JCLCB$  EQU     Ø2Ø3H
ØØ24          Ø229Ø JDCB$   EQU     ØØ24H
ØØCØ          Ø23ØØ JFCB$   EQU     ØØCØH
Ø23Ø          Ø231Ø JLDCB$  EQU     Ø23ØH
ØØ26          Ø232Ø JRET$   EQU     ØØ26H
Ø7D6          Ø233Ø KCK@    EQU     Ø7D6H
ØØ74          Ø234Ø KFLAG$  EQU     ØØ74H
ØØ74          Ø235Ø KFLAG$  EQU     ØØ74H
Ø8FC          Ø236Ø KIDATA$ EQU     Ø8FCH
Ø2Ø8          Ø237Ø KIDCB$  EQU     Ø2Ø8H
Ø2Ø2          Ø238Ø LBANK$  EQU     Ø2Ø2H
ØØ23          Ø239Ø LDRV$   EQU     ØØ23H
ØØ75          Ø24ØØ LFLAG$  EQU     ØØ75H
1566          Ø241Ø LNKFCB@ EQU     1566H
ØØ1E          Ø242Ø LOW$    EQU     ØØ1EH
ØØØD          Ø243Ø LSVC$   EQU     ØØØDH
24ØØ          Ø244Ø MAXCOR$ EQU     24ØØH
Ø4Ø1          Ø245Ø MAXDAY$ EQU     Ø4Ø1H
3ØØØ          Ø246Ø MINCOR$ EQU     3ØØØH
ØØ76          Ø247Ø MODOUT$ EQU     ØØ76H
ØØ76          Ø248Ø MODOUT$ EQU     ØØ76H
Ø4DC          Ø249Ø MONTBL$ EQU     Ø4DCH
ØØ77          Ø25ØØ NFLAG$  EQU     ØØ77H
ØØ78          Ø251Ø OPREG$  EQU     ØØ78H
ØØ78          Ø252Ø OPREG$  EQU     ØØ78H
Ø86E          Ø253Ø OPREG_SV_AREA   EQU     Ø86EH
```

```
0835          02540 OPREG_SV_PTR    EQU     0835H
14DC          02550 ORARET@ EQU     14DCH
003B          02560 OSRLS$  EQU     003BH
0085          02570 OSVER$  EQU     0085H
0069          02580 OVRLY$  EQU     0069H
041Ø          02590 PAKNAM$ EQU     041ØH
0382          02600 PAUSE@  EQU     0382H
07AF          02610 PCSAVE$ EQU     07AFH
001B          02620 PDRV$   EQU     001BH
001B          02630 PDRV$   EQU     001BH
001C          02640 PHIGH$  EQU     001CH
0218          02650 PRDCB$  EQU     0218H
0DCD          02660 PUTA@DE EQU     0DCDH
0DCA          02670 PUT_@   EQU     0DCAH
0DC6          02680 PUT_@_ROWCOL    EQU     0DC6H
007B          02690 RFLAG$  EQU     007BH
007B          02700 RFLAG$  EQU     007BH
0DDØ          02710 ROWCOL_2_ADDR   EQU     0DDØH
1BFF          02720 RST38@  EQU     1BFFH
04C4          02730 RSTOR$  EQU     04C4H
13A2          02740 RWRIT@  EQU     13A2H
0238          02750 S1DCB$  EQU     0238H
1DØØ          02760 SBUFF$  EQU     1DØØH
1A79          02770 SET@EXEC        EQU     1A79H
0CF3          02780 SET_SCROLL      EQU     0CF3H
008C          02790 SFCB$   EQU     008CH
007C          02800 SFLAG$  EQU     007CH
007C          02810 SFLAG$  EQU     007CH
0220          02820 SIDCB$  EQU     0220H
0228          02830 SODCB$  EQU     0228H
2142          02840 SPACE4$ EQU     2142H
038Ø          02850 STACK$  EQU     038ØH
0000          02860 START$  EQU     0000H
0000          02870 START$  EQU     0000H
000B          02880 SVCRET$ EQU     000BH
01ØØ          02890 SVCTAB$ EQU     01ØØH
1B13          02900 SYSERR$ EQU     1B13H
004E          02910 TCB$    EQU     004EH
007D          02920 TFLAG$  EQU     007DH
002D          02930 TIME$   EQU     002DH
002D          02940 TIME$   EQU     002DH
002C          02950 TIMER$  EQU     002CH
002C          02960 TIMER$  EQU     002CH
002B          02970 TIMSL$  EQU     002BH
002B          02980 TIMSL$  EQU     002BH
0713          02990 TIMTSK$ EQU     0713H
04C3          03000 TMPMT$  EQU     04C3H
07B1          03010 TRACE_INT       EQU     07B1H
0A8F          03020 TYPHK$  EQU     0A8FH
0B26          03030 TYPTSK$ EQU     0B26H
0013          03040 USTOR$  EQU     0013H
007F          03050 VFLAG$  EQU     007FH
007F          03060 VFLAG$  EQU     007FH
0080          03070 WRINT$  EQU     0080H
0401          03080 ZERO$   EQU     0401H
13AØ          03090 ZEROA@  EQU     13AØH
No end statement
ØØØØØ Total errors
```

```
                00100 ;SYS5/EQU - Equates from cross reference of SYS5
0000            00110         TITLE   <SYS5/EQU>
                00120 ;
1E32            00130 $?1     EQU     1E32H
1F1D            00140 $?10    EQU     1F1DH
1F2E            00150 $?11    EQU     1F2EH
1F38            00160 $?12    EQU     1F38H
1F8F            00170 $?13    EQU     1F8FH
1F9B            00180 $?14    EQU     1F9BH
1F9F            00190 $?15    EQU     1F9FH
1FA4            00200 $?16    EQU     1FA4H
1FC5            00210 $?17    EQU     1FC5H
1FDF            00220 $?18    EQU     1FDFH
200F            00230 $?19    EQU     200FH
1E37            00240 $?2     EQU     1E37H
2057            00250 $?20    EQU     2057H
205C            00260 $?21    EQU     205CH
2061            00270 $?22    EQU     2061H
2062            00280 $?23    EQU     2062H
2066            00290 $?24    EQU     2066H
20A6            00300 $?25    EQU     20A6H
20A9            00310 $?26    EQU     20A9H
20AA            00320 $?27    EQU     20AAH
20B7            00330 $?28    EQU     20B7H
20F1            00340 $?28A   EQU     20F1H
20F6            00350 $?29    EQU     20F6H
1E49            00360 $?3     EQU     1E49H
20F9            00370 $?30    EQU     20F9H
20FC            00380 $?31    EQU     20FCH
2102            00390 $?32    EQU     2102H
210B            00400 $?33    EQU     210BH
2117            00410 $?34    EQU     2117H
211A            00420 $?35    EQU     211AH
2180            00430 $?36    EQU     2180H
218E            00440 $?37    EQU     218EH
219A            00450 $?38    EQU     219AH
219C            00460 $?39    EQU     219CH
1EB4            00470 $?4     EQU     1EB4H
21BF            00480 $?40    EQU     21BFH
21C3            00490 $?41    EQU     21C3H
21C7            00500 $?42    EQU     21C7H
21CA            00510 $?43    EQU     21CAH
21E1            00520 $?44    EQU     21E1H
21EB            00530 $?45    EQU     21EBH
2223            00540 $?46    EQU     2223H
222B            00550 $?47    EQU     222BH
223B            00560 $?48    EQU     223BH
1EC4            00570 $?5     EQU     1EC4H
1EC5            00580 $?6     EQU     1EC5H
1EEE            00590 $?8     EQU     1EEEH
1F16            00600 $?9     EQU     1F16H
03B7            00610 $A1     EQU     03B7H
03B8            00620 $A2     EQU     03B8H
03B9            00630 $A3     EQU     03B9H
1470            00640 $CKEOF  EQU     1470H
08F0            00650 @$SYS   EQU     08F0H
0000            00660 @@1     DEFL    0000H
0000            00670 @@1     DEFL    0000H
0000            00680 @@1     DEFL    0000H
0000            00690 @@2     DEFL    0000H
0000            00700 @@2     DEFL    0000H
```

```
0000          00710 @@2      DEFL    0000H
0000          00720 @@3      DEFL    0000H
0000          00730 @@3      DEFL    0000H
0000          00740 @@3      DEFL    0000H
0000          00750 @@4      DEFL    0000H
0000          00760 @@4      DEFL    0000H
0000          00770 @@4      DEFL    0000H
1B08          00780 @ABORT   EQU     1B08H
1CDA          00790 @ADTSK   EQU     1CDAH
0877          00800 @BANK    EQU     0877H
1486          00810 @BKSP    EQU     1486H
196F          00820 @BREAK   EQU     196FH
1300          00830 @BYTEIO  EQU     1300H
0689          00840 @CHNIO   EQU     0689H
0553          00850 @CKBRKC  EQU     0553H
1993          00860 @CKDRV   EQU     1993H
158F          00870 @CKEOF   EQU     158FH
1CF5          00880 @CKTSK   EQU     1CF5H
1999          00890 @CLOSE   EQU     1999H
0545          00900 @CLS     EQU     0545H
197E          00910 @CMNDI   EQU     197EH
197B          00920 @CMNDR   EQU     197BH
0623          00930 @CTL     EQU     0623H
07A8          00940 @DATE    EQU     07A8H
199F          00950 @DBGHK   EQU     199FH
19C0          00960 @DCINIT  EQU     19C0H
19C4          00970 @DCRES   EQU     19C4H
19B5          00980 @DCSTAT  EQU     19B5H
1A2B          00990 @DCTBYT  EQU     1A2BH
19A0          01000 @DEBUG   EQU     19A0H
03E1          01010 @DECHEX  EQU     03E1H
18F7          01020 @DIRCYL  EQU     18F7H
18BB          01030 @DIRRD   EQU     18BBH
1803          01040 @DIRWR   EQU     1803H
06E3          01050 @DIV16   EQU     06E3H
1927          01060 @DIV8    EQU     1927H
19AF          01070 @DODIR   EQU     19AFH
19A9          01080 @DOKEY   EQU     19A9H
0642          01090 @DSP     EQU     0642H
052D          01100 @DSPLY   EQU     052DH
1B0F          01110 @ERROR   EQU     1B0FH
1B0B          01120 @EXIT    EQU     1B0BH
1984          01130 @FEXT    EQU     1984H
196A          01140 @FLAGS   EQU     196AH
199C          01150 @FNAME   EQU     199CH
0000          01160 @FRENCH  EQU     0000H
1981          01170 @FSPEC   EQU     1981H
1874          01180 @GATRD   EQU     1874H
1875          01190 @GATWR   EQU     1875H
0000          01200 @GERMAN  EQU     0000H
0638          01210 @GET     EQU     0638H
1990          01220 @GTDCB   EQU     1990H
1A1E          01230 @GTDCT   EQU     1A1EH
19B2          01240 @GTMOD   EQU     19B2H
19E4          01250 @HDFMT   EQU     19E4H
07BD          01260 @HEX16   EQU     07BDH
07C2          01270 @HEX8    EQU     07C2H
06F6          01280 @HEXDEC  EQU     06F6H
1948          01290 @HIGH$   EQU     1948H
1897          01300 @HITRD   EQU     1897H
1898          01310 @HITWR   EQU     1898H
```

```
0000    01320 @HZ50   EQU   0000H
0086    01330 @ICNFG  EQU   0086H
198D    01340 @INIT   EQU   198DH
0000    01350 @INTL   EQU   0000H
1BF2    01360 @IPL    EQU   1BF2H
0630    01370 @JCL    EQU   0630H
0635    01380 @KBD    EQU   0635H
0628    01390 @KEY    EQU   0628H
0585    01400 @KEYIN  EQU   0585H
0089    01410 @KITSK  EQU   0089H
0089    01420 @KITSK  EQU   0089H
1CD0    01430 @KLTSK  EQU   1CD0H
1B38    01440 @LOAD   EQU   1B38H
14B3    01450 @LOC    EQU   14B3H
14DE    01460 @LOF    EQU   14DEH
0503    01470 @LOGER  EQU   0503H
0500    01480 @LOGOT  EQU   0500H
0000    01490 @MOD2   EQU   0000H
FFFF    01500 @MOD4   EQU   0FFFFH
0530    01510 @MSG    EQU   0530H
06C9    01520 @MUL16  EQU   06C9H
190A    01530 @MUL8   EQU   190AH
0066    01540 @NMI    EQU   0066H
198A    01550 @OPEN   EQU   198AH
0084    01560 @OPREG  EQU   0084H
1987    01570 @PARAM  EQU   1987H
0382    01580 @PAUSE  EQU   0382H
14A2    01590 @PEOF   EQU   14A2H
1434    01600 @POSN   EQU   1434H
0528    01610 @PRINT  EQU   0528H
063D    01620 @PRT    EQU   063DH
0645    01630 @PUT    EQU   0645H
19AC    01640 @RAMDIR EQU   19ACH
19D8    01650 @RDHDR  EQU   19D8H
19F4    01660 @RDSEC  EQU   19F4H
18D8    01670 @RDSSC  EQU   18D8H
19E0    01680 @RDTRK  EQU   19E0H
1513    01690 @READ   EQU   1513H
19A6    01700 @REMOVE EQU   19A6H
1996    01710 @RENAME EQU   1996H
149B    01720 @REW    EQU   149BH
1CD7    01730 @RMTSK  EQU   1CD7H
1CEB    01740 @RPTSK  EQU   1CEBH
1473    01750 @RREAD  EQU   1473H
19D4    01760 @RSLCT  EQU   19D4H
0000    01770 @RST00  EQU   0000H
0008    01780 @RST08  EQU   0008H
0010    01790 @RST10  EQU   0010H
0018    01800 @RST18  EQU   0018H
0020    01810 @RST20  EQU   0020H
0028    01820 @RST28  EQU   0028H
0030    01830 @RST30  EQU   0030H
0038    01840 @RST38  EQU   0038H
0FE9    01850 @RSTNMI EQU   0FE9H
19C8    01860 @RSTOR  EQU   19C8H
0680    01870 @RSTREG EQU   0680H
1B1D    01880 @RUN    EQU   1B1DH
13AD    01890 @RWRIT  EQU   13ADH
19D0    01900 @SEEK   EQU   19D0H
1421    01910 @SEEKSC EQU   1421H
1430    01920 @SKIP   EQU   1430H
```

```
19BC          Ø193Ø @SLCT    EQU     19BCH
Ø392          Ø194Ø @SOUND   EQU     Ø392H
19CC          Ø195Ø @STEPI   EQU     19CCH
Ø78D          Ø196Ø @TIME    EQU     Ø78DH
FFFF          Ø197Ø @USA     EQU     ØFFFFH
ØB99          Ø198Ø @VDCTL   EQU     ØB99H
ØD38          Ø199Ø @VDCTL3  EQU     ØD38H
156Ø          Ø2ØØØ @VER     EQU     156ØH
19DC          Ø2Ø1Ø @VRSEC   EQU     19DCH
14EC          Ø2Ø2Ø @WEOF    EQU     14ECH
1979          Ø2Ø3Ø @WHERE   EQU     1979H
1531          Ø2Ø4Ø @WRITE   EQU     1531H
19E8          Ø2Ø5Ø @WRSEC   EQU     19E8H
19EC          Ø2Ø6Ø @WRSSC   EQU     19ECH
19FØ          Ø2Ø7Ø @WRTRK   EQU     19FØH
ØD42          Ø2Ø8Ø @_VDCTL  EQU     ØD42H
ØDF1          Ø2Ø9Ø ADDR_2_ROWCOL    EQU     ØDF1H
ØØ6A          Ø21ØØ AFLAG$   EQU     ØØ6AH
1FF1          Ø211Ø AUTO?    EQU     1FF1H
Ø2Ø1          Ø212Ø BAR$     EQU     Ø2Ø1H
439D          Ø213Ø BOOTST$  EQU     439DH
1C6Ø          Ø214Ø BREAK?   EQU     1C6ØH
1C88          Ø215Ø BRKVEC$  EQU     1C88H
Ø2ØØ          Ø216Ø BUR$     EQU     Ø2ØØH
ØA7B          Ø217Ø CASHK$   EQU     ØA7BH
ØØEØ          Ø218Ø CFCB$    EQU     ØØEØH
ØØEØ          Ø219Ø CFGFCB$  EQU     ØØEØH
ØØ6C          Ø22ØØ CFLAG$   EQU     ØØ6CH
ØØ6C          Ø221Ø CFLAG$   EQU     ØØ6CH
1A7F          Ø222Ø CKMOD@   EQU     1A7FH
1568          Ø223Ø CKOPEN@  EQU     1568H
1FD6          Ø224Ø CMD_AH   EQU     1FD6H
1E81          Ø225Ø CMD_C    EQU     1E81H
2Ø8B          Ø226Ø CMD_CI   EQU     2Ø8BH
1EAB          Ø227Ø CMD_D    EQU     1EABH
1EC9          Ø228Ø CMD_DEC  EQU     1EC9H
1F82          Ø229Ø CMD_G    EQU     1F82H
1EB1          Ø23ØØ CMD_INC  EQU     1EB1H
1ECE          Ø231Ø CMD_O    EQU     1ECEH
2Ø3F          Ø232Ø CMD_R    EQU     2Ø3FH
1E9D          Ø233Ø CMD_S    EQU     1E9DH
1EA1          Ø234Ø CMD_U    EQU     1EA1H
1E9C          Ø235Ø CMD_X    EQU     1E9CH
2Ø3F          Ø236Ø CONFIG$  EQU     2Ø3FH
Ø3ØØ          Ø237Ø CORE$    DEFL    Ø3ØØH
1CFF          Ø238Ø CORE$    DEFL    1CFFH
1BFF          Ø239Ø CORE$    DEFL    1BFFH
1948          Ø24ØØ CORE$    DEFL    1948H
1948          Ø241Ø CORE$    DEFL    1948H
F8ØØ          Ø242Ø CRTBGN$  EQU     ØF8ØØH
221A          Ø243Ø CV2HEX@  EQU     221AH
16AE          Ø244Ø CYL_GRN  EQU     16AEH
1A26          Ø245Ø D@FBYT8  EQU     1A26H
ØØ33          Ø246Ø DATE$    EQU     ØØ33H
ØØ33          Ø247Ø DATE$    EQU     ØØ33H
Ø4C7          Ø248Ø DAYTBL$  EQU     Ø4C7H
ØØAØ          Ø249Ø DBGSV$   EQU     ØØAØH
ØØ31          Ø25ØØ DCBKL$   EQU     ØØ31H
Ø47Ø          Ø251Ø DCT$     EQU     Ø47ØH
1A29          Ø252Ø DCTBYT8@         EQU     1A29H
1A34          Ø253Ø DCTFLD@  EQU     1A34H
```

```
006D  02540 DFLAG$  EQU  006DH
006D  02550 DFLAG$  EQU  006DH
2300  02560 DIRBUF$ EQU  2300H
0846  02570 DIS_DO_RAM   EQU  0846H
0B94  02580 DODATA$ EQU  0B94H
0210  02590 DODCB$  EQU  0210H
0C44  02600 DO_CONTROL    EQU  0C44H
0CB8  02610 DO_DSPCHAR    EQU  0CB8H
0C8C  02620 DO_INVERT_DIS EQU  0C8CH
0C89  02630 DO_INVERT_ENA EQU  0C89H
0C9B  02640 DO_INVERT_OFF EQU  0C9BH
0000  02650 DO_MASK EQU  0000H
0BCB  02660 DO_RET  EQU  0BCBH
0BCC  02670 DO_RET1 EQU  0BCCH
0CCE  02680 DO_SCROLL     EQU  0CCEH
0BEA  02690 DO_TABS EQU  0BEAH
04C0  02700 DSKTYP$ EQU  04C0H
201B  02710 DSPASC@ EQU  201BH
04C2  02720 DTPMT$  EQU  04C2H
0FF4  02730 DVREND$ EQU  0FF4H
0206  02740 DVRHI$  EQU  0206H
2150  02750 ED_TAB  EQU  2150H
006E  02760 EFLAG$  EQU  006EH
0817  02770 ENADIS_DO_RAM  EQU  0817H
19A4  02780 EXTDBG$ EQU  19A4H
000E  02790 FDDINT$ EQU  000EH
000E  02800 FDDINT$ EQU  000EH
006F  02810 FEMSK$  EQU  006FH
006A  02820 FLGTAB$ EQU  006AH
006A  02830 FLGTAB$ EQU  006AH
2031  02840 GETASC@ EQU  2031H
0DAE  02850 GET_@_ROWCOL  EQU  0DAEH
0750  02860 HERTZ$  EQU  0750H
21E4  02870 HEXIN@  EQU  21E4H
040E  02880 HIGH$   EQU  040EH
1A6C  02890 HKRES$  EQU  1A6CH
0072  02900 IFLAG$  EQU  0072H
0072  02910 IFLAG$  EQU  0072H
0420  02920 INBUF$  EQU  0420H
21D5  02930 INPUC@  EQU  21D5H
21C9  02940 INPUT@  EQU  21C9H
003C  02950 INTIM$  EQU  003CH
003D  02960 INTMSK$ EQU  003DH
003E  02970 INTVC$  EQU  003EH
003E  02980 INTVC$  EQU  003EH
0203  02990 JCLCB$  EQU  0203H
0024  03000 JDCB$   EQU  0024H
00C0  03010 JFCB$   EQU  00C0H
0230  03020 JLDCB$  EQU  0230H
0026  03030 JRET$   EQU  0026H
07D6  03040 KCK@    EQU  07D6H
0074  03050 KFLAG$  EQU  0074H
0074  03060 KFLAG$  EQU  0074H
08FC  03070 KIDATA$ EQU  08FCH
0208  03080 KIDCB$  EQU  0208H
0202  03090 LBANK$  EQU  0202H
0023  03100 LDRV$   EQU  0023H
0075  03110 LFLAG$  EQU  0075H
1566  03120 LNKFCB@ EQU  1566H
001E  03130 LOW$    EQU  001EH
000D  03140 LSVC$   EQU  000DH
```

```
2400          03150 MAXCOR$ EQU     2400H
0401          03160 MAXDAY$ EQU     0401H
3000          03170 MINCOR$ EQU     3000H
0076          03180 MODOUT$ EQU     0076H
0076          03190 MODOUT$ EQU     0076H
04DC          03200 MONTBL$ EQU     04DCH
0077          03210 NFLAG$  EQU     0077H
0078          03220 OPREG$  EQU     0078H
0078          03230 OPREG$  EQU     0078H
086E          03240 OPREG_SV_AREA   EQU     086EH
0835          03250 OPREG_SV_PTR    EQU     0835H
211F          03260 OP_TAB  EQU     211FH
14DC          03270 ORARET@ EQU     14DCH
003B          03280 OSRLS$  EQU     003BH
0085          03290 OSVER$  EQU     0085H
0069          03300 OVRLY$  EQU     0069H
0410          03310 PAKNAM$ EQU     0410H
0382          03320 PAUSE@  EQU     0382H
07AF          03330 PCSAVE$ EQU     07AFH
001B          03340 PDRV$   EQU     001BH
001B          03350 PDRV$   EQU     001BH
001C          03360 PHIGH$  EQU     001CH
0218          03370 PRDCB$  EQU     0218H
0DCD          03380 PUTA@DE EQU     0DCDH
0DCA          03390 PUT_@   EQU     0DCAH
0DC6          03400 PUT_@_ROWCOL    EQU     0DC6H
007B          03410 RFLAG$  EQU     007BH
007B          03420 RFLAG$  EQU     007BH
0DD0          03430 ROWCOL_2_ADDR   EQU     0DD0H
1BFF          03440 RST38@  EQU     1BFFH
04C4          03450 RSTOR$  EQU     04C4H
13A2          03460 RWRIT@  EQU     13A2H
0238          03470 S1DCB$  EQU     0238H
1D00          03480 SBUFF$  EQU     1D00H
1A79          03490 SET@EXEC        EQU     1A79H
0CF3          03500 SET_SCROLL      EQU     0CF3H
008C          03510 SFCB$   EQU     008CH
007C          03520 SFLAG$  EQU     007CH
007C          03530 SFLAG$  EQU     007CH
0220          03540 SIDCB$  EQU     0220H
0228          03550 SODCB$  EQU     0228H
2142          03560 SPACE4$ EQU     2142H
0380          03570 STACK$  EQU     0380H
0000          03580 START$  EQU     0000H
0000          03590 START$  EQU     0000H
000B          03600 SVCRET$ EQU     000BH
0100          03610 SVCTAB$ EQU     0100H
1B13          03620 SYSERR$ EQU     1B13H
004E          03630 TCB$    EQU     004EH
007D          03640 TFLAG$  EQU     007DH
002D          03650 TIME$   EQU     002DH
002D          03660 TIME$   EQU     002DH
002C          03670 TIMER$  EQU     002CH
002C          03680 TIMER$  EQU     002CH
002B          03690 TIMSL$  EQU     002BH
002B          03700 TIMSL$  EQU     002BH
0713          03710 TIMTSK$ EQU     0713H
04C3          03720 TMPMT$  EQU     04C3H
07B1          03730 TRACE_INT       EQU     07B1H
0A8F          03740 TYPHK$  EQU     0A8FH
0B26          03750 TYPTSK$ EQU     0B26H
```

```
0013          0376 USTOR$   EQU     0013H
007F          0377 VFLAG$   EQU     007FH
007F          0378 VFLAG$   EQU     007FH
2211          0379 WR1HEX@  EQU     2211H
2215          0380 WR2HEX@  EQU     2215H
0080          0381 WRINT$   EQU     0080H
2231          0382 WRSPA@   EQU     2231H
2157          0383 XY_TAB   EQU     2157H
0401          0384 ZERO$    EQU     0401H
13A0          0385 ZEROA@   EQU     13A0H
No end statement
00000 Total errors
```